RAYPACK®

Software Packaging

User Guide RayPack
6.5

RayPack is part of RaySuite.

•rayNET

User Guide RayPack

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

Contents

Introduction

RayPack is a framework for the creation and management of software packages that can be deployed to target machines. It is designed to support a broad variety of package formats, target operating systems, and deployment systems.

The RayPack components allow enterprises to implement well-structured processes, which control package evaluation, creation, manipulation, import, export, validation, storage, and deployment. At the present time, RayPack consists of six main components, PackDesigner, PackTailor, PackRecorder, PackBot, PackWrapper,and PackBench. The documentation of the PackBench can be found in in the *PackBench User Guide* and it is not part of the *<% PRODUCTNAME%> User Guide*.

With the current release 6.5, Raynet extends the core components of the framework, which allow users to create MSI packages from scratch or build them upon the result of installation capturing processes on Windows platforms. It is also possible to extend standard installations with transforms (MST), create patches for fixes and small / minor updates (MSP), or edit their native contents.

Supplementary information can be obtained from the following resources:

Online documentation for Microsoft MSI technology
Online documentation for Microsoft App-V 4.x technology
Online documentation for Microsoft App-V 5.x technology
Online documentation for Symantec Workspace Virtualization technology
Online documentation for VMware ThinApp

Tip:
Various online references are used throughout this document. Although an online connection is not required for this help file, further reading and reference material is available via external links that are present in this document, provided there is an internet connection.

Core components overview

As previously mentioned, there are currently three main components provided by the RayPack framework:

- PackRecorder
- PackDesigner
- PackTailor
- PackBench (the documentation can be found in the *PackBench User Guide*)

The diagram below displays typical packaging procedures performed with these components, including the possible input resource types, matching project file formats per component (RCP in PackRecorder and RPP in PackDesigner), and the build options for output generation:
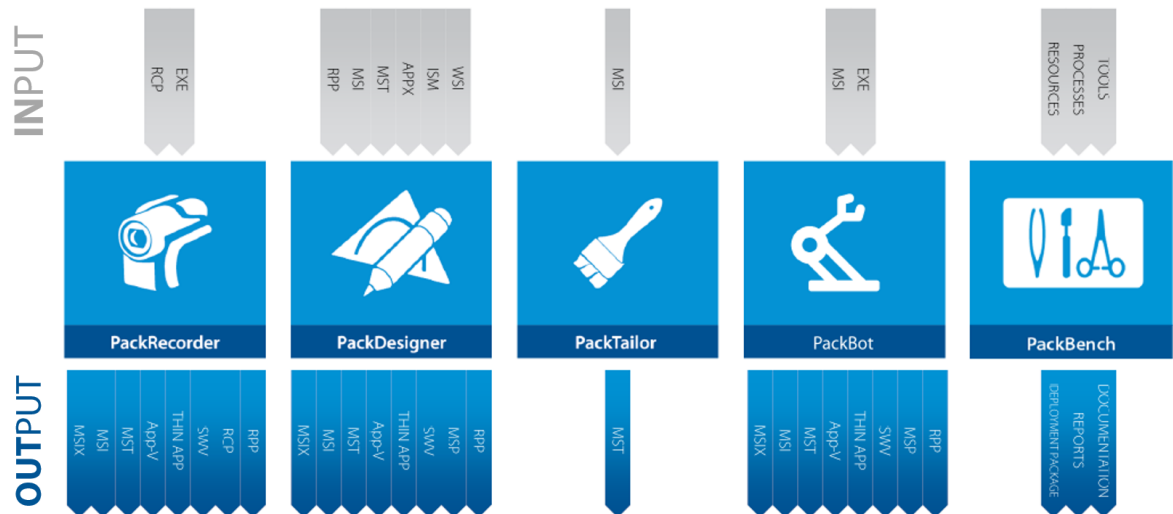
Please refer to the explicit component chapters for each tool in order to get further details about the mechanisms and interfaces used to accomplish typical packaging related tasks.

Development Roadmap

Upcoming releases will introduce additional new components and features, resulting in boosted productivity, saving time and resources, and overall improvement of the product experience. The development of RayPack is partner and customer oriented and driven; therefore, any ideas or suggestions for further development or to meet your specific business needs are not only welcomed, but encouraged. Simply let us know your idea and then watch it come into fruition before your very eyes. Our sales team will be happy to help you.

Also make sure to check our website http://raynet.de to never miss upcoming releases, announcements, special offers and product trainings.

System Requirements

The given requirements name prerequisites for devices running the RayPack framework.

Hardware Requirements

Minimal

- CPU: Intel Core i5
- Screen resolution: 1024 x 768 pixels
- RAM: 4GB
- Disk space: 10GB

Recommended

- CPU: Intel Core i7
- Screen resolution: 1280 x 1024 pixels
- RAM: 16GB or higher
- Disk space: 100GB or more

> **Note:**
> The installation of the RayPack framework itself requires about 300MB of disk space. The amount of additional space required depends on the amount of packaging material and the location of your data store.

Supported OS

The following represents the list of supported operating systems at the time of release.

- Windows Vista SP2
- Windows 7 SP1
- Windows 8
- Windows 8.1
- Windows 10
- Windows Server 2008 R2
- Windows Server 2008 SP1
- Windows Server 2012
- Windows Server 2012 R2
- Windows Server 2016

- Windows Server 2019

> 📎 **Note:**
> Using PackBot for the bulk virtualization to App--V 4.6 format requires that the Guest Operating System (virtual) meets the Microsoft App-V 4.6 Sequencer requirements. Thus, this type of conversion is not supported on Windows 10.

Prerequisite Software

General

- .NET 4.5 Client & Full for Windows Vista up to Windows 8 systems (both 32-bit and 64-bit). Windows 10 already contains the required framework.

Virtualization Pack

- To create SWV packages, the Symantec Workspace Virtualization Agent 7.5 has to be installed on the packaging machine.
- To create Thin-App packages, the VMware ThinApp must be installed on the packaging machine.

Generation of MSIX Files

If using Windows 8.1 or Windows Server 2012 R2 or an older version of Windows or Windows Server, an update for the CRT in Windows is needed in order to be able to generate MSIX files.

More information on the CRT update can be found here: https://support.microsoft.com/en-us/help/2999226/update-for-universal-c-runtime-in-windows.

RayFlow integration

To use the RayFlow functionality directly from RayPack, a running RayFlow server has to be accessible.

Compatibility and Quality Control

To use Quality features (checklists, compatibility, virtualization, and conflict testing) RayQC and / or RayQC Advanced have to be installed on the local machine.

Sequencing to App-V 4.6 / App-V 5.x Using PackBot

To convert legacy setups to the Microsoft App-V 4.6 / 5.x format using a virtual machine, the virtual machine must have the Microsoft App-V Sequencer installed. Additional requirements for specific operating system versions / platforms may by required by the Microsoft Sequencer tools.

Converting to Thin-App Using PackBot

To convert legacy setups to Thin-App, the Thin-App converter must be installed either on the host or on the virtual machine.

Hyper-V integration

- Both host and guest machine must have PowerShell 3.0 or newer installed.
- Windows Remote Management

- RayPack Studio Tools for Hyper-V need to be installed on the guest machine.

The tools can be installed from a Windows Installer package that is present in the RayPack subfolder `Tools\HyperVTools\Packaging Suite Tools for Hyper-V.msi`.
The installation of the tools is required, so that the user can see interactive prompts and windows on Hyper-V machines. It is recommended to install the tools as a part of the base snapshot.

VMware Workstation / ESXi5.5 - 6.0

RayPack supports the following products:

- VMware vSphere 5.5-6.0
- VMware Workstation 10 and newer
- VMware Workstation 7, 8, 9 and for VMware vSphere 4.x, 5 and 5.1 are underlined experimentally supported.

To use any of VMware Workstation / ESXi machines, one of the following must be installed in an appropriate version:

- VMware Workstation
- VMware VIX API (https://my.vmware.com/web/vmware/details?productId=26&downloadGroup=VIX-API-162)
- vSphere

The required VIX API version depends on the systems that it needs to connect to. The below table presents the supported versions of VMware products depending on the installed VIX API version.

VIX API Version	VMware Platform Products	Library Location
1.11	Workstation 8 or earlier	Workstation-8.0.0-and-vSphere-5.0.0
1.12	Workstation 9 or earlier	Workstation-9.0.0-and-vSphere-5.1.0
1.13	Workstation 10 or earlier	Workstation-10.0.0-and-vSphere-5.5.0
1.14	Workstation 11 or earlier	Workstation-11.0.0
1.15.0	Workstation 14 or earlier	Workstation-12.0.0 Workstation-14.0.0

ESXi 6.5 and newer

To make use of ESXi 6.5+ servers, the following prerequisites must be met:

- PowerShell 3.0
- PowerShell Execution Policy set to Unrestricted or RemoteSigned
- PowerCLI installer (https://www.powershellgallery.com/packages/VMware.PowerCLI/11.2.0.12483598)
- VMware Tools installed on the VM
- **Guest operations** and **System** permissions granted to the user executing the product.

Combination of supported versions is presented in the following table:

	VMware PowerCLI															
	12.0.0	11.5.0	11.4.0	11.3.0	11.2.0	11.1.0	11.0.0	10.2.0	10.1.1	10.1.0	10.0.0	6.5.4	6.5.3	6.5.2	6.5.1	6.5.0
VMware vSphere Hypervisor (ESXi)																
7.0	✅	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
6.7 U3	✅	✅	✅	✅	—	—	—	—	—	—	—	—	—	—	—	—
6.7 U2	✅	✅	✅	✅	✅	—	—	—	—	—	—	—	—	—	—	—
6.7 U1	✅	✅	✅	✅	✅	✅	✅	—	—	—	—	—	—	—	—	—
6.7.0	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	—	—	—	—	—	—
6.5 U3	✅	✅	✅	✅	—	—	—	—	—	—	—	—	—	—	—	—
6.5 U2	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	—	—	—	—	—	—
6.5 U1	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	—	—
6.5.0	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅
6.0 U3	—	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅
6.0.0 U2	—	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅
6.0.0 U1	—	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅
6.0.0	—	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅
5.5 U3	—	—	—	—	—	—	—	✅	✅	✅	✅	✅	✅	✅	✅	✅
5.5 U2	—	—	—	—	—	—	—	✅	✅	✅	✅	✅	✅	✅	✅	✅
5.5 U1	—	—	—	—	—	—	—	✅	✅	✅	✅	✅	✅	✅	✅	✅
5.5	—	—	—	—	—	—	—	✅	✅	✅	✅	✅	✅	✅	✅	✅

More information about PowerCLI:

- https://pubs.vmware.com/vsphere-51/index.jsp?topic=%2Fcom.vmware.powercli.cmdletref.doc%2FGet-VMGuest.html
- https://pubs.vmware.com/vsphere-51/topic/com.vmware.powercli.cmdletref.doc/Invoke-VMScript.html
- https://pubs.vmware.com/vsphere-50/index.jsp?topic=%2Fcom.vmware.wssdk.pg.doc_50%2FPG_ChD_Privileges_Reference.22.3.html

Product Installation

Preparing the Installation

The RayPack installation resources come along as an MSI file with additional documents, such as the current Release Notes and a Product Activation Guide. Before the application is installed on a device, some preparations are needed:

1. Please make sure to have your order number or license file at hand. Either one will be provided by our support team, or your Raynet sales representative.

2. The packaging machine has to meet the system requirements described within the previous chapter.

3. A Windows User with sufficient rights for installations has to be logged in.

4. Close all dispensable applications during the setup routine execution.

Installing RayPack

Launch the RayPack setup with a double-click on the MSI file and wait for the Welcome Screen to be prepared.

Click **Next >** to proceed.
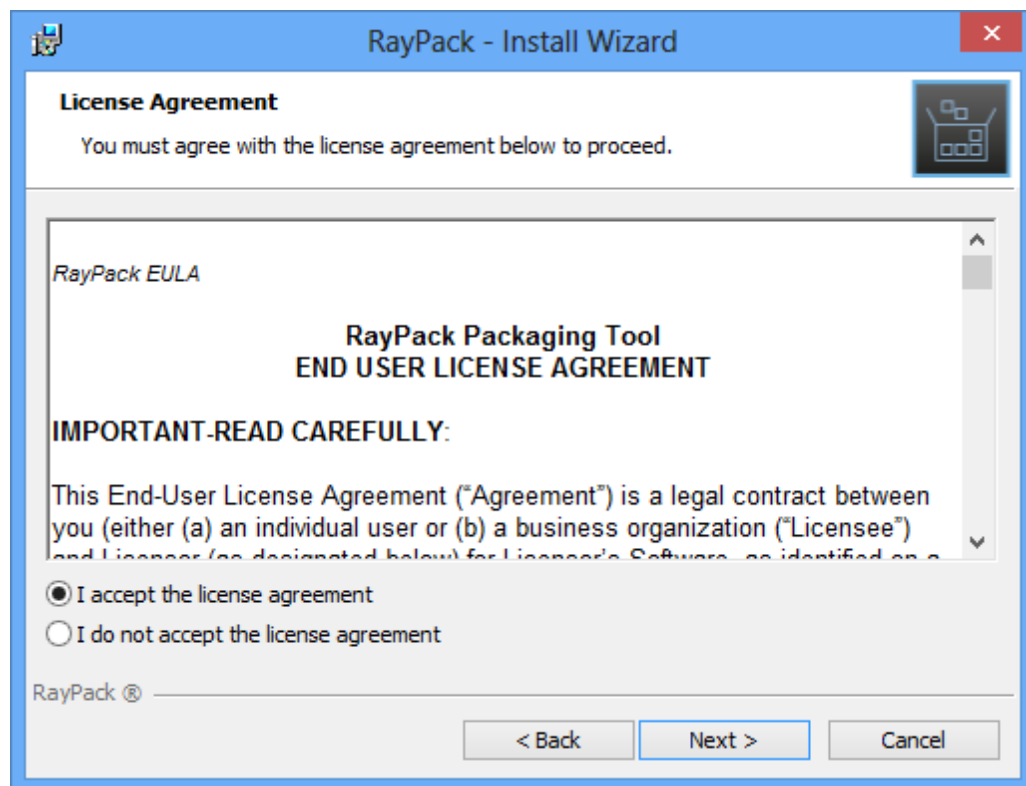
In order to be allowed to install RayPack, the End User License Agreement has to be accepted. To do so, select the upper radio button and click **Next >** to proceed.
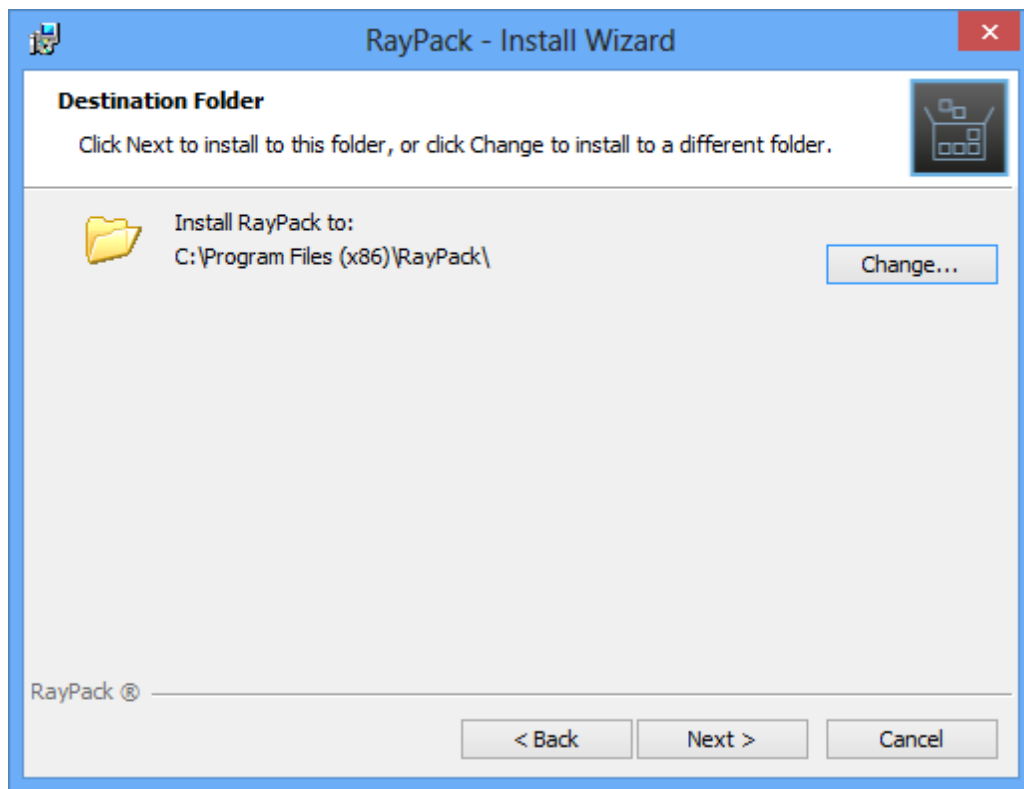
Please choose the destination folder by either keeping the suggested default or clicking on the **Change...** button to select another target directory.
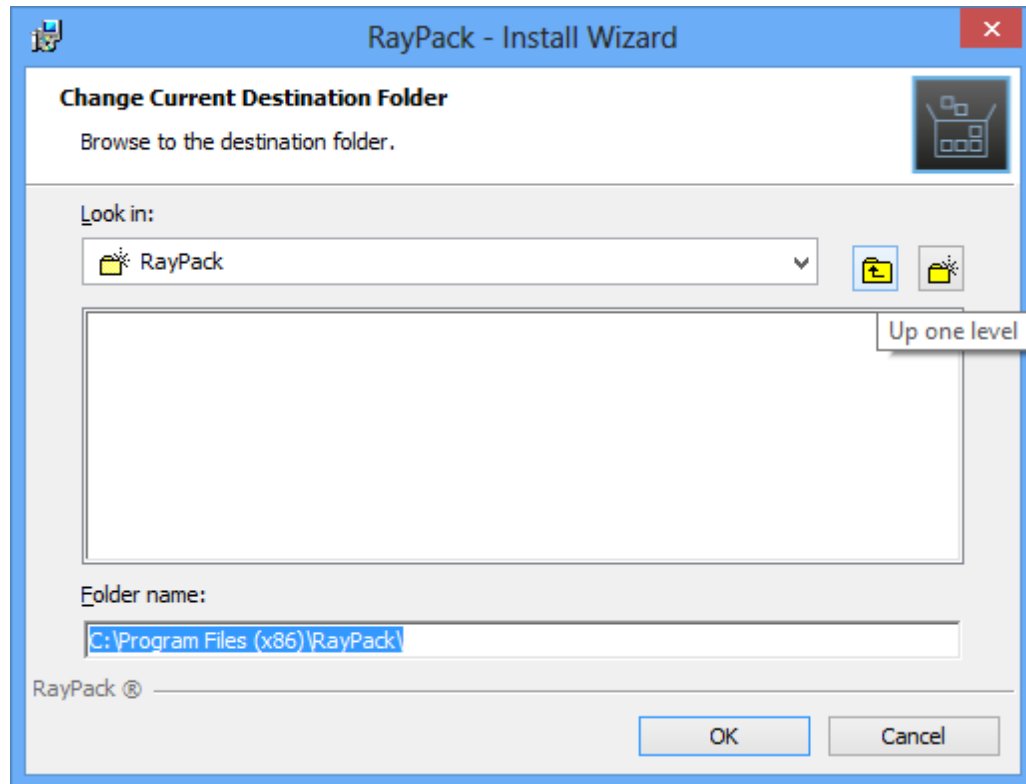
If a custom destination folder has to be defined, the dialog displayed above has to be used. Please use the icons to navigate to the desired installation directory or to create a new one where necessary. As soon as the right target is defined, click **OK** to return to the Destination Folder screen. It will be updated to display the custom destination recently selected.

On the Destination Folder screen, click **Next >** to proceed.

The Setup Type screen is displayed, presenting the three options available for RayPack installations:

- **Typical** installs the full RayPack application with all additional resources provided along with the software core.

- **Standalone** installs the Standalone repackager, which may be run on clean repackaging machines without product activation. The standalone mode allows to capture setups, and transfer the results into RPP files. To perform extended packaging tasks, or to tailor a transform, another machine with a full RayPack instance is required.

- **Custom** allows to remove some additional resources from the installation, such as the Merge Module Library and the Example files provided for a smooth start into packaging with our product.

Please select the desired option and click **Next >** to proceed. The content of the upcoming step depends on the selected type:

Setup Type: Standalone

RayPack is already prepared to be installed in the Standalone Repackager Mode.
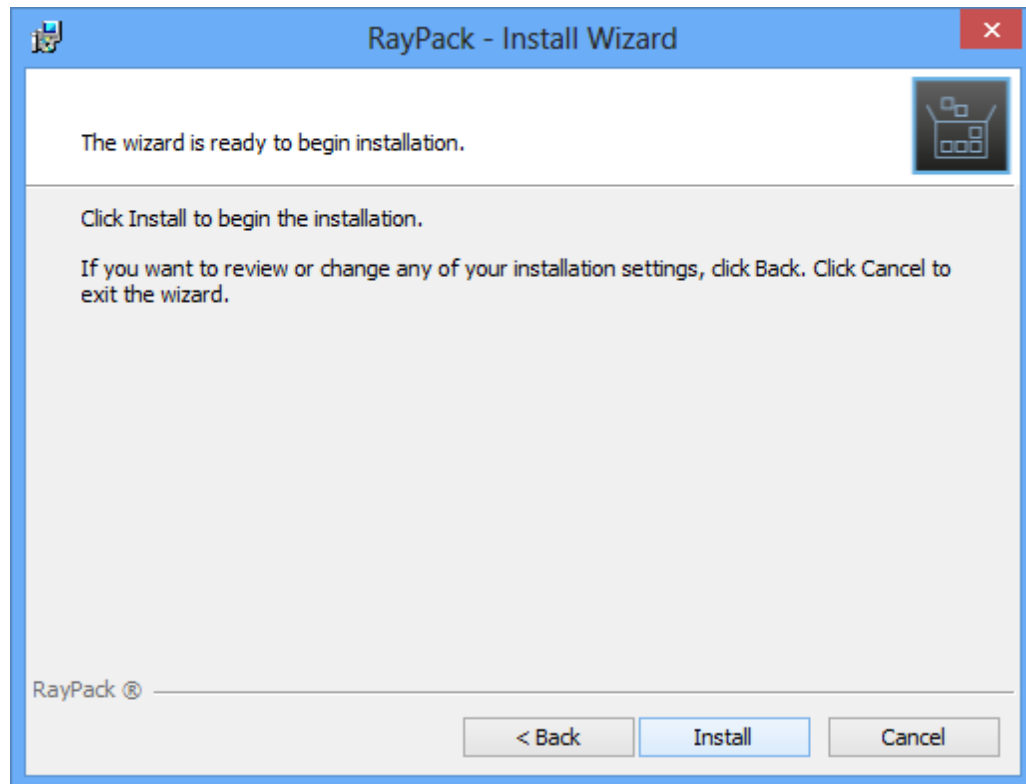
Click **Install** to start the installation. Please skip the following instructions and jump directly to the installation progress indicator dialog description.

Setup Type: Custom

The custom setup screen is displayed, presenting the application features. To remove a feature from the installation, expand the installation status modificator menu as displayed within the screenshot below, and then select "**This feature will not be available.**"
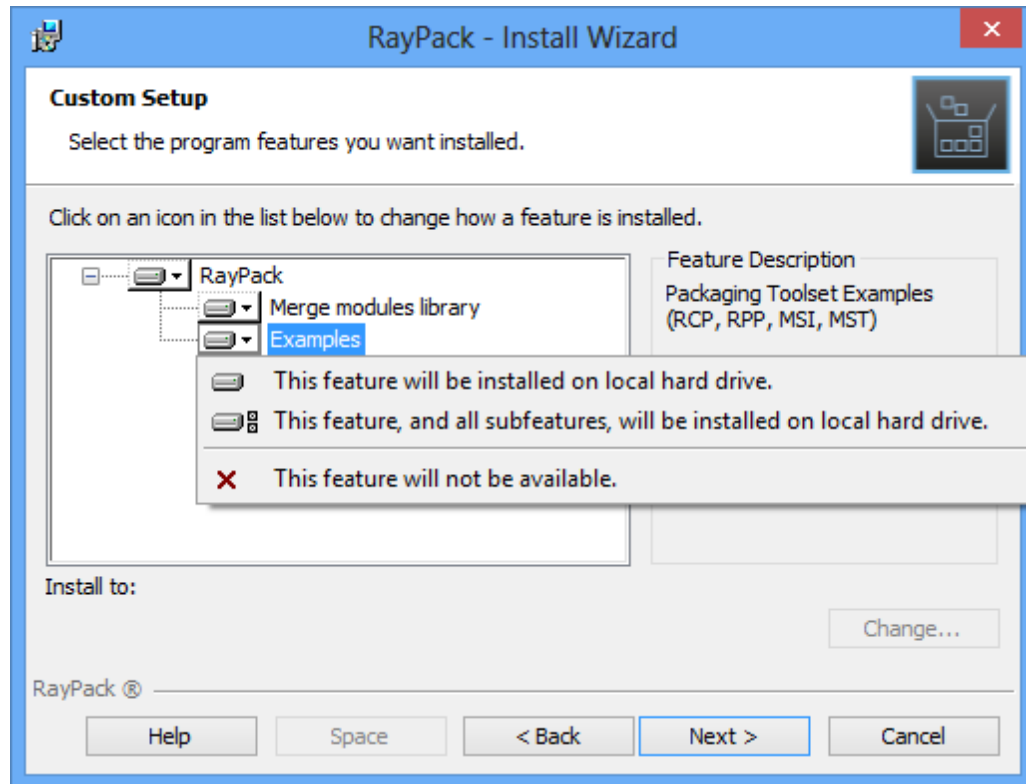
Click **Next >** to proceed.

The PackPoint Configuration screen is displayed.
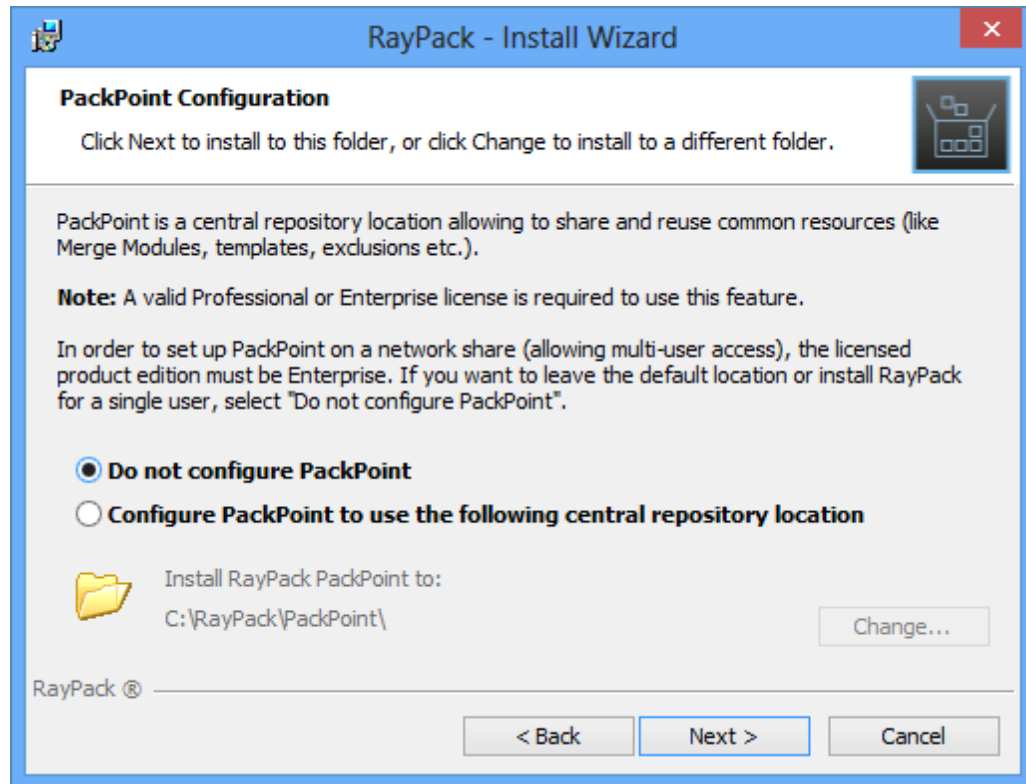
Using PackPoint is optional, and RayPack is fully operational without this central repository for enterprise settings and configuration resources. However, using PackPoint is recommended for packaging factories and team oriented packaging processes. It helps to preserve customizations beyond product upgrades and the lifetime of local instance installations.

Please keep in mind that accepting the EULA includes accepting the license limitations for PackPoint as well. If the decision is made to use the PackPoint resource organization unit, the location of the PackPoint directory needs to comply with your licensed product edition:

- PackPoint may not be used for instances with Standard Editions.
- PackPoint may be used from a local directory with Professional Editions.
- PackPoint may be located locally or remotely for all Editions above Professional.

If PackPoint is about to be used for the new RayPack instance, it has to be configured. To do so, activate the radio option "**Configure PackPoint to use the following central repository location**". Please choose the PackPoint destination folder by either keeping the suggested default or clicking on the **Change...** button to select another target directory.

If a custom target directory definition is required, use the icons available within the Define custom target destination screen to navigate to the desired installation directory or to create a new one where necessary. As soon as the right target is defined, click **OK** to return to the PackPoint Configuration screen. It has now been updated to display the custom destination just selected.

Click **Next >** to proceed.
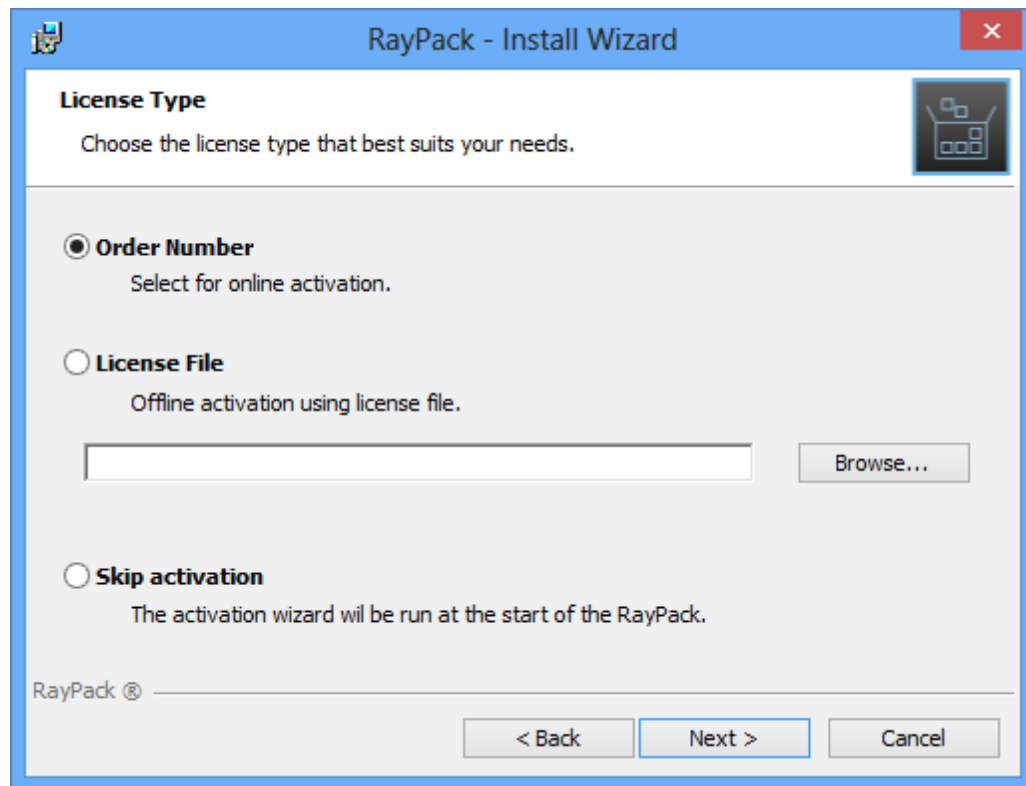
The License Type screen provides options to either activate the RayPack instance via order number and online activation service, by using an already prepared license file (`*.rsl`), or to skip activation for now. If the activation is skipped, it will need to be performed later when RayPack is launched.

To use an already existing license file, which most likely has been provided by the Raynet support team, the **Browse...** button has to be clicked. Use the controls of the system browser dialog to navigate to the `*.license` file and select it with a click on the **Open** button.

Click **Next >** to proceed.

If activation by order number has been selected during the previous step, the Customer Information screen is displayed:

Please enter your individual RayPack Order number and provide required user information, such as E-Mail, user name, and company name. The information will be used to verify the order number during the upcoming installation execution procedure.

Click **Next >** to proceed.
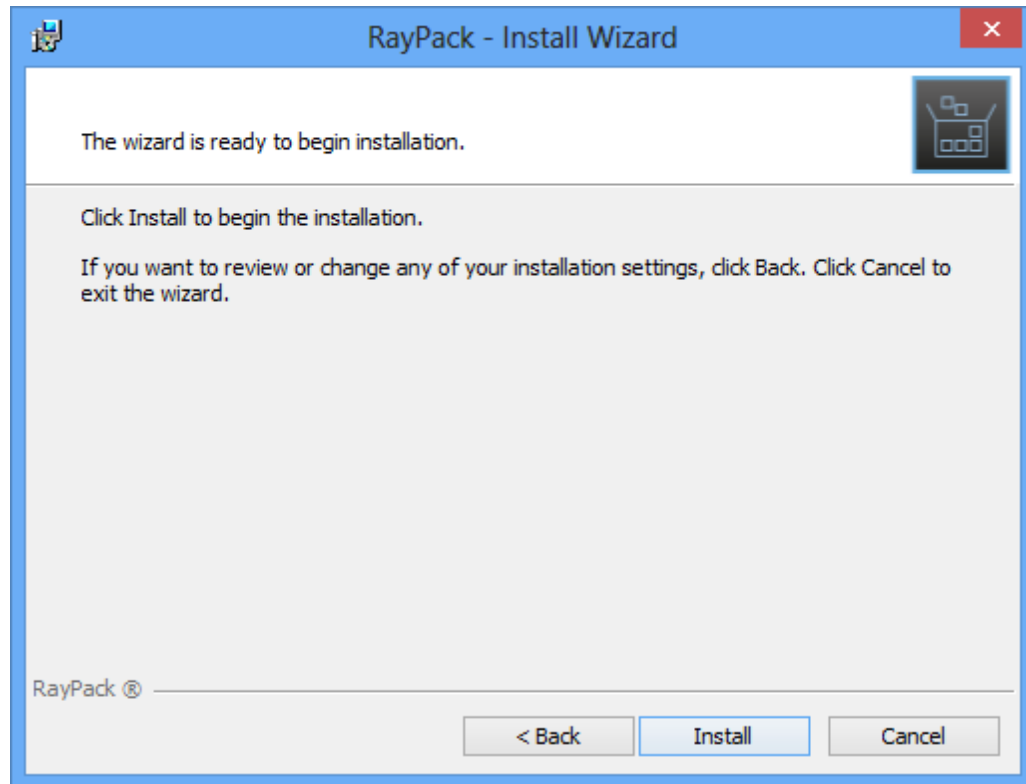
All required settings and properties are defined – RayPack is ready to be installed. Click **Install** to start the process.

A Progress Indication dialog is displayed as long as the installation steps are executed.
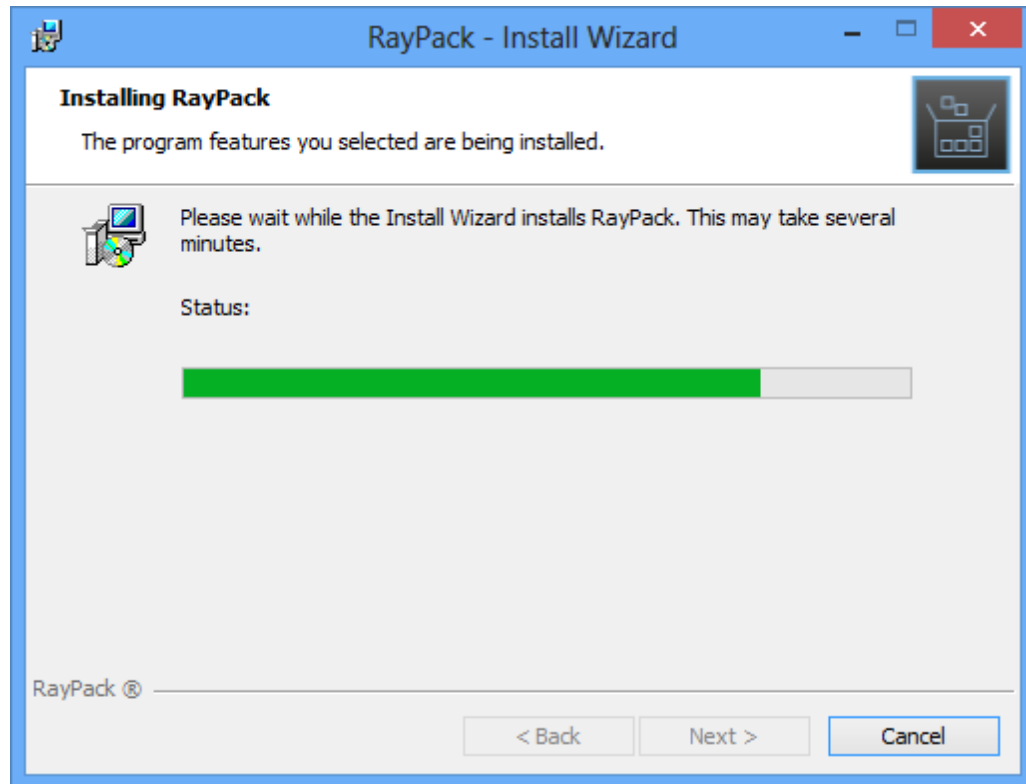
Whilst the installation is executed, Windows may present a User Account Control message like the one shown below:
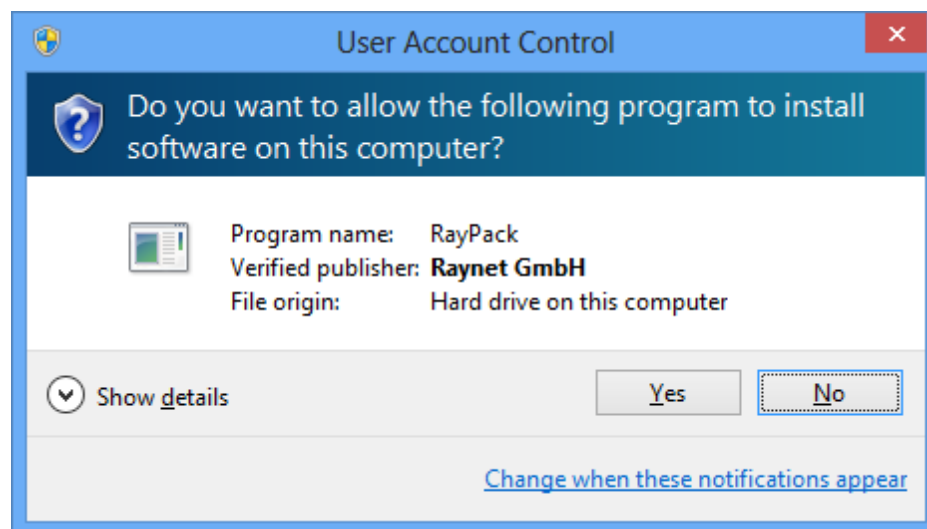
Click **Yes** to perform the installation.
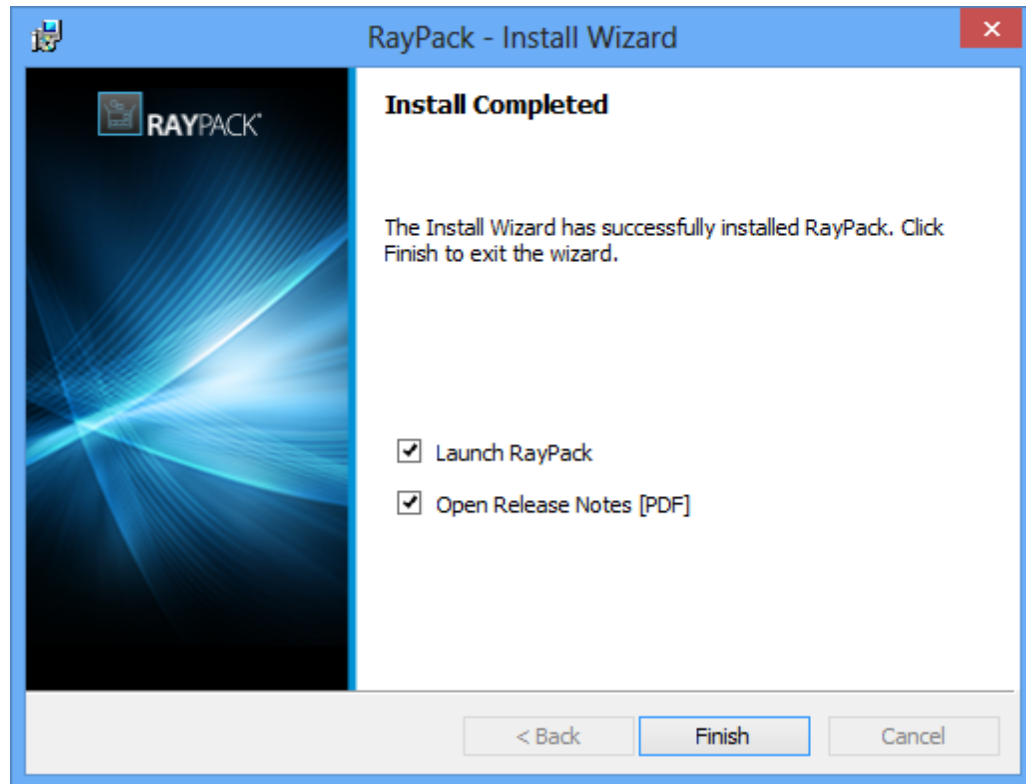
As soon as all required measures are done, the Install Completed dialog is presented. Activate the checkboxes to immediately launch RayPack and to read the Release Notes with essential information about the requirements of the installed product version, as well as listings of new features and issues which have been solved or documented.

Click **Finish** to exit the setup.

If the "Launch RayPack" option has been activated, the application is loaded immediately. During the software launch, another Windows User Account Control message may be displayed.
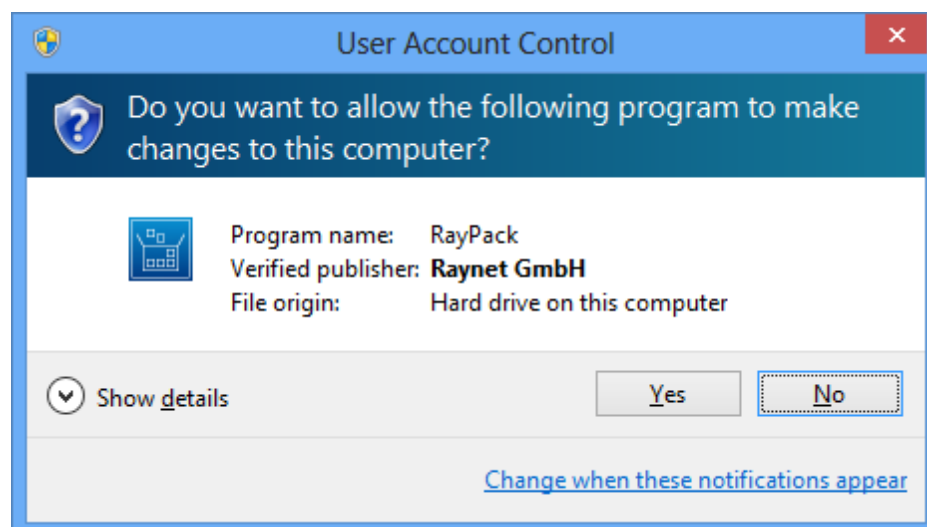
Please click **Yes** to proceed if this message occurs.

If the product activation has been executed already, there will be a RayPack loading screen whilst all required application resources are prepared.

If the RaySuite License Activation Tool is shown instead, there has either been no activation during setup or it failed. Please use the options provided by the License Activation Tool interface to handle the different product activation options and measures. Further information regarding this process is available from the License wizard section later on.

As soon as the Home Screen of RayPack is displayed, your packaging tool is ready for action.

Product Activation

The product can be activated using one of the following methods:

- Directly within installation
  - o By supplying the order number
  - o By supplying an already generated license file (`.rsl` format)
- When the product is started for the first time.

If RayPack detects that no valid license is present on start-up, the license activation wizard will be shown after starting the main executable. The tool can be also started manually, by executing `Raynet.LicenseActivation.exe` from the main installation folder.

License Wizard

This section describes the usage of the licensing wizard.

On the initial start of RayPack, the licensing wizard is shown. If the need to transfer an existing license arises, the license wizard can be started manually. There are a variety of ways in which a license can be activated and below they are described in detail.
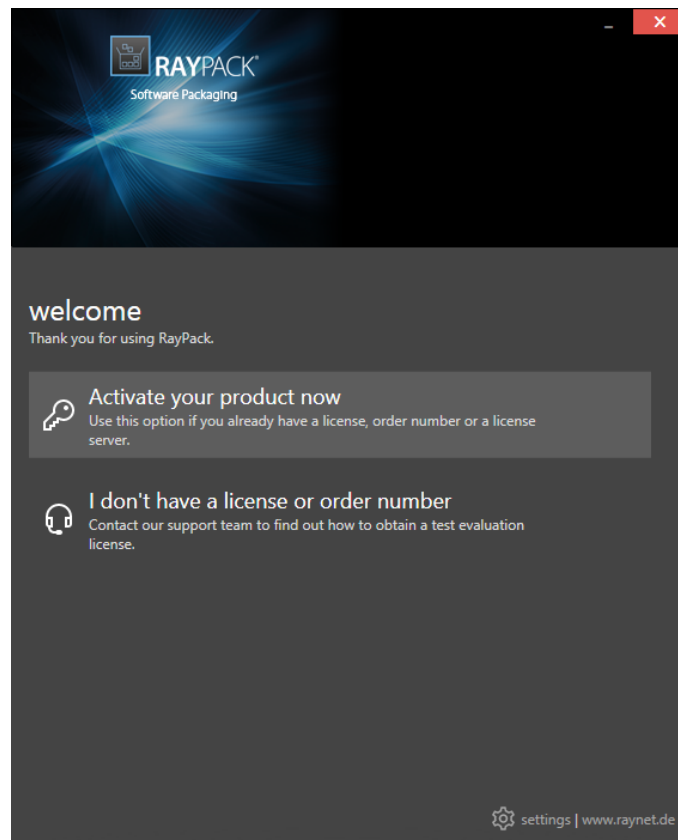
First time activation
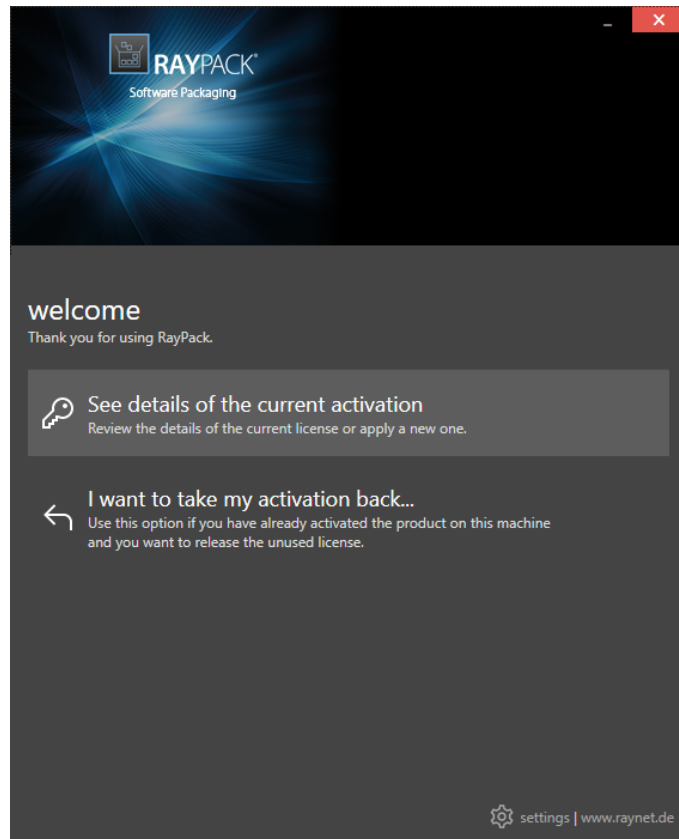
The main screen when the product has been already
activated

Activate your product now

This option should be used to activate the product using one of the following methods:
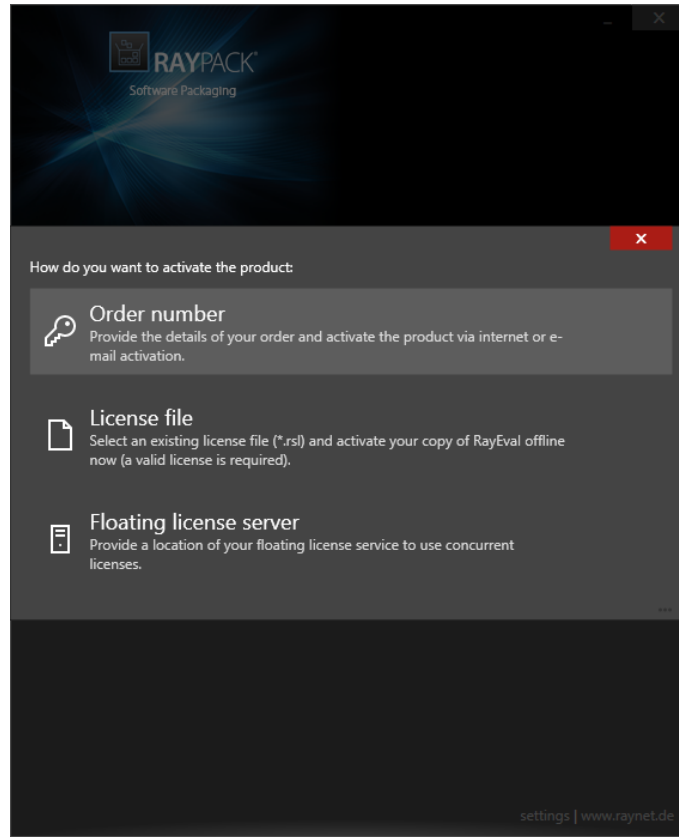
- **Order number**
  Online activation using a valid order number received from Raynet (recommended for most users)

- **License file**
  Offline activation using a license file (`.rsl`) received from Raynet

- **Floating license server**
  Activation using a local floating license server.

See details of the current activation
This options shows the details of the current activation. This option is only visible if the product has been already activated.

I don't have a license or order number
Choose this option if there is neither a license nor order number. For in-depth information please read this section. This option is only visible if the product has not been activated yet.

I want to take my activation back...
Use this option to deactivate a currently licensed version of RayPack. For in-depth information please read this section. This options is only visible if the product has been already activated.

Once the license file has been generated or copied to the correct location the following will be shown...
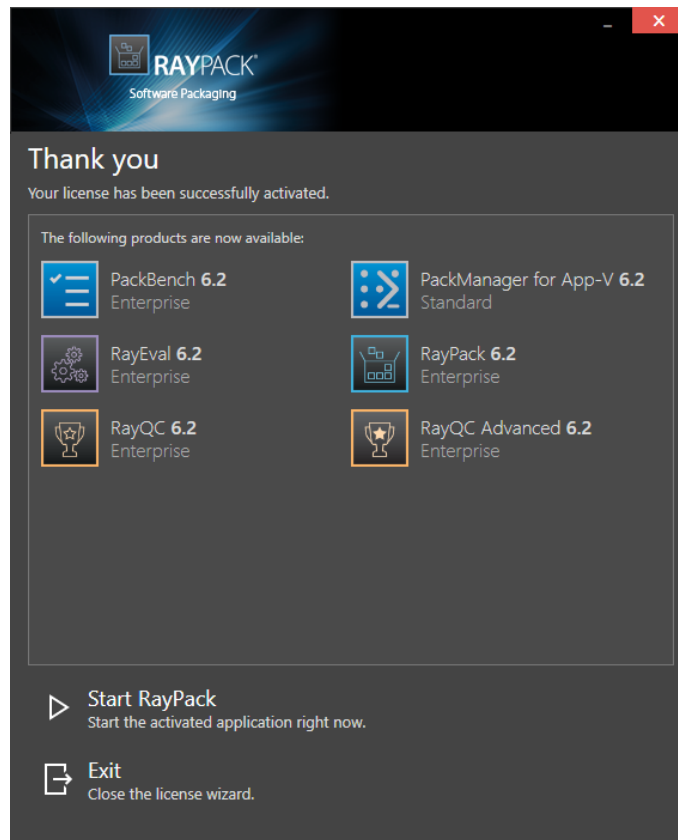
> **Note:**
> Depending on the license, more available products may be shown, as pictured above.
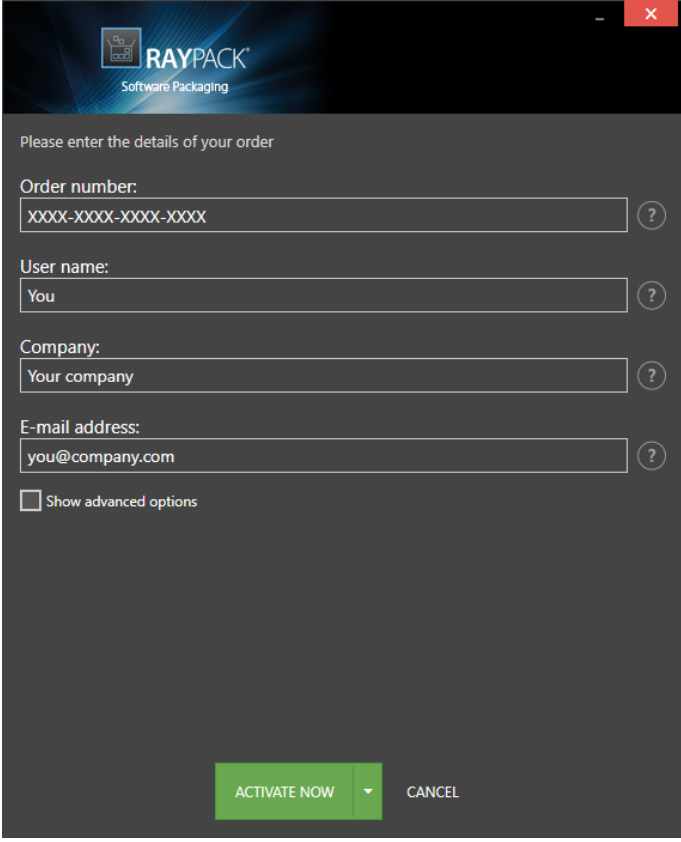
Then the option of starting RayPack or just closing the activation wizard is made available.

Troubleshooting

If any issues arise during the activation process, please contact our help desk to receive assistance in activating RayPack.

Order Number

RayPack can be activated either directly online or via email once the order number has been delivered. The activation process generates a license file (`*.rsl`) that is created (or must be copied) to the installation directory of RayPack (in the same location as the `RayPack.exe`). When performing an online activation, sufficient permissions must be readily available to allow the creation of the license file in the installation directory. The activation **binds** the license to the machine on which it was activated on. This is the only time that an active connection to the internet is required (if activating online).

Choosing the **ACTIVATE NOW** button, connects to the Raynet license server using the information provided and will dynamically generate a license file. Choosing the **ACTIVATE MANUALLY** button will open a dialog as shown here. Choosing the **CANCEL** button will abort the activation process.

Order Details

Order number:
This is the unique order number received when RayPack has been purchased. If it is necessary to recover the order number, please contact our sales team.

User name:
This is the name of the user that is activating RayPack. It does not need to be the same name used to order RayPack.

Company:
This is the name of the company for which RayPack will be licensed. This name will appear in the License and Edition view of RayPack.

E-mail address:
This is the email address of the person that performs the activation. We respect the privacy of our customers, this email address will only be used by Raynet and only when there are any

problems or important information regarding the license.

Advanced Options

On choosing the advanced options check box, extended information and possibilities of the licensing and activation of RayPack are shown.

Hardware ID:
This is an ID calculated based on the hardware on which the activation is taking place on. The ID is unique, but cannot be used to personally identify a user. It is used to generate the license for the machine on which the activation process is carried out on.
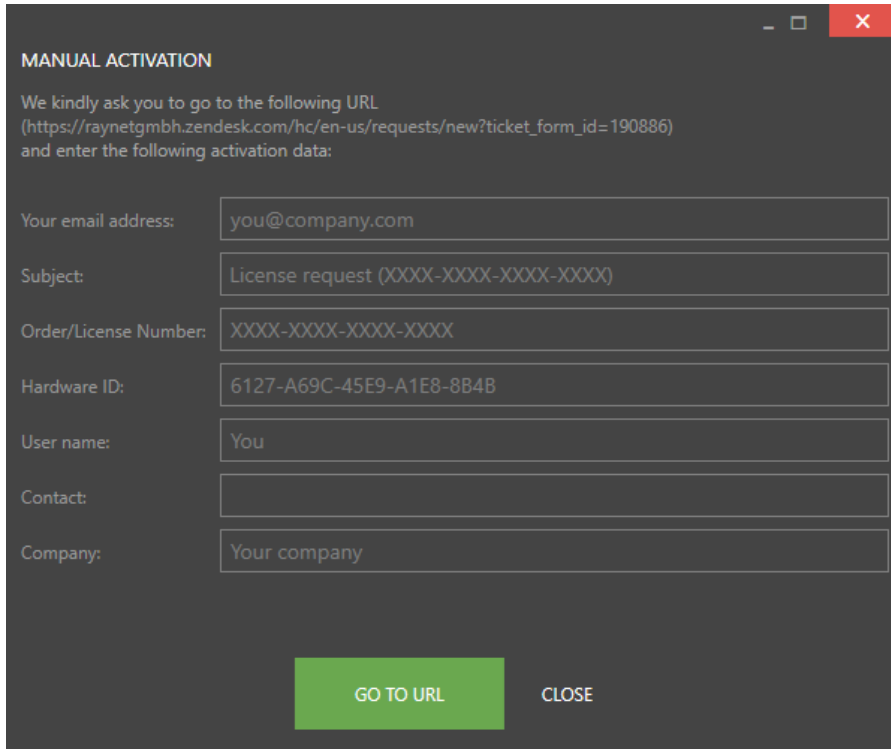
Transfer the license
If this option is selected, the order number and details may be used to activate RayPack on a second machine, that has differing hardware (which obviously has a different Hardware ID). This assumes that RayPack has been uninstalled from the machine on which it was previously activated on. The transfer license functionality is logged on our license servers and is periodically checked to ensure that no abuse is made of this functionality.

If the license transfer is part of a regular maintenance and can therefore be prepared and scheduled, it is highly recommended to use the deactivation function first, to disconnect license and packaging machine. This is the standard way for transferring licenses. The option offered here is intended for unscheduled transfers, required if a machine, for whatever reason, cannot be accessed or used operational any longer.

Manual Activation

On choosing the manual activation, the dialog shown below is displayed.

This basically shows the contents of the ticket form that will be opened at Raynet. If there is an internet connection available on the machine, click on the **GO TO URL** button to open the URL shown in the top of the window in the default browser of the system. After a File Order has been opened in the Raynet Support Panel, a license file will be delivered. Information of how to use this file are available here.

If no internet connection is present on the machine on which the activation process is taking place, copy the contents of the dialog onto a machine which has an internet connection and use the URL on that machine. On receiving the ticket, a license file will be generated and sent back. Information on how to use the license file can be found here.

	Tip: Please ensure that when copying the information from the **MANUAL ACTIVATION** dialog everything is added as shown above.

Once the license file has been generated the following will be shown:
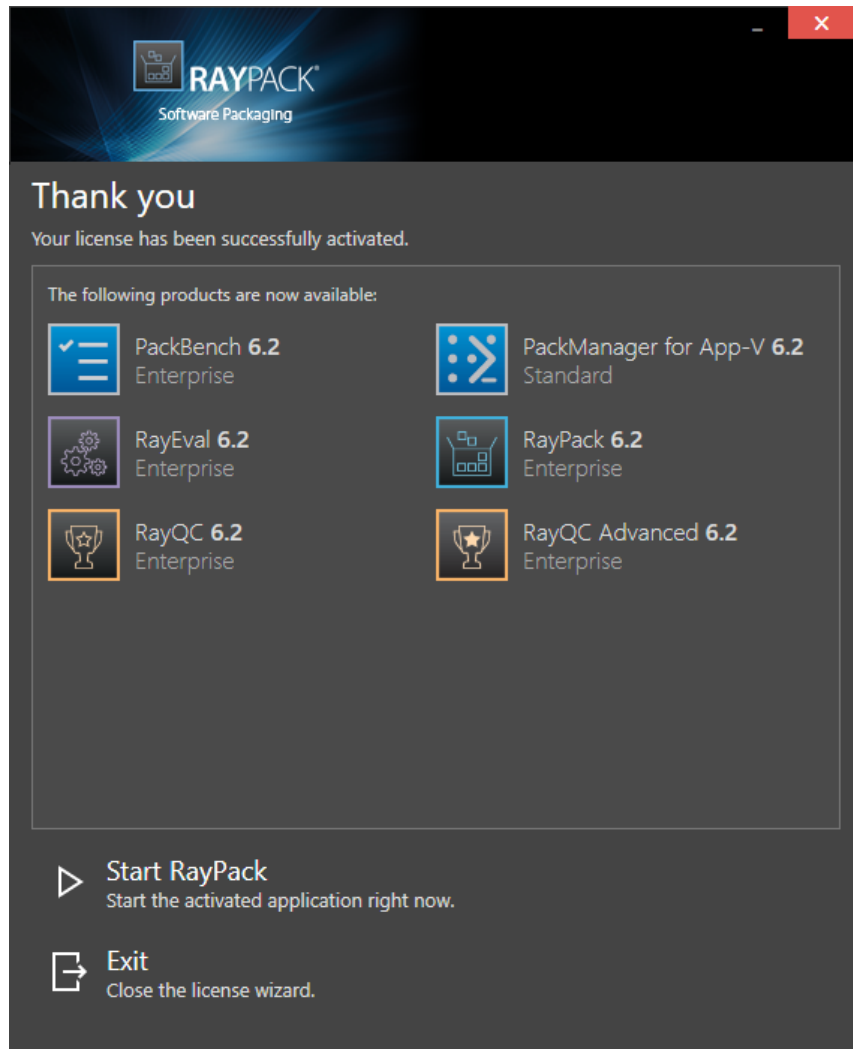
> **Note:**
> Depending on the license, more available products may be shown. As an example, see the image above.

The option of starting RayPack or just closing the activation wizard are available now.

Troubleshooting

If there are any problems during the activation process, please contact our help desk for receiving assistance in activating RayPack.

License File

If a license is already available, or a license file has been received as a result of activating RayPack via e-mail, then all that is required is to copy the license file into the installation directory of RayPack (the directory in which the `RayPack.exe` resides). Clicking on the **I have a license**

button on the **License wizard** dialog opens a dialog box which allows to choose the license file. Once chosen, the file will be copied automatically to the RayPack installation directory. Please ensure that sufficient permissions to allow the creation/copying of a file to the installation directory of RayPack are available.
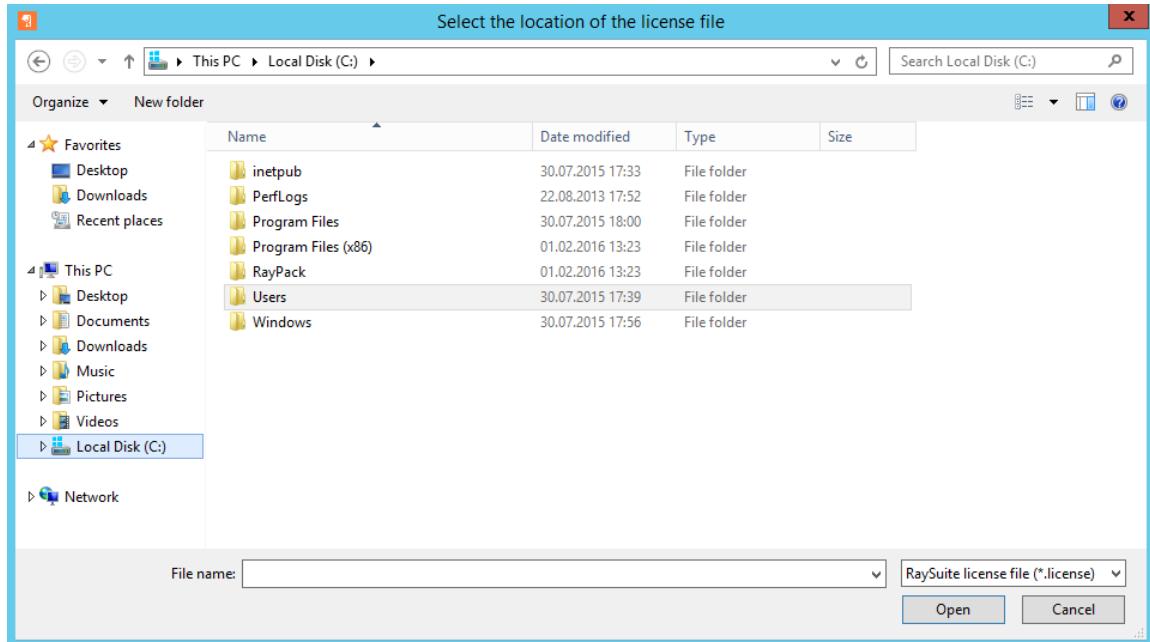
Once the license file has been copied to the correct location the following will be shown:
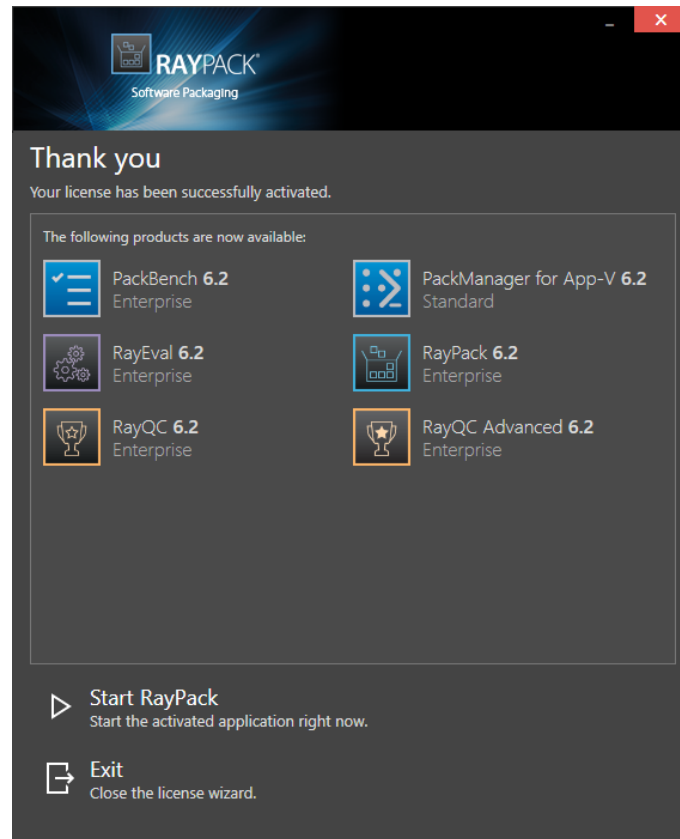
> **Note:**
> Depending on the license, more available products may be shown. As an example, see the image above.

The option of starting RayPack or just closing the activation wizard are available now.

Troubleshooting
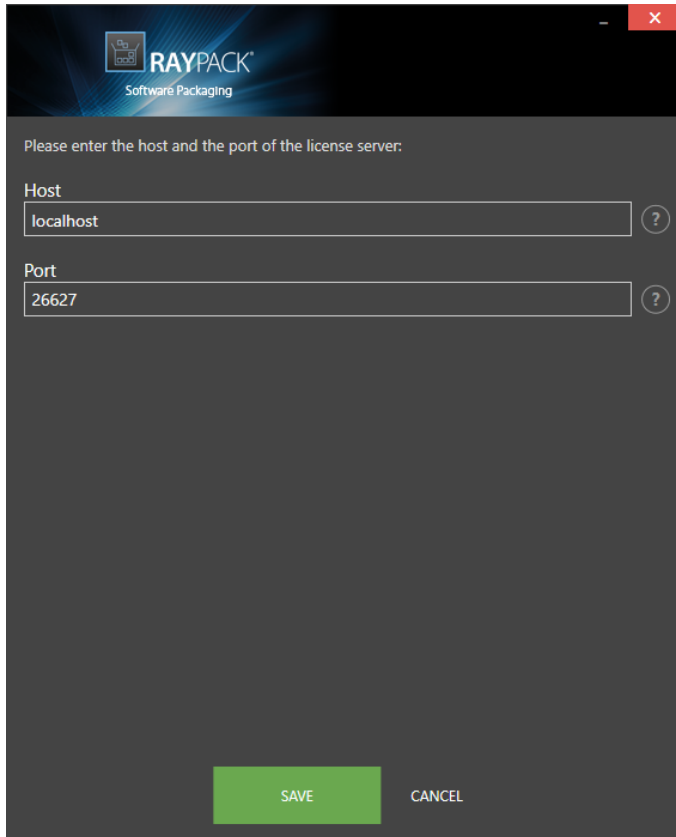
If there are any problems during the activation process, please contact our help desk for receiving assistance in activating RayPack.

Floating License Server

RayPack can be activated using a local floating license server. This requires that the server component is installed (the installation is available separately from the product installer).

Once the server is configured, the following details are required from the server administrator:

- Server name or IP address
- Configured port (by default `26627`)

Enter required values and confirm them by clicking on the **SAVE** button. The server will be contacted once to verify the correctness of the data. If the server is not available at that time, an option will be presented to write the data anyway.

Once the connection details are saved, please restart the product to activate it using the floating license server.

I Don't Have a License or Order Number

If neither a license or order number is available, then just simply register with Raynet to download an evaluation license for RayPack. This allows potential customers to test and work with RayPack before purchasing. Choosing **I don't have a license or order number** opens the Raynet website in the default browser, allowing potential customers to download an evaluation copy of RayPack.

I Want to Take My Activation Back

Deactivating an existing license for RayPack may be required if the packaging machine used has to be switched. Whenever there is a scheduled migration, e. g. when a virtual machine is transferred in a way that affects the **Hardware ID**, or when a physical machine is no longer used for packaging purposes, deactivating the license is the right thing to do.

To Deactivate a Licensed RayPack Installation

1. Launch RayPack and open the license and edition tab of the **about** area.

2. Click on the **Open the license wizard** button on the lower left hand side of the application window.

3. Use the option **I want to take my activation back...**

4. Enter the **order number** that was originally used to activate RayPack on the current machine. It was part of the resources and information material delivered during product purchase.

5. If required, adjust the user name already entered into the input field **User name**. The users who activate and deactivate an installation do not necessarily have to be the same.

6. Click on **DEACTIVATE NOW**.

   The license wizard will connect to the Raynet licensing server and send the deactivation information. On success, the number of licenses available for activation, which are bound to the used order number, is incremented by one. With this new free license it is possible to activate any RayPack installation, on the current machine or any other.

Troubleshooting

If any problems during this process occur, please contact our help desk for receiving assistance in deactivating RayPack.

Home Screen

When RayPack launches a highly functional home screen is displayed, which is a tile based dashboard that offers quick access to key functionality and a user-specific history of recently active packaging projects.
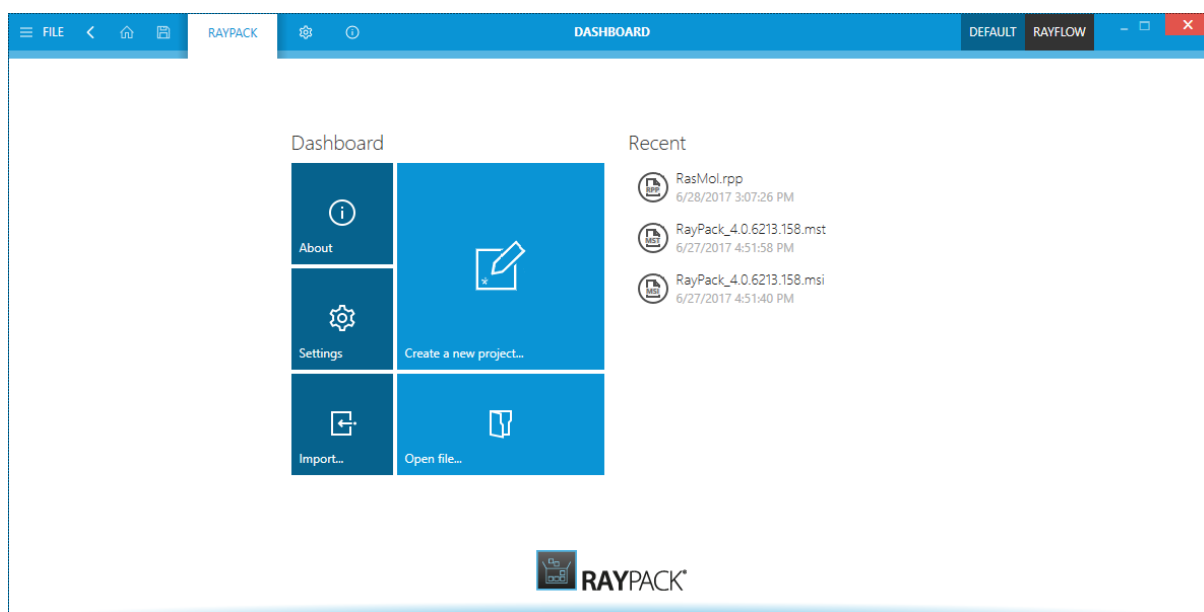
Click on the items to see more information about the individual topics.

The Main Toolbar

Throughout RayPack the Main Toolbar is visible and depending on the contents of the view shown, adds or removes menu items dynamically. As a rule of thumb the items shown below are always present on the Main Toolbar.
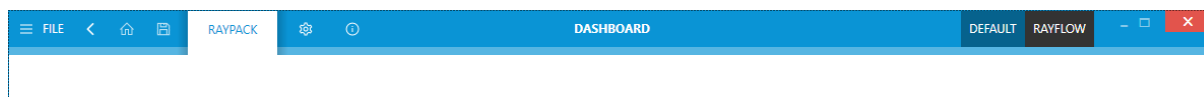
Click on the items to see more information about the individual topics.

FILE
This opens the **FILE** menu. The **FILE** menu is dynamically created, depending on what Tool is currently active. Please refer to the *Common Dialogs* section to read more about *the FILE menu*.

View history
With a left-click on the arrow button, users navigate one step back within the history of recently opened views. Right-clicking the arrow displays the recently visited views, and allows to return

to a specific view from that list.

This view history is limited to those views without project relation, or with relation to the currently opened project. Thus, returning back to a view is not possible if it is called for a project that is no longer opened.

Home

Choosing this button will result in returning to the Home Screen. If any projects and or files are opened and there is a requirement to save any changes, there will be a prompt to save before returning to the Home Screen.

Save

Saves the current file / project. This button is only active if any changes have been made and required saving.

Settings

Opens the settings for RayPack.

About

Contains the information provided by the views Get Started, License and Edition, and Troubleshooting.

Window title

The window title displays the current scope of activity. If an editor is active (e. g. PackDesigner or the PackRecorder editor interface), the file name of the currently opened project is part of the window title as well.

Standard window controls

The standard window controls allow to minimize, maximize, resize and close the application window. The availability of each control follows the Windows schema for standard controls as known from any desktop application.

Profile Switcher

Profile switcher is a quick selector of profiles, defining settings used during repackaging and building. The current profile name is displayed in the right top corner (for example *DEFAULT* in the previous picture). Press the profile name to open a drop-down menu showing a list of available profiles. To change a profile, select its name and press **Switch to selected profile**. The button **Go to profile settings** open the respective screen where new profiles can be created.
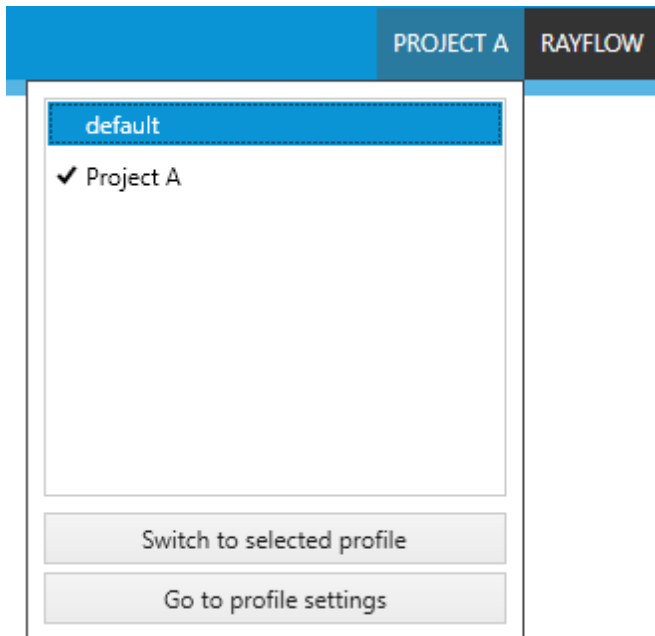
Create a New Project

Upon selecting **create a new project** from the Home Screen, the method of how the new project will be created can be chosen.
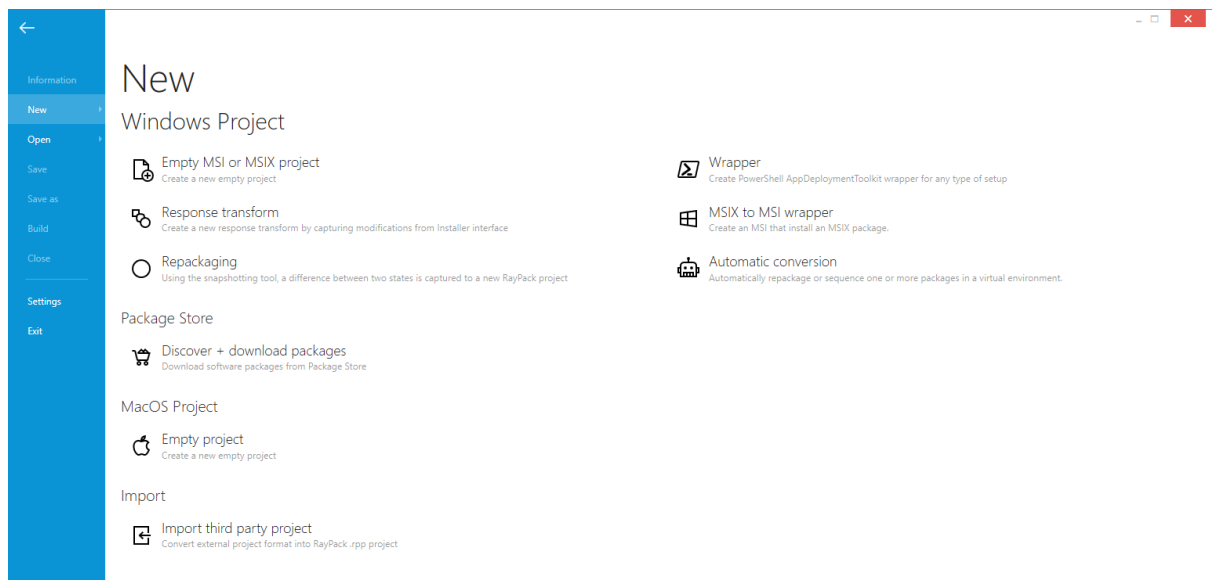
Create

Empty MSI or MSIX project

Choosing this option will create a new empty MSI based project and open it in the PackDesigner, which allows the customization of the package with custom files, registry keys, and resources.

Response transform
Choosing this option will start the PackTailor tool, which allows to create a Windows Installer Transform file based on an existing Windows Installer MSI file.

Repackaging
Choosing this option opens the PackRecorder. The PackRecorder wizard guides the user through the capture, configuration, and creation of a packaging project. Depending on which Capture mode has been selected, the actual screens and wizard process shown may differ slightly.

Wrapper
Choosing this option will start the PackWrapper tool, which allows for the creation of a PowerShell-based wrapper to bootstrap installation and uninstallation of MSI and EXE setups.

MSIX to MSI wrapper
Choosing this option will start the universal apps import dialog. The visibility of this option depends on the RayPack license used.

Automatic conversion
Choosing this option will start the PackBot tool, which allows for the automatic repackaging and sequencing on virtual machines.

Import

Import third-party project
Choosing this option will start the 3rd-party project import dialog. The visibility of this option depends on the RayPack license used.

Choosing the leftwards pointing arrow at the upper left corner of the dialog will close this dialog without making any changes.

Empty Project Options
When choosing to create a new project additional settings will be shown. These settings offer the ability to control and fine-tune new project settings.
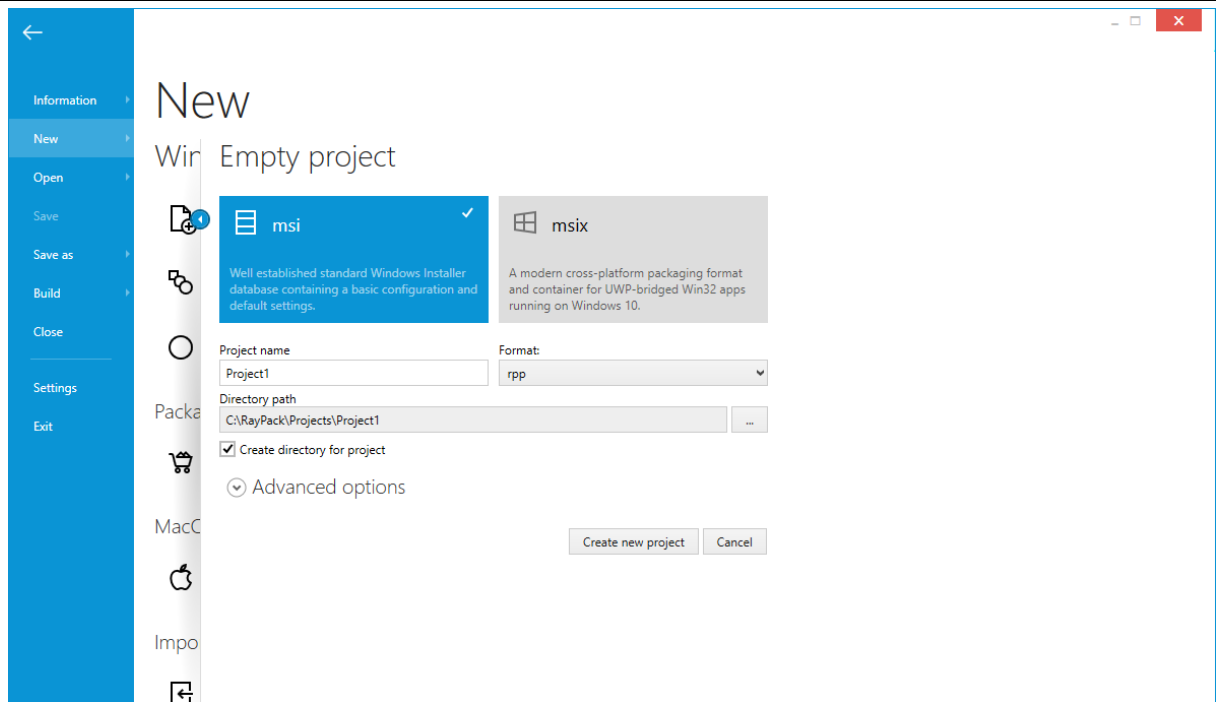
Project Type
Select the type of the project to be created:

- o **MSI**
  Creates a new blank MSI file. You can choose between an XML-based Raynet Package Project (recommended) or create a blank MSI from scratch.
- o **MSIX**
  Creates a new empty MSIX-like project, which can be built to MSIX format hosting win32 apps with help of UWP Desktop Bridge or cross-platform apps.

Directory Path
The root folder where the project is to be saved. Depending on whether the checkbox **Create directory for project** is selected, the file is created either directly in the selected location or in its subdirectory.

Project Name
This is the file name of the new project.

Create Directory for Project
With this option selected, a subdirectory is created inside the selected directory. The name of the subfolder is taken from the Project Name textbox. If the option is not selected, then the file is created directly in the selected folder.

Language (MSI Projects Only)
Defines the language of the setup user interface. When **Default** is selected, the language present in the *blank MSI template* will be used. The language selection affects the following MSI / RPP features:

- o Translated messages and UI texts.
- o Adjusted `ProductLanguage` property.
- o Adjusted code page to support international characters.

Platform (MSI Projects Only)
Defines the target platform of the setup. When **Default** is selected, the platform present in the *blank MSI template* will be used. The Windows Installer engine does not support multiple target platforms.

	Note: If 64-bit platform is selected, the package will not run on 32-bit systems. In order to create a package that runs on both 32-bit and 64-bit platforms use 32-bit platform using the lowest common denominator principle.

Schema (MSI Projects Only)
Defines the Windows Installer database schema. When **Default** is selected, the database schema present in the *blank MSI template* will be used. The schema may affect availability of certain MSI tables and columns, and on which Operating Systems the package runs. The list of available MSI engines which have been released so far is available here: https://msdn.microsoft.com/en-us/library/windows/desktop/aa371185(v=vs.85).aspx.

Create a New Project

When creating a new project is chosen, the **PackDesigner** view will be opened and the full functionality of the PackDesigner will be available to create and edit a project from scratch. To view the help information for the PackDesigner, please read the *corresponding help section.*

Managing Default Templates (MSI Projects)

The default set of platforms and templates can be extended and customized. The advanced topic *Managing Default Platforms and Languages* contains detailed information about customization options.

Open Capture Project

This section describes the process of opening an existing project file. On clicking the **open capture project** button from the Home Screen, a typical "File-open" dialog that has been specifically configured to filter files based on the capture project format will be presented. Selecting the file and then choosing the **open** button will open the file / project in RayPack. This can also be achieved by simply double-clicking on the file, which will also open the file / project in RayPack.

An alternative approach to opening a capture project is to click on the **FILE** button from the application menu bar. Select **Open** from the displayed option set at the left-hand side and pick **PackRecorder Project (*.rcp)** from the list of available project types. A standard windows "File-open" dialog is displayed, ready for location and file browsing.
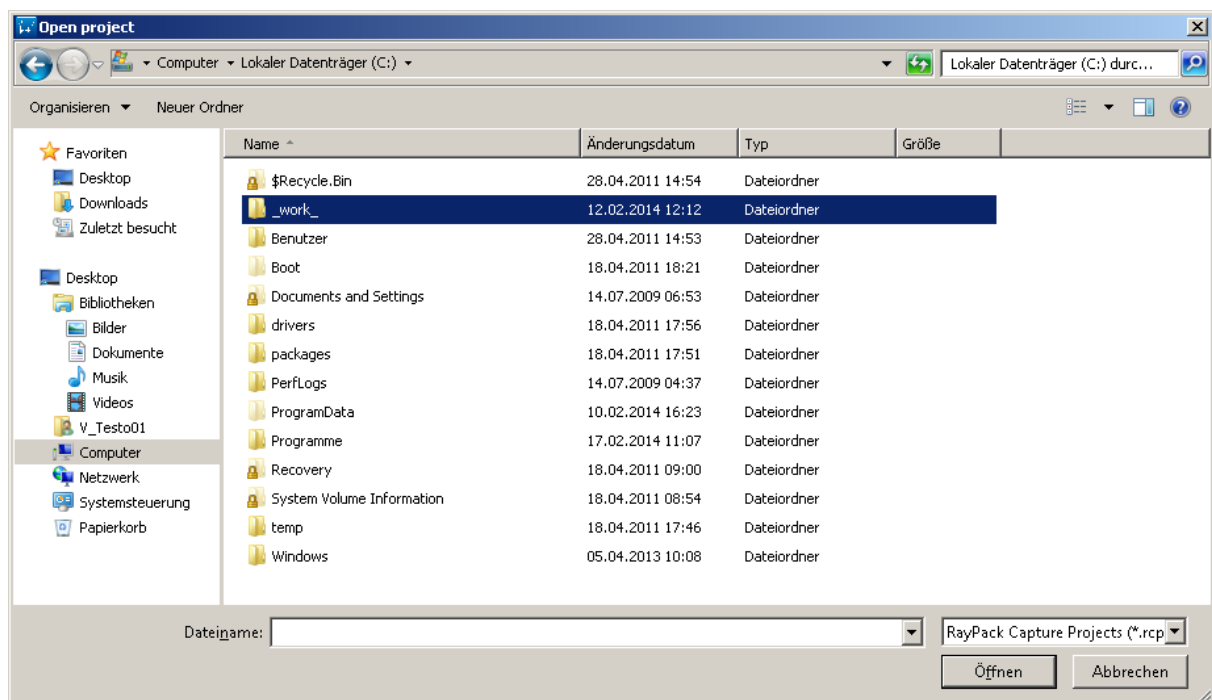
On opening the file / project, the CaptureEditor view will be shown.

Open Project

This section describes the process of opening an existing file and / or project that RayPack supports. When clicking upon the **open project** button from the Home Screen, a typical "File-open" dialog, which has been specifically configured to filter files based on the supported file types of RayPack will be presented. Initially all file types will be shown, but it is possible to change the filter by selecting the drop-down combo box to the lower left of the dialog and choosing the file / project types to be shown. Selecting the file and then choosing the **open** button will open the file / project in RayPack. This can also be achieved by simply double-clicking on the file, which will also open the file / project in RayPack.

An alternative approach to open a project is to click on the **FILE** button from the application menu bar. Select **Open** from the displayed option set on the left-hand side and pick either **Windows Installer project** (`*.msi`, `*.mst`, `*.rpp`) or **PackRecorder Project** (`*.rcp`) from the list of available project types. A standard windows "File-open" dialog is displayed, ready for location and file browsing.
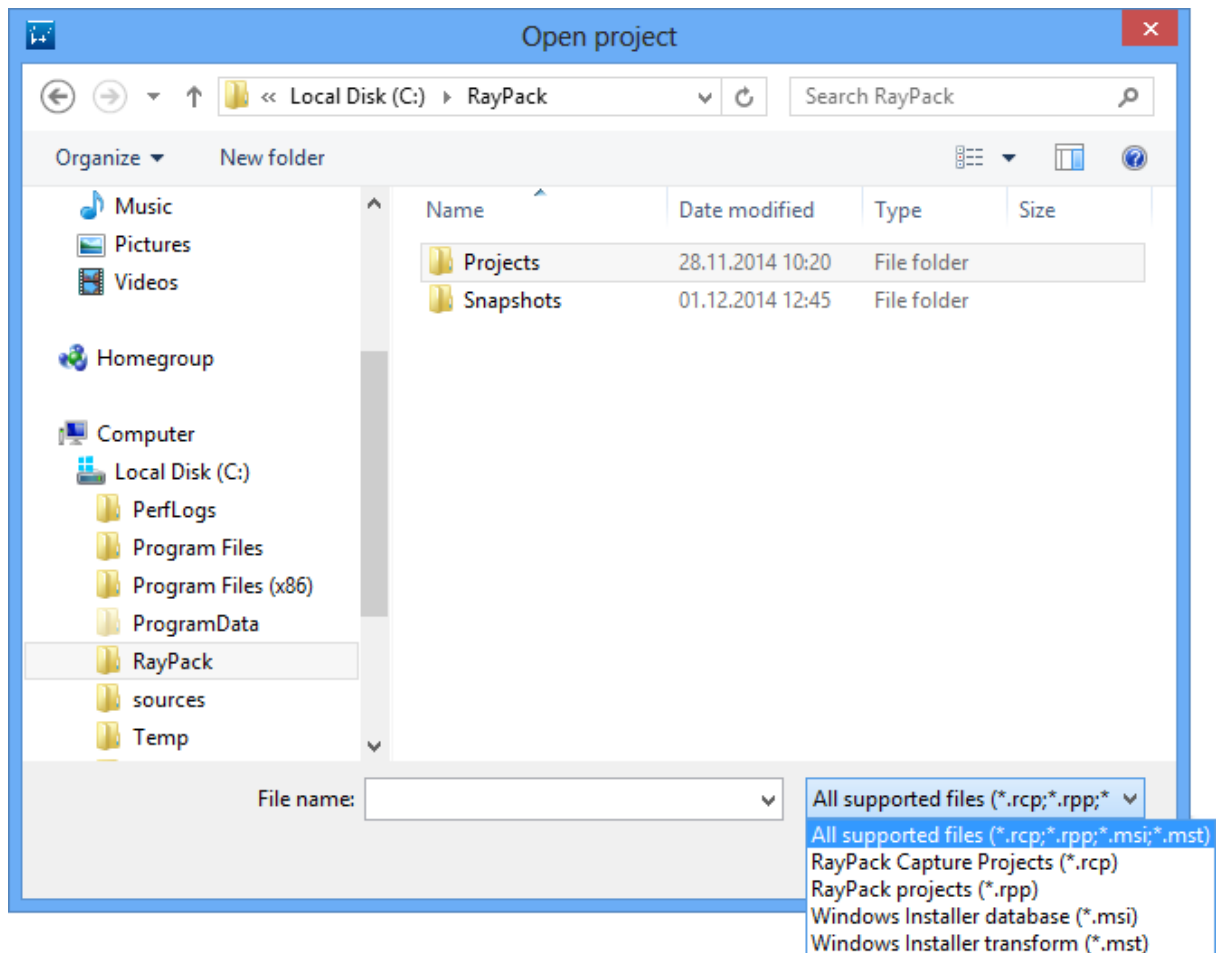
Currently, the following file / project formats are supported:

- RayPack capture projects (`*.rcp`)
- RayPack packaging projects (`*.rpp`)
- Windows Installer database (`*.msi`)
- Windows Installer Transforms (`*.mst`)

When opening MSI files, RayPack scans the folder for presence of a single matching MST file and shows a prompt if a candidate is found:
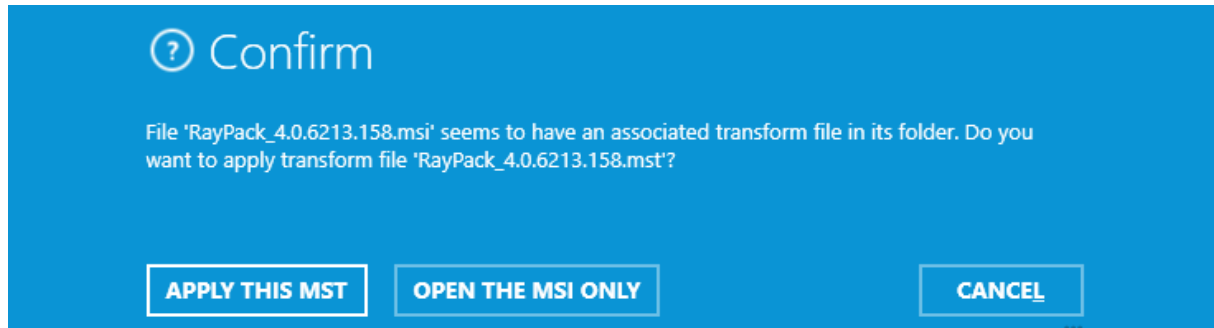
Press **APPLY THIS MST** to apply the MST to the base MSI. Press **OPEN THE MSI ONLY** to ignore this suggestion.

This behavior is configurable via *profile settings*.

Recent

This section describes the functionality provided by the recent view and its contents. The list of recently used projects / packages is available from the **Home Screen**, as well as the **Open** dialog of the **FILE** menu.
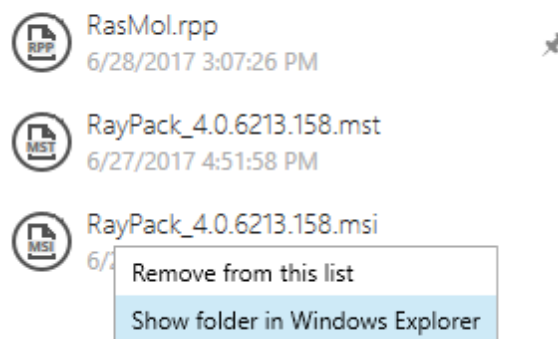
Context Menu

Right-clicking on an item in the recent list opens a context menu that allows the operations displayed below to be carried out.

Remove From This List

Removes the selected item from the recent list. Please note that this does not delete any files, but simply removes the item from being displayed in the recent list. If the file/project is reopened in RayPack it will naturally be again visible in the recent list.

Show Folder in Windows Explorer

Opens the folder containing the selected file/project.

Pinning / Unpinning Projects

Frequently used or favorite projects can be pinned/unpinned from the list. To pin a project, click a small pin icon next to its name. The project will be always visible in the Recent view. To unpin project, press the small pin icon again.

About

Choosing the **about** tile from the Home Screen displays the about area. It contains the get started view, providing information about the individual tools of RayPack, Samples and links to various resources. Additional supportive views regarding license and edition as well as troubleshooting are available by clicking on the other view tabs of the about area.

Get Started

Choosing the **get started** button from the Home Screen reveals this view in the about section. It contains information about the individual tools of RayPack, Samples, and links to various resources. Additional supportive views regarding license and edition, as well as troubleshooting are available by clicking on the view tabs.
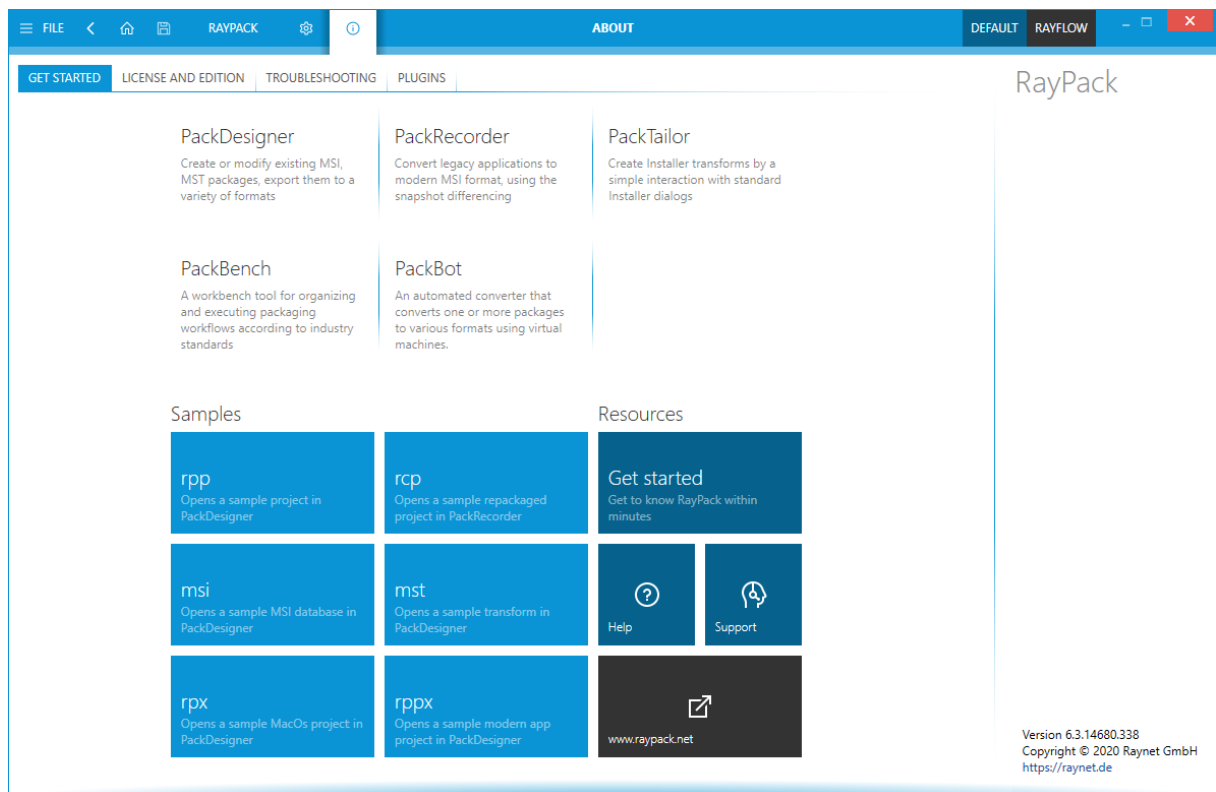
Click on the buttons to get more information on the individual topics.

Meet RayPack

These are the three main tools that RayPack comprises of:
- PackDesigner
- PackRecorder
- PackTailor
- PackBench

Samples

RayPack includes samples, which show how the different components can be used to create projects, edit projects and files, and build packages.

Resources

RayPack includes various resources that can be used to make the experience with RayPack more productive and provide help where needed. Please note that some resources (including some items in this help file) are only available on-line or with an internet connection.

Get Started

This opens the get started help file, which includes tips & tricks how to configure and use RayPack.

Help

Opens this document.

Support

Opens the web site for Raynet product support.

www.raypack.net

RayPack on the web.

License and Edition

The about area contains the License and Edition view, providing all usage relevant license information about the current product instance. Additional supportive views regarding get started guidance, as well as troubleshooting are available by clicking on the view tabs.
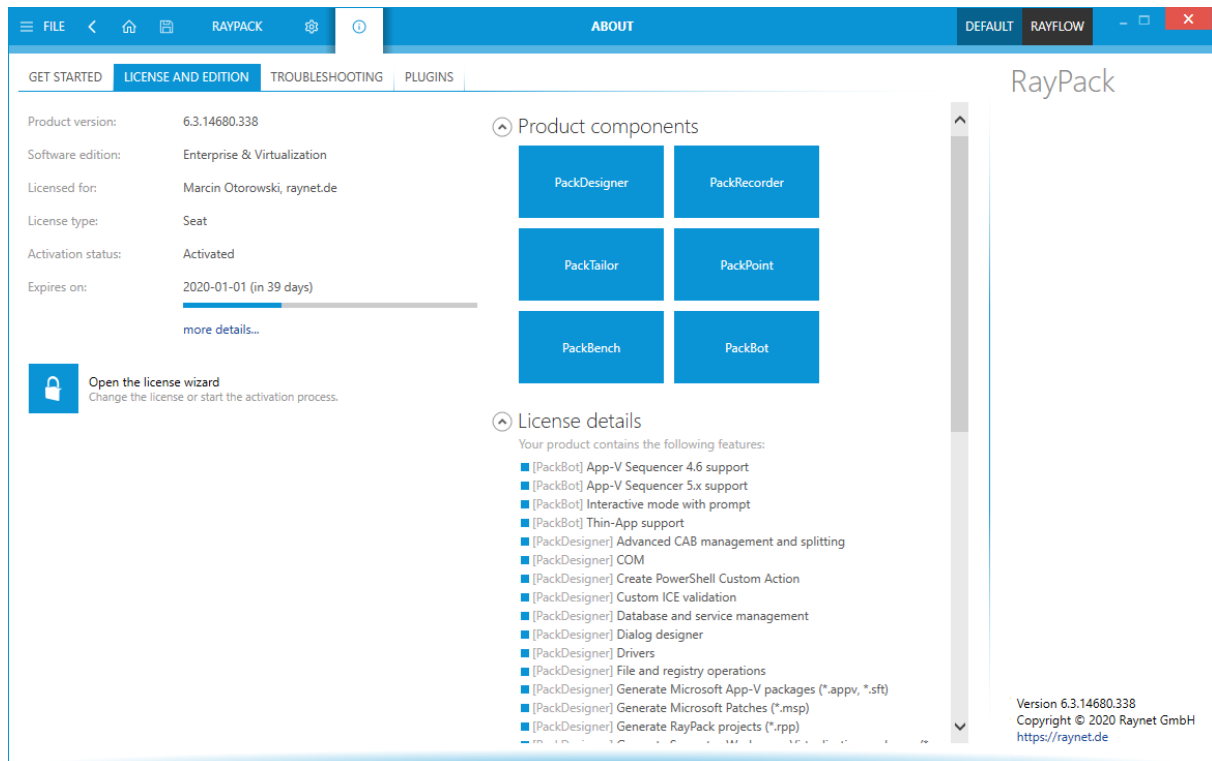
Click on the items to get more information on the individual topics.

Here, a variety of information is shown regarding the current version of RayPack, license information and the installed features that are currently licensed. License information can be entered using the license wizard by choosing Open the license wizard.

License Details

Depending on the license that has been purchased, the various features that are available for that particular license are displayed when expanding the **"License details"** section.

Add-ons

Depending on the license that has been purchased, the various add-on modules that are available for that particular license are displayed when expanding the **"Add-ons"** section.

License Files Management

After the activation of RayPack, an encrypted license file (`.rsl`) is generated and saved in the root folder where the program has been installed.

Subsequent runs of the application will require no internet connection/activation as long as a valid license file is present in any of the following folders:

1. The main installation folder (for example `C:\Program Files (x86)\RayPack`).
   o When running a portable copy, the path from where the main executable has been started applies.
2. `\Raynet\Licenses` folder in Common 32-bit Program Files directory (for example `C:\Program Files (x86)\Common Files\Raynet\Licenses`).
3. `\Raynet\Licenses` folder in Application Data directory (for example `C:\Users\Administrator\AppData\Roaming\Raynet\Licenses`).
4. `\Raynet\Licenses` folder in Program Data directory (for example `C:\ProgramData\Raynet\Licenses`).
5. Finally, if the current instance of RayPack is running from share or a removable disk, then the remote license from `C:\Program Files (x86)\RayPack\RemotePackRecorder` is used.

The required extension of a license file name is `.rsl`.

The license file is portable and will not be removed when RayPack is uninstalled. A backup of the license or a copy on the stick in the portable installation location allows to start RayPack, given that the machine is the same as used to activate the product.

Resolving Multiple License Files

In case more than one license is found, the following logic is applied

1. The order in which folders are probed is equal to the standard probing described in the above section.
2. Should two or more license files be found in the same level, the one that is active and valid for the current machine wins. If more than one license is active and valid, the first one wins (note: the order in which license files from a folder are probed may be not deterministic).
3. Should no license in the same level be valid and active, the next folder is processed.
4. If no valid and active license has been found in any of the probed folders, the program will not start, and the activation wizard will show up.

Logging and Troubleshooting

Detailed information about the order and which licenses are applied on start-up can be found in the RayPack logs. Refer to the Troubleshooting section for more information.

Troubleshooting

The about area contains the **troubleshooting** view, providing handy information about the crucial system paths defined for the current product instance. Additional supportive views regarding get started guidance, as well as license and edition details are available by clicking on the view tabs.

Any information displayed within the **troubleshooting** tab is read-only. It is provided to ensure transparency about the current instance settings, which is vital for proper troubleshooting measures in case of support relevant issues.
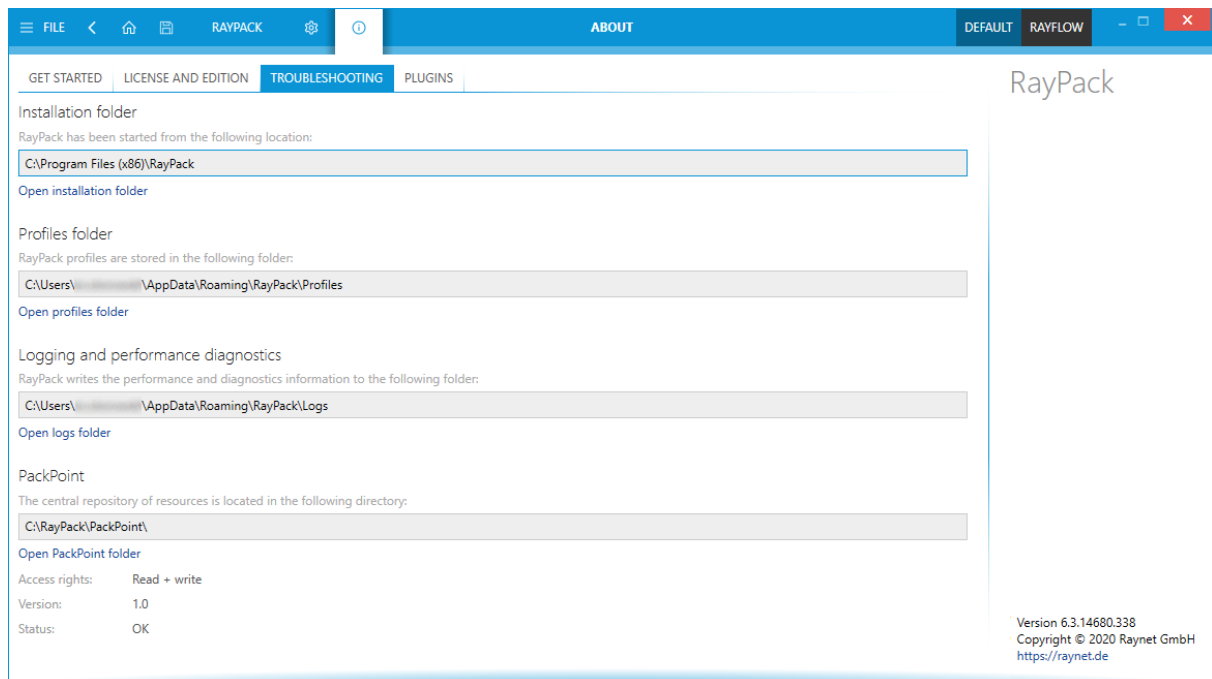
Click on the items to get more information on the individual topics.

Installation Folder

This path shows where the currently running instance of RayPack is located. Click on the **Open installation folder** link to open a windows explorer instance at the displayed location for further instance resource review.

Profiles Folder

This path is used as default for the creation of new setting profiles via the user interface of the current RayPack instance. Please refer to the Profiles topic to read further details on how to change this default path. Click on the **Open profiles folder** link to open a windows explorer instance at the displayed location for further instance resource review.

Logging and Performance Diagnostics

RayPack is by default configured to write log files with information about each product work session. A new log file will be generated for every launch of the current application instance. These log files contain vital information for any troubleshooting or help desk measure, such as environment information about the physical machine RayPack resides on, the steps performed during a session, and exception details thrown by the application in case of operational issues. Please make sure that the target directory displayed here provides sufficient disk space and is accessible for the currently logged in user. Click on the **Open logs folder** link to open a windows explorer instance at the displayed location for further instance resource review.

By default, RayPack logs minimal amount of information. The chapter *Advanced logging options* shows how to enable more verbose output, which allows quicker troubleshooting of the application.

PackPoint

If a multi-user repository has been defined to be used for the currently active product instance, the path to the PackPoint location can be reviewed. Additional information may be retrieved as well (see below). Click on the **Open PackPoint folder** link to open a windows explorer instance at the displayed location for further resource review.

Access rights
The level of access RayPack has towards the PackPoint location. The access rights are derived directly from the rights of the user profile that runs the currently active RayPack instance.

Version
The PackPoint version reflects resource and template states. It is used to manage updates and synchronizations. If the PackPoint version differs from the local settings version, a synchronization will be triggered at the next application launch.

Status
Depending on the current system configuration, this status may vary between the following options:

- **New** - When RayPack has been started on a machine for the first time, and there is no RayPack configuration and no active PackPoint.
- **Ok** - PackPoint is up and running in expected standard mode.
- **Reset** - If a local RayPack configuration for a PackPoint exists, but the PackPoint itself is not set up (which may be the case after a RayPack upgrade or if the PackPoint resources have been deleted manually)
  In this state, RayPack shows a warning message at launch, informing the user about the imperfect operational state of PackPoint.
- **Not Available** - Status when PackPoint is configured to use resources from a currently unavailable location (disconnected network share/mapped drive).
  In this state, RayPack shows a warning message at launch, informing the user about the imperfect operational state of PackPoint.

- **Configuration corrupted** - Status whenever the local RayPack configuration file for PackPoint is missing or corrupted.
  Users will be prompted to repair the PackPoint and restart RayPack. These warnings may be ignored, since RayPack is still operational by continuing to maintain the current session based upon local resources.
- **Standalone** - If a minimal installation is performed, there is no active PackPoint connection. However, even if not used, PackPoint may very well be configured anyway.
- **Required Resources Missing** - Present if crucial resources are missing from PackPoint (due to damage, missing access rights, or unavailable connectivity).
  Users will be prompted to repair the PackPoint and restart RayPack. These warnings may be ignored, since RayPack is still operational by continuing to maintain the current session based upon local resources.

> **Note:**
> All paths are absolute. If you run RayPack from an MSIX app, the paths are additionally virtualized. This means that the actual physical path where logs or binaries are stored may be different from the displayed one.

Plugins

This section contains an overview of currently installed PackDesigner plugins.

Settings

Choosing the settings button from the Home Screen allows users to make changes to the configuration.

An alternative access to the settings section is provided by clicking the settings tab displayed within the main toolbar, which is present within all RayPack editor views.

The setting screen is divided into two sections

- Section categories
  - General settings (instance-specific)
    - Interface
    - Profiles
    - Resources
  - Profile settings (profile-specific)
    - Projects
    - Repackaging
    - Designing
    - Signing + tagging
    - Conversion
    - RayFlow + PackageStore
    - Deployment
    - Virtual Machines

> **Note:**
> In order to apply these settings, press the **OK** button on the bottom of the screen, or navigate away from the screen. RayPack will prompt you to save the changes if you haven't saved them yet.

Interface

This section of the **SETTINGS** contains the options to define the language that is used throughout RayPack and the option to switch the animations on and off.

Language

The language option enables users to choose the language that should be used for the RayPack application. the language is chosen by selecting one of the options that can be found in the drop-down menu which is available under the **User language interface** option.
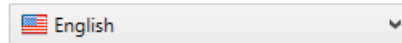
As shown in the screenshot there are currently three different languages available for the user interface of RayPack: The available languages are English, German, and Polish. The active language can be chosen by selecting it in the drop-down menu and then saving the changes by clicking on the **Save changes** button which is located in the swipe-bar.
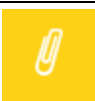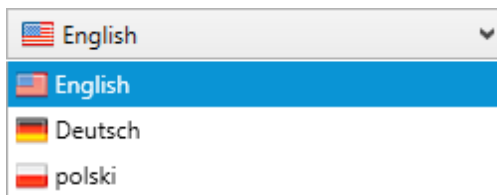
> **Note:**
> In order to apply the language settings, it is necessary to restart RayPack.

Animations

When RayPack is going to be used via remote connections, it may save precious bandwidth and performance if the user interface animations are deactivated. They are active by default, but may be deactivated by setting the slide toggle **Enable user interface animations** to **No**.

Profiles

RayPack uses profiles to store the various options and settings required for operation. By utilizing profiles, settings and options can be stored for individual situations. For example, if packages need to be created for two different departments of your organization, or for two different customers. As each department or customer may have differing requirements, simply storing the settings and options for each department / customer in a separate profile allows for easy switching between profiles without having to configure RayPack and any templates every time a

package needs to be created.

The *Profiles* section provides an easy overview and management tools over profiles of RayPack. These include:

- The location of the profiles folder
- The Profile to use during the capture process and the settings and options that are used when a new package project is created
- Creating a new profile, or editing existing profiles

The central section shows the currently selected profile and the full path where the defined profile settings can be found.

Creating a New Profile

In order to create a new profile, click on the button *Select or create another profile...*
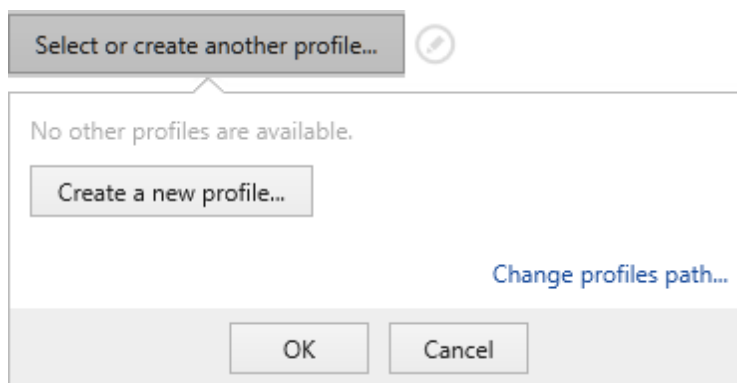
Note:
Depending on your environment, this pop-up may list other co-existing profiles.

---

1. Click on button **Create a new profile...**
2. A modal dialog will be shown, prompting for the name of the new profile. By default, the input

will be set to the name of the current profile.

<table>
<tr><td>📎</td><td>**Note:**
The name of the profile will be used as a name of its physical data file. Thus, RayPack requires a name that may be used as a valid file name. Additionally, the name has to be unique.</td></tr>
</table>

3. Press OK to create a new profile. RayPack will automatically switch to the new profile.

Changing the Current Profile

RayPack supports multiple profiles which can be switched as required. To access the profile selector, click on *Select or create another profile...* button, and then pick up the profile from the list below:
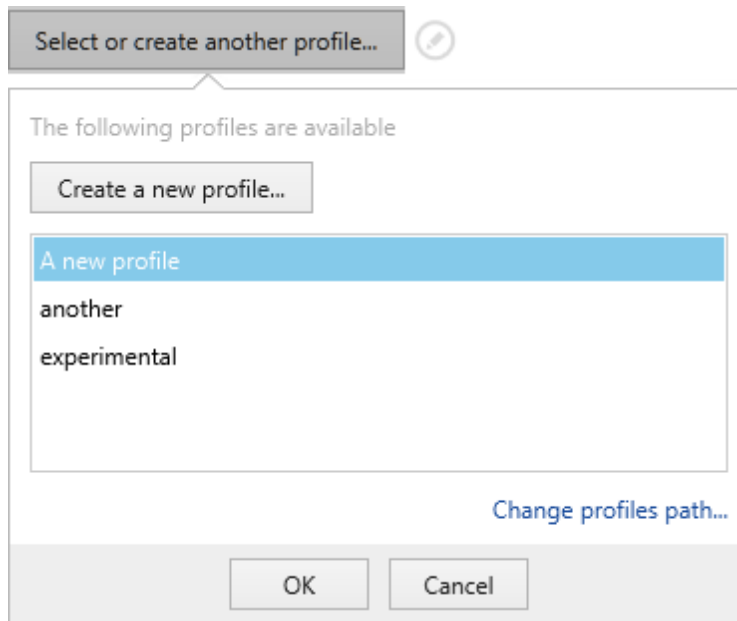
The list contains names of other profiles present on the current system. For a better clarity, the currently selected profile is not being displayed.

In order to apply another profile settings, select the name of the profile to apply, and click on *OK* button to change the current profile.

Changing the Profiles Path

By default, all profiles are stored in the `%appdata%\RayPack\Profiles` subfolder. Advanced users may easily change the folder to point to a shared location or elsewhere.

To change the profiles path:

1. Click on *Select or create another profile...*
2. Click on *Change profiles path...*
3. Pick a folder that contains at least one `*.rpprofile` file

4. RayPack will automatically switch to the first profile available in the selected location.
5. If the folder is empty or contains no `*.rpprofile` file the old profile will be kept. New profiles will now be created inside of the newly selected folder.

Resources

Please refer to the PackPoint section for a general overview regarding the PackPoint concept and mechanisms. The settings available for global product instance definition determine, how RayPack reacts on conflicts between local and PackPoint resource versions. Please select the more suitable mode for the given packaging environment:

In case of conflicts:

- PackPoint overrides local settings (recommended default setting)
  In this case, users will lose local adjustments in case of version conflicts. This means that PackPoint is the leading entity for resource management. If there is a PackPoint instance, users are forced to follow the updates dictated by the PackPoint administrators.

- Local adjustments are preserved
  In this case, differences between PackPoint and local settings are merged. This is a softer mode of PackPoint support, allowing users to keep influence on their local profiles. However, this setting makes it harder to ensure a uniform basic resource stock for all instances that are connected to a shared PackPoint. It is recommended to double-check the outcome of this behavior for the RayPack machines that are part of a packaging factory.

Projects

This section contains various global settings defining the template configuration and the default folder where projects are saved.

Project Folder

This is the location to which all projects are saved. For each project, a subdirectory will be created in this location based on the name of the project. For example, if a project named "HelloPackage" is created, then a directory named `C:\Users\packager\Documents\RayPack\HelloPackage` will be created. This directory will contain all of the relevant resources that are required for that particular project. It is recommended that this project directory will be located on an external resource, such as on a shared drive. The projects directory location can be changed by using the **BROWSE** button and selecting an alternate location.

> **Be aware:**
> Read and write permissions for this directory are required!

This location is used as a default value by:

- **PackDesigner**, as a default location when saving an unsaved RPP project,
- **PackTailor**, as a default location for output MST transforms,
- **PackRecorder**, as a default location for output RCP projects.

Template for New RPP Projects

The template file defined here is used to determine basic settings and properties for new packaging projects and for build processes targeting MSI based package formats. Please refer to the *Authoring Large Packages* section for details regarding the specific usage of the default templates, which are delivered along with the general RayPack application resources.

To use another template, press the three dots button on the right, and select the new MSI file to be used as new default template.

The section *Adjusting the Default Template* contains additional information on how to customize

the default templates.

> **Be aware:**
> The checkbox wihch is located above the template field is disabled, because the RPP template is always required. Failing to provide a valid template results in being unable to create a new project.

Template for Windows Installer Transforms

This field defines the default template used when a new MST project is created. The setting is used by:

- **PackDesigner**, as a default set of optional changes available from **FILE** > **TRANSFORM** > **Apply changes from transform template...**
- **PackTailor**, as the set of changes added to the captured response transform,
- **PackRecorder**, as the set of changes added to the captured transform.

This setting is optional and can be disabled by unchecking the checkbox located above.

> **Note:**
> The transform template is not an `.mst` transform but a proprietary XML definition.

To use another template, press the **Browse** button [**...**] on the right, and select the new MST file that is to be used as the new default MST template.

The section *Adjusting the MST Templates* contains additional information on how to customize the MST templates.

Template for Wrappers

This field defines the source folder containing template file for PackWrapper.

This setting is optional and can be disabled by unchecking the checkbox.

The section *Custom PackWrapper Templates* contains additional information on how to customize the templates.

Repackaging

These settings affect how the PackRecorder is executed. These settings are also used by PackBot.

SCANNING

Scanned local drives

This section sets the local drives that are to be monitored for changes during the capture

process. When initially started, all drives that are set as local drives on the machine will be present and selected (this includes any mapped drives). Select / deselect the desired drives to be monitored during the capture process.

Scanned registry hives

This section sets the registry hives that are to be monitored for changes during the capture process. Select or deselect the desired registry hives to be monitored during the capture process.

Advanced scanning

This section enables the selection of advanced options that can be used during the capture process.

- **Permissions changes (files and registry)**
  When this option is set, changes to permissions are monitored during the capture process, both for the file system, as well as, for the registry.
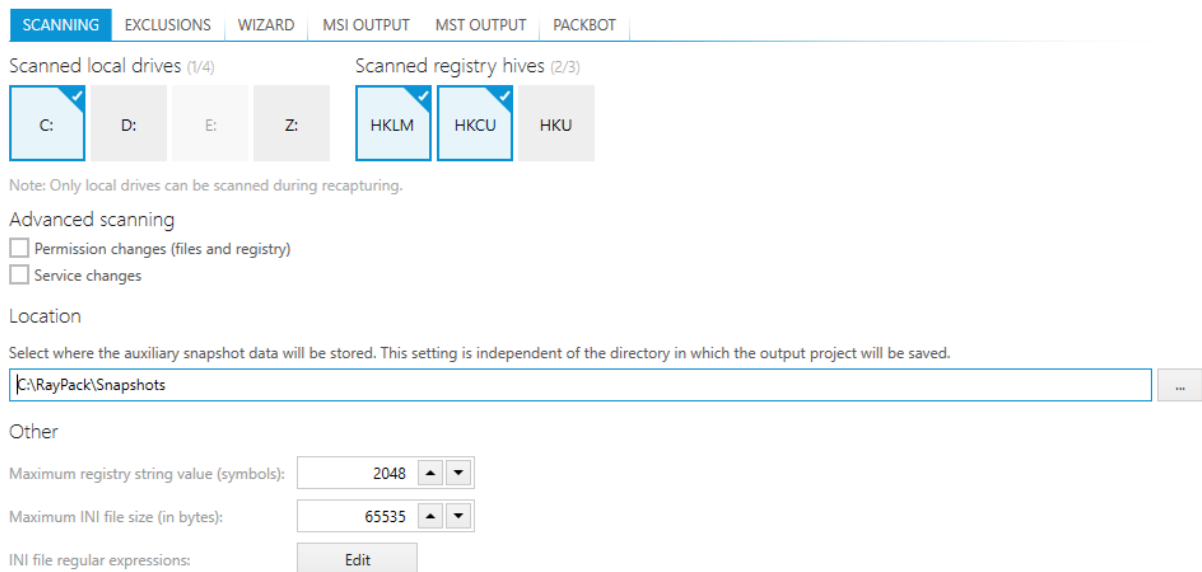
> **Note:**
> Selection of this option will significantly increase the time that it takes to finish for the scanning and the capture process.

- **Service changes**
  When this option is set, changes to any existing services are monitored during the capture process. Normally only the creation of new services is monitored during the capture process. Selecting this option allows the monitoring of changes to existing services during the scanning and capture process.

Click the items within the screenshot to navigate to the various sections of the profile configuration

Location

This is the location where any temporary data and the snapshot data will be stored during the capture process.

Other

This section deals with the general options available for the capture process that are relevant for the snapshot. These include:

- **Maximum registry string value (symbols)**
  The maximal size of a registry string to be kept as a string value, instead of being stored as a binary *blob*. Storing and retrieving large strings in the snapshot data is a time consuming process. Storing and retrieving binary data, however, is relatively fast, with the disadvantage that binary data is usually not *human-readable*, i.e. it is not possible to visually check the data. Setting this value allows the fine tuning of the size of any strings from the registry that are monitored during the capture process and saved as a string. It is recommended not to change the default value.

- **Maximum INI file size (in bytes)**
  INI files are files that usually contain data used for the configuration of an application. In typical scenarios, these contain information that is usually specific to a particular environment or runtime. The MSI technology allows for `.ini` files to be installed in such a way that any values can be configured dynamically during the installation. Whilst this is an advantage over having to work with hardcoded values and eventually having to introduce customizations to change these values, it also has a disadvantage with large `.ini` files in that it dramatically increases the installation time required.

- **INI file regular expressions**
  Choosing this button allows the selection of files that are to be considered as `.ini` files and to prefilter content of any files that are considered to be `.ini` files. Choosing this button will open the INI Editor. The INI Editor is described in its own section.

EXCLUSIONS

Click the items to navigate to the various sections of profile configuration

Capturing Empty Folders

Prior to RayPack 6.3, empty folders were not captured and had to be included manually. From version 6.3, the user can control whether the legacy behavior (ignoring empty folders) or respecting empty folders should be used when building RCP files.

Note that this setting applies to folder that are within your installation directory (so that temporary and system directories are always excluded if they are empty).

Exclusion Lists

The currently available exclusion lists (exclusion lists present in the **Exclusions** configuration directory) are listed in the left hand view. Selecting an existing exclusion list allows the user to **EDIT**, **RENAME**,or **DELETE** it. Selecting **DELETE** will permanently remove the list from the directory!

ADD

This allows an exclusion list to be added to the currently selected directory. When selecting the **ADD** button a valid name for the list must be entered, which will then be added to the **Available exclusions** view.

> **Note:**
> The **Exclusion list** will be empty and must be edited by selecting the list and choosing the **EDIT** button!

EDIT

Once a list has been chosen to be edited from the available exclusions view, select the **EDIT** button opens the exclusions editor windows. For more detailed information on editing an exclusion list, please read this section.

RENAME

Selecting an exclusion list and choosing this button, allows for the renaming of an existing exclusion list.

Note:
The name of the file will not be changed. Only the name of the exclusion list by which it is represented.

DELETE

Selecting an exclusion list and choosing this button, allows for the deletion of an existing exclusion list.

Warning:
The files will be deleted permanently!

Active Exclusions

This list shows the currently active exclusions. Multiple exclusion lists can be active at any one time. For example, the user could create a generic exclusion list for all typical exclusions, multiple exclusion lists for virtual packages, one for creating App-V packages, and another for creating ThinApp packages. The generic exclusion list would always be active, at all times, and the exclusion list for virtual packages could be applied when and where needed. To remove an exclusion list from the active configuration, select the exclusion list and click the checkbox so that it is no longer selected. Parallel to that, to select an exclusion list to be active, select the exclusion list, and ensure that the checkbox is activated.

Note:
The exclusion list is not deleted. It will only be removed from the active configuration.

Note:
The installation of the RayPack includes three default exclusion lists which can be applied individually, all together, or in combination with one another.

The default exclusion lists delivered with RayPack are:

- **RayPack - All OS** is a generic exclusion list which should fit most environments.
- **RayPack - Legacy and MSI Installer** is a list that explicitly excludes legacy installation information (Wise, InnoSetup, InstallShield), as well as MSI installer information (logs, cache, Darwin Descriptors etc.).
- **RayPack - Aggressive** is an exclusion list that should be examined before being used, because as the name suggests, it is radical in what it excludes from the capture process. This list may not always be suitable for an environment.

WIZARD

This is where the settings for the **Capture Wizard** are defined.

Capture Mode

The capture mode defines the level of detail settings required to be entered during the capture process and the visibility of options which can be changed during the capture process.

Expert Mode

The **Expert mode** allows all options to be changed / modified during the capture process. This capture method is recommended for power users. The **Expert mode** is described in more detail

here.

Basic Mode

The **Basic mode** only allows for the minimal required options to be changed or modified. Default values and settings are used for the majority of options. This capture method is recommended for quick repackaging or for users who are not familiar with packaging technology. The **Basic mode** is described in more detail here.

> **Tip:**
> When RayPack is installed, the **Basic mode** is used as the default setting for the **PackRecorder** wizard. Click on the red **Expert mode** icon to activate the more flexible wizard configuration.

MSI OUTPUT

For more detailed information regarding the MSI file format and its functionality, please refer to the *MSDN MSI online help*.

Click the items to navigate to the various sections of profile configuration

Selecting the various options on this dialog (File options, Advertising, etc.) exposes settings for the individual items.

File Options

File compression
These options define how files are stored in the MSI when the package is created. There are three options available.

- **Uncompressed**
  The files are located externally in a *Pseudo* directory structure in the same directory as the MSI file.

> **Be aware:**
> The files and any directories need to be distributed together with the MSI file.

- **CAB**
  The files are compressed within CAB files, which are located in the same directory as the MSI file.

> **Be aware:**
> The CAB files must be distributed together with the MSI file.

- **MSI**
  The files are compressed and are stored in the file stream within the MSI file.

Check for .NET assemblies
If this option is activated, all .NET assemblies that have been captured during the **PackRecorder** wizard run will be organized within the MSI database tables `MsiAssembly` and `MsiAssemblyName`. If the option is inactive, .NET assemblies are handled in the same way as any plain file resource.

Use MSI HashFile Table for PE files
This option defines that any PE files that are to be included in the MSI package are to have entries in the `HashFile` Table. If this option is selected, then a Hash value is calculated based on the PE file, and this information is stored in the corresponding table of the MSI. This ensures that any file that has been corrupted or manipulated will not have the same Hash value of said file in the `HashFile` table and will prevent the installation of the MSI package. This is a security measure intended to prevent manipulation of MSI packages. It is recommended that this option is enabled.

Check for Merge Modules
This option enables the checking for Merge Modules during building.

Advertising

Convert the following shortcuts to advertised shortcuts
This option permits advertised shortcuts to be created and for what type of target file (of the shortcut)

- **All shortcuts**
  All shortcuts that are monitored and captured during the capture process will be authored as **Advertised Shortcuts** in the resulting MSI package.

- **Shortcuts to executable files**
  Only shortcuts to target resources that are executables will be authored as **Advertised**

Shortcuts in the resulting MSI package. Any other shortcuts will be created as standard shortcuts.

- **None (use only non-advertised shortcuts)**
  No shortcuts will be authored as **Advertised Shortcuts**. Shortcuts included in the MSI package will be authored as standard shortcuts.

For packages that are deployed in a corporate environment, it is recommended to select the option **All shortcuts**.

Use advertising tables for resources
Selecting this option allows RayPack to examine the captured package or the existing MSI project and where possible, add information to the advertising tables to create advertising information. If this option is not selected, then the resources that would normally be placed into the corresponding MSI tables are authored into the MSI package as *raw* information, usually in the `Registry` table. It is recommended that this option is enabled.

In the following a a brief overview about advertising and shortcuts can be found.

- **Shortcuts**
  The Windows Installer introduces a special type of shortcut that, while transparent to the user, contains additional meta data that the Windows Installer uses through its shell integration to verify the state of installation of the specified application prior to launching the application.

- **File Associations**
  The Windows Installer provides a mechanism for intercepting calls for an application associated with a document or so that when a user opens a document or file using the shell, the Windows Installer can verify the application before launching the associated application.

- **COM Advertising**
  The Windows Installer provides a mechanism that is hooked into the COM subsystem, so that any application that creates an instance of a COM component installed by Windows Installer (and configured to use this feature) will receive an instance of that component after the Windows Installer has verified the state of of installation for that component.

Using advertising, the MSI package can create so called *entry points* during the installation process. When utilized, these entry points let the Windows Installer know that an application (or part of an application) is to be used. The Windows Installer can then check if all of the required resources that this application needs are present on the system before the application is actually launched. If, for any reason, resources are not present, these can be (re)installed (self-repair). Afterwards the application is launched as normal. It is recommended to add / author as many *entry points* (advertising information) as possible into an MSI package, as this ensures that even if (for any reason) resources have been removed or are not present after the self repair, the application will function as required.

User Specific Options

These options define how user specific data is handled when creating an MSI package. Normally

an MSI package is installed on a per machine base or by using a software deployment mechanism (RayManageSoft, Altiris, SCCM, etc.) that is executed in a userless context. Therefore, resources that are to be installed in user specific locations (HKCU, My Documents, etc.) need to be handled differently. These options allow the control of how user specific resources are deployed on the target machine.

Use ActiveSetup for user specific data

If this option is selected, a component that contains the required information to implement an Active Setup is created. This component automatically installs any user specific resources using the MSI self-repair mechanism.

> **Tip:**
> For a more detailed explanation of the Active Setup, please read this section.

Name of the ActiveSetup Registry key

This is the name of a subkey that is to be created in the Active Setup Registry. By default, it is configured to be `[PackageCode]`, but it can be overwritten to match a companies defaults, for example `Company_[PackageCode]`.

Feature name

This is the name given to the feature that will be created in the MSI package when user specific resources are compiled into the MSI package. This feature will contain user specific components (see below). Entering a name that is easily identifiable ensures that when editing the package, the user can see at a glance which resources are machine specific and which resources are user specific.

Component name

This is the name given to the component that contains user specific resources. This component is automatically assigned to the user specific feature (see above). Entering a name that is easily identifiable ensures that when editing the package, users can see at a glance which resources are machine specific and which resources are user specific.

> **Tip:**
> For a more detailed explanation of the MSI technology, including Features and Components, please refer to the MSDN online help.

Shortcut Settings

These options define how advertising information and shortcuts are handled during the creation of the MSI package.

Exclude Desktop shortcuts

Selecting this option enables RayPack to attempt to automatically remove any information regarding **Desktop shortcuts** from the resulting `.msi` package on compilation.
It is recommended to **enable** this option for packages that are to be deployed in a corporate environment.

Exclude QuickLaunch shortcuts

Selecting this option enables RayPack to attempt to automatically remove any information regarding **QuickLaunch shortcuts** from the resulting `.msi` package on compilation.
It is recommended to **enable** this option for packages that are to be deployed in a corporate environment.

Exclude Uninstall / Remove shortcuts

Selecting this option enables RayPack to attempt to automatically remove any information regarding **Uninstall / Remove shortcuts** from the resulting `.msi` package on compilation.
It is recommended to **enable** this option for packages that are to be deployed in a corporate environment.

> **Tip:**
> For a more detailed explanation of the MSI technology, including advertising, entry points and self-repair, please refer to the *MSDN online help*.

MST OUTPUT

For more detailed information regarding the MST file format and its functionality, please refer to the *MSDN online help*. These settings determine how the PackRecorder handles the capture of Windows Installer based installations. During the capture process, the PackRecorder (where possible) will automatically recognize that a Windows Installer based setup is being executed. These options define how the resulting transform file is created after the capture process has been completed.

This section deals with the options available for creating `.mst` files (Transform files). Transform files are used to customize existing MSI packages. This is the recommend option for making changes to existing MSI installation packages.

Transform Options

These options define how the various windows installer functionalities are applied to the transform that is created by RayPack.

Transform Features
If this option is active, changes that have been captured that are relevant to the Features are added to the resulting transforms `Feature` table.

Transform Properties
If this option is active, changes made to properties are added / updated in the `Property` table of the resulting transform file.

Add Files
If this option is active, any files that have been captured during the capture process that **are not** present in the original `File` table, are added to the resulting transforms `File` table.

Update Files

If this option is active, there are changes to files that are present in the original MSI, but which are now different. For example, once the installation is complete a manual change is made to the file `application.exe`, which had the version 1.0.0 with the same file name, but a higher version 1.2. This information is then updated in the resulting transforms `File` table. Please use this option with caution.

Add Registry Entries

If this option is active, changes to the registry that **are not** present in the original `Registry` table are added to the `Registry` table of the resulting transform.

Update Registry Entries

If this option is active, changes to registry values that are present in the original `Registry` table are updated in the `Registry` table of the resulting transform.

> **WARNING**
> Use this option with caution because it may lead to hardcoded values in the resulting transform!

PACKBOT

The following settings define the default values for new PackBot tasks.

Delay second snapshot
The default value for delaying the second snapshot. For more information about the meaning of this setting, refer to the *Interactive Processing (Aka "Delayed Second Snapshot")* section in the following chapter *Basic Concepts*.

Power off virtual machines after completing the job
When activate (default value), after a virtual machine is not needed anymore it is automatically powered off. Uncheck this option to leave the machine running.

> **Note:**
> Even with this option active, machine is going to be reverted if next task(s) require it. This effectively means that the setting effectively affects machines that are not needed by any other task in the current processing sequence.

Default machines
This matrix visually represents the assignment of available virtual machines to particular task types. The virtual machines are shown on the left side of the table and supported formats on the top. When creating new tasks (or changing the type of existing tasks) the pool of the machine is arranged so that it matches the required package type. This is how it is possible to specify, for example, that conversion to virtual formats (which usually requires a presence of third-party runtimes / tools, for example Microsoft App-V Sequencer for App-V packages) runs only on those virtual machines which fulfill its prerequisites.

Vendor MSI settings
This setting defines the desired behavior of PackBot should a Windows Installer session be detected in the background AND the target format is MSI. Three available options are:

- o **Create Windows Installer transform (.mst)**
  This produces an MST containing changes to the original MSI, preserving original GUIDs, components structure etc.

- o **Create repackaged Windows Installer file**
  This simply converts the repackaged output to an MSI file and disregards the original vendor MSI

- o **Create both transform and repackaged .msi file**
  This combines both previous options. This option is the slowest, but also the most reliable since the user can decide on his own what was actually intended by that particular conversion process. This is a default value.

More information about vendor MSI behavior and options are available in chapter *Detection of Vendor MSI Setups*.

INI Editor

This section describes the use of the INI editor. This view allows the configuration of what resources are to be examined as INI type files, which INI filers are to be explicitly included/ excluded, and what is to be considered as an INI structure. This user interface section enables editing of the INI options files that are stored in the profile selected for use in RayPack.

To save any changes, click the **OK** button. To cancel any changes click the **Cancel** button.

Action

This column signifies if the entry is to be included or excluded from the generated project. This column can have the values **Exclude** or **Include**. To easily identify and distinguish between inclusions and exclusions, the entries are color-coded. Red signifying that the entry is to be excluded and green to signify that the entry is to be included. Using the regular expression technology for files allows the parsing of custom format (text based) files to be variable and configurable in the generated project.

Applied to

This column signifies when and how the entry is to be applied. This can have one of the following values:

- **FileExtension**
  The regular expression value in the RegExp column in combination with the Action value is applied to file extensions.

- **FileName**
  The regular expression value in the RegExp column in combination with the Action value is applied to file names.

- **ContentCommentStart**
  The regular expression value in the RegExp column in combination with the Action value is applied to define what are classified as comments within a file or when comments precede data.

- **ContentCommentInside**

The regular expression value in the RegExp column in combination with the Action value is applied to define what is classified as data when it has succeeding comments.

- **ContentSection**
  The regular expression value in the RegExp column is applied to define what parts of the file are the section headers.

- **ContentKey**
  The regular expression value in the RegExp column is applied to define what parts within the section headers are the key tokens.

- **ContentValue**
  The regular expression value in the RegExp column is applied to define what parts within the section headers are the corresponding values to the key tokens.

RegExp

This column contains the regular expression for this particular entry.

For example:
```
\.(ini|url|inf|INI|URL|INF)$
```
signifies in combination with the Applied to column set the **FileExtension** that any files that have the extension ini, url, INI, URL or INF are to be handled as ini files, and if found the defined structure are to be parsed as such. Where as the entry
```
^desktop.ini$
```
in combination with the value Exclude in the Action column, and **FileName** set as the value in the Applied to column explicitly excludes it from any processing as an ini file. This is similar to excluding the file via the Exclusions.

Description

This column is used purely for information, and in this case, it is recommended that a description for the entry is added.

Warning:
Discretion is advisable when editing the INI regular expressions, as errors here can cause serious problems and lead to the failure of parsing and recognizing ini files, and / or files with a similar structure to ini files.

Exclusions Editor

This section describes the use of the exclusions editor. This view allows the user to configure which resources are to be excluded in any projects that are generated from the capture process. This user interface section permits editing of the exclusion files that are stored in the Exclusions configuration directory. The exclusion files are XML based and use regular expressions to exclude or explicitly include resources when capturing an application installation. The more finely tuned the exclusion file(s) are, the less work and effort is required to create a clean and functional package. For more information regarding regular expressions, please see this section.

Warning:
It is recommended that the exclusion files are edited via the RayPack user interface only, as directly editing the files could cause errors and or instability during the operation of RayPack.

The Exclusions Editor is split into tabbed views with specific exclusion information for the following capture source types:

- **FOLDERS**
- **FILES**
- **REGISTRY**
- **ENVIRONMENT VARIABLES**
- **SERVICES**

Clicking the **OK** button from within any of the tabs saves all changes and closes the Exclusions Editor window. Clicking **Cancel** discards any changes and closes the window.

Read further to get an overview of the properties required for each exclusion item type, or directly move on to the general sections about editing and testing exclusion definitions.
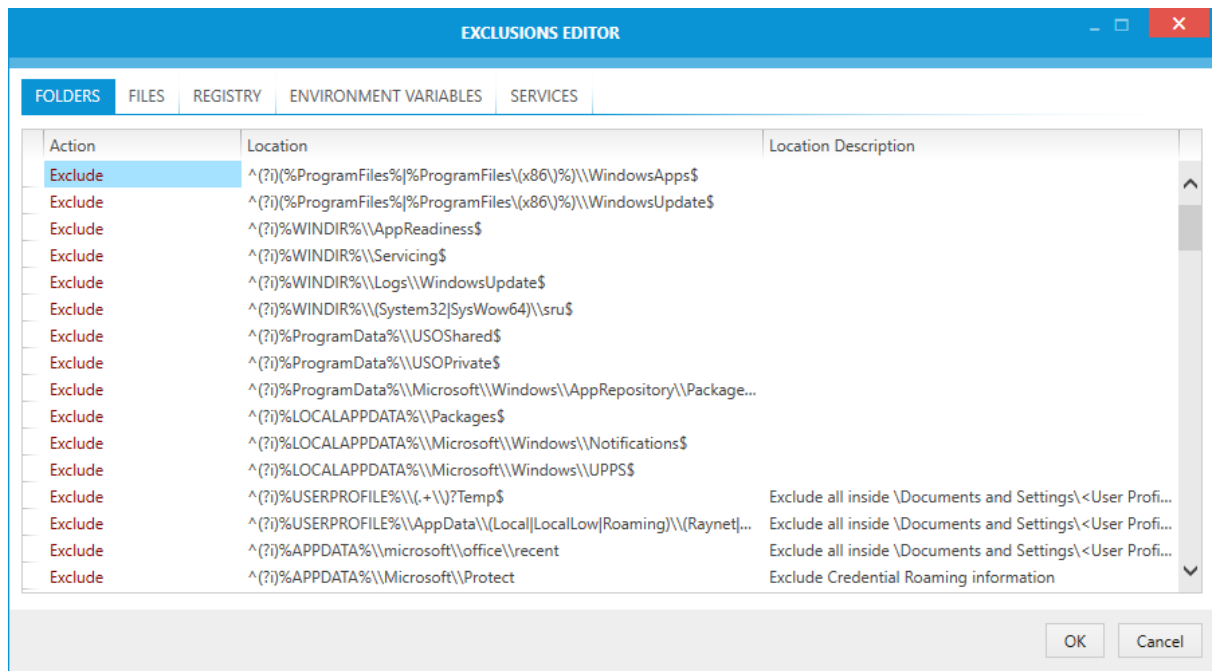
Folders

This section deals with folders that are to be explicitly excluded or included in a project generated from the capture process. The folders view has 3 columns:

Action
This column signifies if the entry is to be included or excluded from the generated project. This

column can have the values **Exclude** or **Include**. To easily identify and distinguish between inclusions and exclusion, the entry's are color-coded. Red signifying that the entry is to be excluded and green to signify that the entry is to be included.

Location

In this instance, this column sets the folder that is to be excluded or included. This is a string containing a regular expression.

For example:

`^%USERPROFILE%\\(.+\\)?Temp$`

Signifies that any directories named "Temp" in the current user profile. In combination with the Action **Exclude**, this directory, as well as, any files and or sub-directories will be excluded. It might be apparent that environment variables can be used within the regular expression. This column **cannot** be empty.

Location Description

This column is used purely for information, and in this case it is recommended that a description for the exclusion / inclusion is added.

Examples

Exclude any internal Windows Installer files

Action : `Exclude`
Location : `^%SystemDrive%\\Config\.msi`
Location Description : `Exclude all inside \Config.msi\`

Exclude any information from VMWare application installation directory on both 32 and 64bit systems.

Action : `Exclude`
Location : `^(%ProgramFiles%|%ProgramFiles(x86)%)\\Vmware`
Location Description : `Program Files\VMWare`

Files

This section deals with what files are to be explicitly excluded or included in a project generated from the capture process. The files view has 5 columns:

Action

This column signifies if the entry is to be included or excluded from the generated project. This column can have the values **Exclude** or **Include**. To easily identify and distinguish between inclusions and exclusion, the entry's are color-coded. Red signifying that the entry is to be excluded and green signify that the entry is to be included.

Location

In this instance, this column sets the folder that files are to be excluded or included. This is a string containing a regular expression.

For example:

`^%USERPROFILE%`

Signifies that the current user profile directory is the directory to use in the inclusion / exclusion.

If the Location is empty, then any subsequent information is for any location.

Location Description
This column is used purely for information, and in this case it is recommended that a description for the exclusion / inclusion is added.

Name
This is the name of the file(s) to be included / excluded. It is a string containing a regular expression.
For example:
```
^((?i)ntuser|(?i)usrclass)\.(?i)dat
```
Signifies that any files that contain ntuser or usrclass with the extension .dat are to be used for the inclusion / exclusion.

In combination with the Action and Location this transcribes to any files that are in the current user profile directory, that have the names `ntuser.dat` or `usrclass.dat` (case insensitive) are to be excluded.

Name Description
This column is used purely for information, and in this case it is recommended that a description for the exclusion / inclusion is added.

Examples

Exclude and pre-compressed driver inf files (pnf) from the generation of a project
Action : `Exclude`
Location : `^%WINDIR%\\inf`
Location Description : `Windows inf directory`
Name: `\.PNF$`
Name Description : `Compressed inf files (devices)`

Exclude the cookie index for the currently logged on user
Action : `Exclude`
Location : `^%APPDATA%\\Microsoft\\Windows\\Cookies`
Location Description : `Cookie location`
Name: `^(?i)index.dat$`
Name Description : `Cookie index`

Registry

This section deals with what registry entries are to be explicitly excluded or included in a project generated from the capture process. In this view, complete registry keys, specific named sub-keys, or specific values can be included / excluded.
The registry view, like other views has 7 columns:

Action
This column signifies if the entry is to be included or excluded from the generated project. This column can have the values **Exclude** or **Include**. To easily identify and distinguish between inclusions and exclusion, the entry's are color-coded Red signifying that the entry is to be

excluded and green to signify that the entry is to be included.

Location

In this instance, this column sets the root registry key that is to be excluded or included. This is a string containing a regular expression.

For example:

`^(HKEY_LOCAL_MACHINE|HKEY_CURRENT_USER)\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVers`

Signifies that any registry keys in the current user hive or local machine hive that are sub-keys of `Software\Microsoft\WindowsNT\CurrentVersion` are to be included / excluded (dependent on the Action setting) provided the Name column **is not** empty. This column **cannot** be empty.

Location Description

This column is used purely for information, and in this case it is recommended that a description for the exclusion / inclusion is added.

Name

In this view, this column represents the name of a String, Binary data, DWORD, QWORD, Multi-String Value or an Expandable String Value that is to be included / excluded (dependent on the Action setting).

For example:

`RegisteredOrganization`

In combination with the previous entries, this transcribes to ignore any changes to the registry values

`HKEY_LOCAL_MACHINE\Software\Microsoft\WindowsNT\CurrentVersion`
`\RegisterdOrganization` **or**
`HKEY_CURRENT_USER\Software\Microsoft\WindowsNT\CurrentVersion\RegisterdOrganization`

Name Description

This column is used purely for information, and in this case it is recommended that a description for the exclusion / inclusion is added.

Value

In this view, this column represents a specific name of a registry key value that can be included / excluded.

Value Description

This column is used purely for information, and in this case it is recommended that a description for the exclusion / inclusion is added.

Examples

Ignore any changes to the machine / users keys representing the DateLastConnected or ShutdownFlags

Action : `Exclude`

Location : `^HKEY_LOCAL_MACHINE\\SOFTWARE\\Microsoft\\Windows NT\` `\CurrentVersion\\`

Location Description : `Dynamic Information`

Name: `^(DateLastConnected|ShutdownFlags)$`

Name Description: `Dynamic system related values`

Exclude any changes to the Keyboard LED's.

Action : `Exclude`

Location : `^(HKEY_LOCAL_MACHINE|HKEY_CURRENT_USER)\\Control Panel\\Keyboard`

Location Description : `Control Panel Settings`

Value: `^InitialKeyboardIndicators$`

Value Description : `The initial State of the Keyboard LED's`

> **Note:**
> Ensure that the Name and Name Description columns are empty for the second example.

Environment Variables

This section deals with what environment variable entries are to be explicitly excluded or included in a project generated from the capture process.
The environment variables view, like other views has 5 columns:

Action

This column signifies if the entry is to be included or excluded from the generated project. This column can have the values **Exclude** or **Include**. To easily identify and distinguish between inclusions and exclusion, the entry's are color-coded. Red signifying that the entry is to be excluded and green to signify that the entry is to be included.

Name

This field contains the name of the environment variable to be included or excluded.

Name Description

This field is an information field and is optional. It is recommended to add text here that describes the name of the environment variable.

Value

This field is optional. If only part of the environment variable is to be included / excluded, it can be entered here.

Value Description

This field is an information field and is optional. It is recommended to add text here that describes the value of the environment variable.

Example

Ignore any changes to the Path environment variable
Action : `Exclude`
Name : `^Path$`
Name Description : `The System and User Path environment variable`

Services

This section deals with what services are to be explicitly excluded or included in a project generated from the capture process.
The services view, like other views has 7 columns:

Action
This column signifies if the entry is to be included or excluded from the generated project. This column can have the values **Exclude** or **Include**. To easily identify and distinguish between inclusions and exclusion, the entry's are color-coded. Red signifying that the entry is to be excluded and green to signify that the entry is to be included.

Name
The name of the service to be included or excluded. Please be aware this is not the display name of the service, but the actual name of the service as used in the SC Manager. For example, the printer service is named "spooler", so to include or exclude the printer service, the value to be entered into the Name column is **spooler** (in the example above, the InstallShield ROT service is to be excluded).

Name Description
This field is an information field and is optional. It is recommended to add a value here that describes the service, as the value in the Name column is not always descriptive.

Editing Exclusions

There are two ways of editing exclusion rules:

- Editing the expression inline
- Editing the expression using the RegExp editor/tester

To Edit the Expression Inline

1. In the exclusions editor, double click any entry
2. The inline editor will be enabled, allowing to change the content of the clicked cell
3. To save the changes, click outside of the cell or press **ENTER**
4. To discard the changes, press ESC

To Edit the Expression Using the Regular Expression Editor / Tester

1. In the exclusions editor, right click a row containing the cell that is about to be edited
2. From the context menu, choose **Test** menu item
3. The Regular Expression editor will be opened
4. To save the changes, press **OK** or **APPLY**
5. To cancel the changes, press **CANCEL**
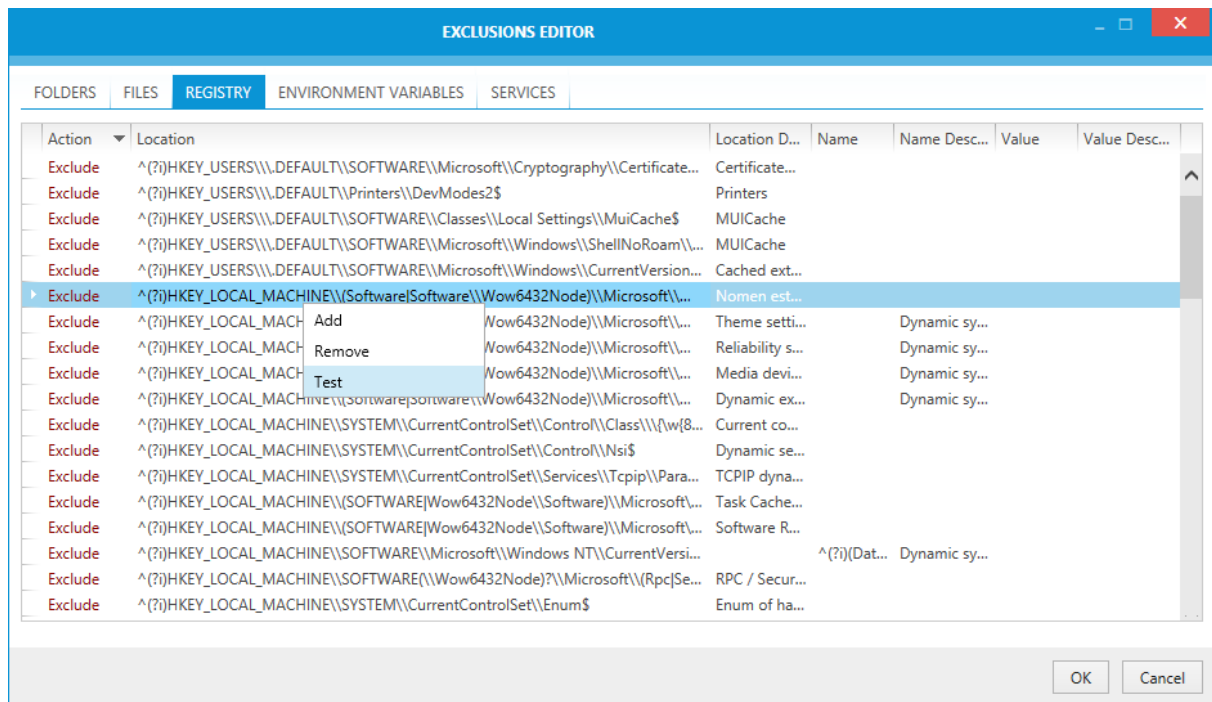
The exclusions are defined using standardized regular expressions. More details and a get started guide on how to dive into working with regular expressions are provided later.

Testing Exclusions

Using the Regular Expression Editor / Tester

The editor shows the content of the currently edited row, showing each allowed customizable value as a separate tab. Depending on the type of the object being edited (registry, file, folder, environment variable or service) different tabs may be available. For example, the rule for a registry entry contains customizable:

- Location (denoting the key)
- Name (denoting the name of the value)
- Value
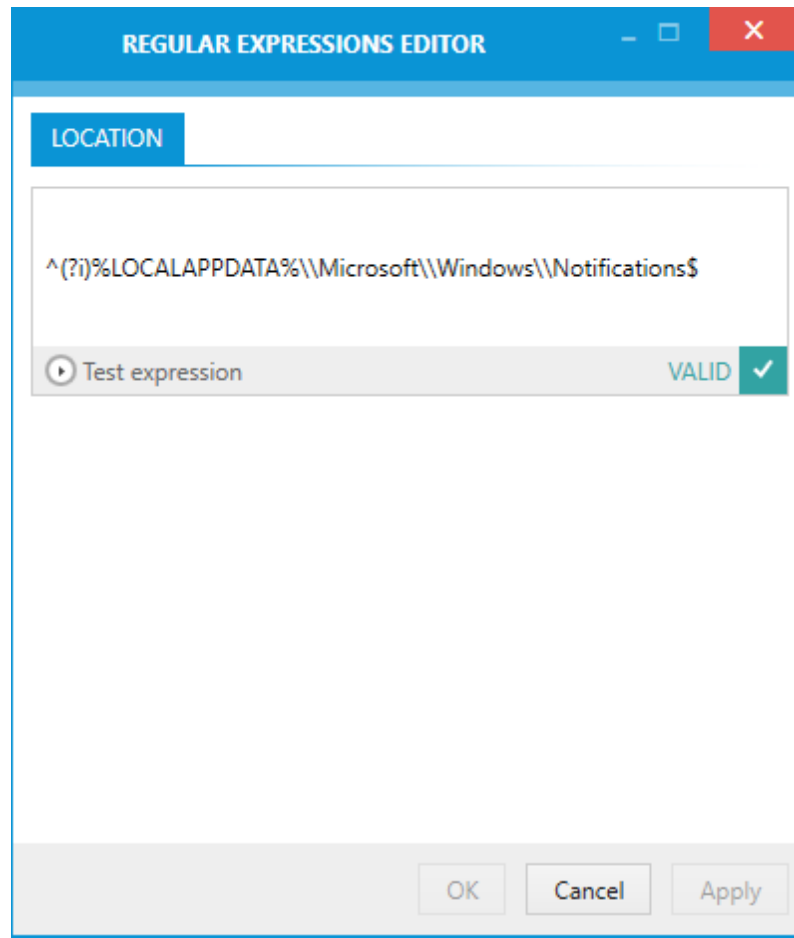
If any of these is left blank, it means that all values will match. For example, leaving the **NAME** blank would mean that all entries that match the **LOCATION** and **VALUE** are matching the whole rule.

Each tab is subdivided into two section:

- An expression editor
- An expression tester (hidden by default)

The expression editor is the field containing the regular expression used to match the excluded/included resource. The field has to contain a valid regular expression. Entering invalid value here would ignore the whole rule when the delta is generated from the captured resources.

In order to make sure that the regular expression is valid, a small indicator is shown under the expression field. It validates the typed input as it is being typed, it is being typed,pe, providing immediate feedback whether the rule is a valid regular expression or not.

For example the following expressions are valid:
- `^(?i)(HKEY_LOCAL_MACHINE|HKEY_CURRENT_USER)\\Software\\Microsoft\\Windows\\CurrentVersion\\(Installer|Uninstall|Setup)`
- `Logging`
- `(File|Folder)A`

Test the Patterns

Once the regular expression is valid and understandable by the RegExp engine, it is always a good idea to make sure that the rule can be actually used to find a certain string/pattern.

For example, the following expression:
`%UserProfile%\\MyFolder`
is a valid regular expression, but the following resources will not match the rule:

```
1.%USERPROFILE%\MyFolder
2.%USERPROFILE%\YourFolder
3.C:\Users\HardcodedUserName\MyFolder
```

The problem with the test string number 1 is a different casing. By default, all regular expression must match the casing of the tested string. A solution would be to add the `(?i)` bit at the beginning of the regular expression to match the test string regardless of the casing. The correct rule would be:
`(?i)%UserProfile%\\MyFolder`

The problem with the test string number 2 is that it simply points to another folder. If that folder has to be also excluded, the rule has to be adjusted:
`(?i)%UserProfile%\\(My|Your)Folder`

The problem with the test string number 3 is similar as the string number 2. One important difference is in the usage of environment variable in the expression string and an absolute path in the test string. During application of the exclusion lists, environment variables in the test string are resolved so that the full path will be matched. The regular expression tester can be used on different machines than the actual repackaging so that it does not resolve any variables. Therefore, the test result will be false, although it may actually return true when `HardcodedUserName` is installing the package and applying the exclusions filters.

In order to make sure that `HardcodedUserName` folder is always excluded, the solution is to add a new rule that will capture `MyFolder` using the full path. For example:
`(?i)C:\\Users\\([^\\)\\MyFolder`

Determining such issues during repackaging may be problematic and time consuming. A better solution is to always test the expression against desired results when the exclusion lists are built.

To Test the Pattern Against the Test String

1. Type the regular expression into the expression field.

   Make sure that the expression is valid (the indicator below the text field should display the VALID badge).

2. Click the toggle button **Test expression** under the expression editor.
   The test field is displayed.

3. Enter the desired string into the new text field. RayPack will evaluate whether the expression is matching the specified string.

4. If the status displayed under the tester shows **NOT MATCHING**, it means that the regular expression is valid, but was not found in the test string.

   That also means that when the delta engine evaluates a specified string during applying the exclusion filters, a given string will not match the exclusion rule.

5. If the status displayed under the tester shows **MATCHING**, it means that the regular expression is valid and was found in the test string.

   This means that when the delta engine evaluates a specified string during applying the exclusion filters, a given string will match the exclusion rule.

Designing

These settings affect how the PackDesigner creates packages and projects as well as resource handling within the projects.

The settings available for PackRecorder operations are grouped into a set of view tabs:

- **BUILD OPTIONS**
- **SAVE OPTIONS**
- **BEST PRACTICES**
- **NAMING CONVENTION**
- **MST**
- **ADVANCED**
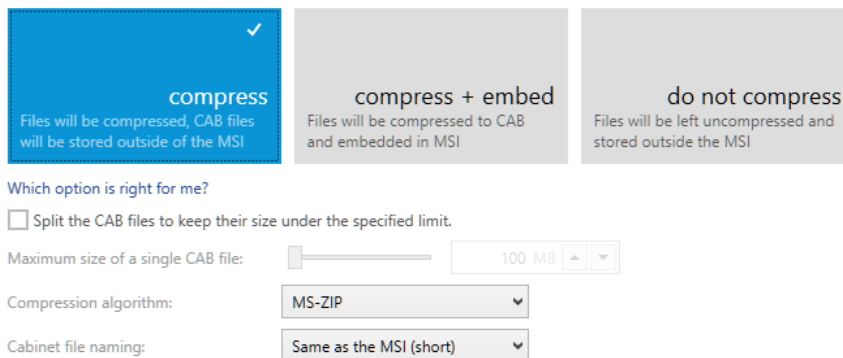
Build Options

The **Build options** tab specifies the default settings used when an RPP / MSI / MST project is built or rebuilt. These settings can be overridden per project (for RPP files) and per session for other types of files. Unless the templates define it otherwise, the settings defined here are also used as default values for new projects.

Compression and Media Layout

This setting controls how the source files of the output package should be stored.

- **compress**
  The files will be compressed to a CAB file (or several CAB files, see the section *Cabinet archives splitting*). Using this option will produce one relatively small MSI file and one or more `*.cab` files saved to the same location. This is the default method.

- **compress + embed**
  The files will be compressed to a CAB file and embedded into the MSI file. Using this option will produce one MSI file containing all necessary resources. Because of compression and embedding, this method takes longer to finish, but produces a file that is easy to deploy and does not require any companion files.

- **do not compress**
  The files will be stored without any compression. Using this option will produce one relatively small MSI file and a set of files in the source tree. Because no compression is needed, this method is the fastest method of building, but some additional considerations may be needed during the package deployment.

Maximum Size of a Single CAB File

> **Note:**
> Some of these settings are available only when the **compress** or **compress + embed** option has been selected in the Compression options section.

This setting controls the layout of produced CAB files. By default, one CAB per product is built by RayPack, but in certain scenarios it is desirable to split the compressed archive into several files. Typical situations requiring use of this option are:

- The compressed archive would be bigger than 2GB (a maximum size allowed by the Windows Installer technology).
- The project has more than 64 000 files (the maximum number of files in a single `*.cab` file).
- Performance reasons (low WAN speed, etc.)

In order to split compressed archives into more parts, the checkbox **Split the CAB files to keep their size the specified limit** has to be checked. Then, the slider below can be used to specify the required maximum archive size. When a precise value is required, the textbox on the right side can also be used.

When the project is build, RayPack will recompress all files into a specified configuration of compression layout. If the maximum size is bigger than the actual size of compressed resources, only one `*.cab` file will be created. Otherwise, several `*.cab` files will be created, all but the last one having the size of the specified maximum.

> **Note:**
> The visibility of the compression setting depends on the licensed product edition type.

Compression Algorithm

> **Note:**
> This setting is not available when the **do not compress** option has been selected in the **Compression options** section.

This setting controls the compression algorithm used to create the output `*.cab` files. There are three options available.

- **MS-ZIP**
  This option will use MS-ZIP compression to produce `*.cab` files. Compared to the option

None, the output will be smaller, but may take more time to finish. This option is faster than LZX, but produces slightly bigger output files.

- **LZX**

  This option will use LZX compression to produce `*.cab` files. Compared to the option **None**, the output will be smaller, but may take more time to finish.

- **None**

  This option will use no compression. The archives will be created faster, but there will be no size gain.

Cabinet File Naming

The names of the `*.cab` files created by RayPack are by default the same as the name of the package to which they belong. This dropdown box allows a finer control over the naming convention:

- **Same as the MSI (long)**

  The default setting. The final CAB(s) name will be the same as the file name of the MSI project (without extension).

- **Same as the MSI (short)**

  The final CAB(s) name will be same as the file name of the MSI project (without extension). If the file name without extension is longer than 8 characters, an 8.3-compliant name will be used (for example `MYPACK1.CAB`).

- **Data.cab**

  The final CAB names will be `Data1.cab`, `Data2.cab`, `Data3.cab`, etc.

- **Custom**

  A custom CAB name will be used. When this option is selected, a field below is used to enter the desired name for the CAB.

> **Note:**
> Custom CAB names have to be valid candidates for file names. Certain characters (for example ":", ">", "/" or "|") are not allowed. It is recommended to use only latin letters and characters (a-z, 0-9).

Signing and Wrapping

Signing and wrapping

- ☐ Sign the MSI each time the project is built
  - Note: This option requires a valid signature configuration
- ☐ After the build is complete, create an executable bootstrapper / wrapper
  - Executable (*.exe) bootstrapper (NSIS) ⌄

- **Sign the MSI each time the project is built**
  When this option is selected, RayPack will automatically sign the package once it has been built. Using this option requires that the settings for package signing have been defined in the **Signing + tagging** screen. To do so, users call the general settings manipulation dialog for the currently active settings profile with a click on the **signature configuration** button at the right-hand side of the **Sign the MSI each time the project is built** checkbox.

  Signing of MSI has an extra logic that makes sure that Media files (CAB) are also signed and correctly attributed inside the database. Whenever an MSI file is signed, RayPack signs its compressed resources and creates two MSI tables: **MsiDigitalSignature** and **MsiDigitalCertificate** which contain information about signed Media.

- **After the build is complete, create an executable wrapper**
  When this option is selected, RayPack will automatically produce a bootstrapper that can be used to install the package. RayPack supports three types of bootstrappers:

  o **Executable bootstrapper** (NSIS-compatible) - This is recommended for setup developers and to bundle a set of packages and prerequisites into one executable file.
  o **Batch file bootstrapper** - This is recommended to bootstrap a set of packages and prerequisites into a batch file to quickly install a package.
  o **PowerShell bootstrapper** - This is powered by PowerShell AppDeploymentToolkit - a modern approach to script complex installations using native PowerShell scripting.

> **Note:**
> Custom CAB names have to be valid candidates for file names. Certain characters (for example ":", ">", "/" or "|") are not allowed. It is recommended to use only latin letters and characters (a-z, 0-9).

Metadata Update

These settings control additional actions done by RayPack when the project is build.

- **Update MSI file properties (size, version, language) on build**
  When this option is selected, RayPack will make sure that the file properties in the `File` table are in sync with the source files.

- **Generate a new ProductCode each time the build is made**
  When this option is selected, RayPack will generate an unique `ProductCode` for the package each time it is built.

Executable Bootstrapper

The executable NSIS bootstrapper is a multi-purpose wrapper over an MSI + required prerequisites. It is a lightweight application which contains packed metadata, prerequisites, and

the installation package, and installs them in a specified order. Executable bootstrapper is a great way of consolidating an installation package consisting of many supporting files and prerequisites into a single `.exe` file, which can be then redistributed. All necessary prerequisites can be bundled together and installed if required.

When a bootstrapper is build it reads the package settings from the **Prerequisites** section and builds a script which ensures that dependencies are installed in a correct order, with proper conditions and installation failure detection. The process is automatic - as soon as the prerequisite is added to the project the necessary information is present.

> **Note:**
> Bootstrappers are created only when building an RPP project or rebuilding an MSI / MST package. Using the *File > Save / Save as* option does not start the build.

Command Line Switches

The executable bootstrapper supports command line parameters for:

- Silent installation
- Uninstallation
- Logging
- Passing command line parameters to the MSI database

The command line options are described in *Executable bootstrapper command line switches* chapter.

Customizing Bootstrapper Appearance

Refer to *advanced topics* section for more information about how to customize the bootstrapper.

Command Line Bootstrapper

The command line bootstrapper is a `.cmd` file that installs prerquisites and the required package, and supports a limited set of condition checking, including registry, file and system lookup + exit code validation.

> **Note:**
> Bootstrappers are created only when building an RPP project or rebuilding an MSI / MST package. Using the *File > Save / Save as* option does not start the build.

By default, when a command line bootstrapper has been selected, two files will be created next to the output MSI file:

- **($ProductName)_($ProductVersion)_install.cmd**
  (where `($ProductName)` and `($ProductVersion)` are values of `ProductName` and `ProductVersion` properties respectively.)

This wrapper can be used to install the package and its prerequisites.

- **($ProductName)_($ProductVersion)_uninstall.cmd**
(where `($ProductName)` and `($ProductVersion)` are values of `ProductName` and `ProductVersion` properties respectively.)

    This wrapper can be used to uninstall the package.

Be aware:
When the product is uninstalled, dependencies are left on the system. For example, a package that installs itself and Visual C++ Redistributable package as its dependency will not uninstall the Visual C++ package when the uninstallation is triggered from the command-line wrapper.

Customizing Command Line Bootstrapper

Refer to *advanced topics* section for more information about how to customize the bootstrapper.

Save Options

The **Save options** tab specifies the default settings used when an MSI / MST project is saved.

Compression

This setting controls how the source files of the output package should be stored.

- **compress**
  The files will be compressed to a CAB file (or several CAB files, see the section Cabinet archives splitting). Using this option will produce one relatively small MSI file and one or more `*.cab` files saved to the same location. This is the default method.

- **compress + embed**
  The files will be compressed to a CAB file and embedded into the MSI file. Using this option will produce one MSI file, containing all necessary resources. Because of compression and embedding, this method takes longer to finish, but produces a file that is easy to deploy and does not require any companion files.

- **do not compress**
  The files will be left uncompressed. Using this option will produce one relatively small MSI file and additional files will be saved to the same location.

> **Note:**
> These settings are not used when a project is being built. For build-related settings, refer to the Build options section.

Best Practices

Portable Executable Files

This setting determines how PE files are handled when added to the project using the PackDesigner. Please note that this setting only effects projects that are compiled as MSI files or when editing / creating an MST file.

Create components according to Best Practices

When this option is checked, files are added to the project and they are PE files. Then they will be assigned to their own component and the file will be assigned as the KeyPath of the component.

Portable executable file extensions

The predefined PE file types (.`exe`, .`dll`, .`ocx`), along with the file extensions .`chm` and .`hlp` are set as standard. In this field it is possible to add custom file extensions to ensure that the file type(s) are assigned to their own component and set as the KeyPath of the component. This ensures that files that are critical to the correct operation of an application are always present (see the self-repair functionality of the Windows Installer).

Font file extensions

The standard font files extensions (.`ttf`, .`otf`) are set as standard. In this field it is possible to add custom font file extensions to ensure that the file type(s) are assigned to their own component with a correct attribute set (reference counting) and registration in the Installer's `Font` table. This ensures that the fonts are correctly installed and only uninstalled if other programs installing them are already removed.

Components in User Profile

These settings determine how RayPack should treat the folders created in the user profile part (for example in the `AppDataFolder` directory).
If the checkbox is ticked, for each folder created in the user profile RayPack will:

1. Create a component,
2. Create a registry key in the `HKEY_CURRENT_USER` node,
3. Set the registry key as a KeyPath of the component,
4. Create folder operations `CreateFolder` and `RemoveFile` for the folder.

These actions ensure that the package passes ICE validation, but may create additional overhead and boilerplate MSI elements (additional components and registry which are not part of the product itself).

Additional settings allow a full control on naming convention for the created registry key. It is possible to define the key, name and value. MSI properties enclosed in square brackets are allowed, as they will be written directly into the MSI. Additionally, a special placeholder `$(ComponentId)` can be used. The token resolves to the name of the component that controls the installation of the registry key.

Naming Convention

The naming convention options deal with the following topics in the pack designer:

- Features
- Components
- Create and remove entries
- Drivers

Click on the individual items to view more information.

Features

This section deals with the naming and description of the main feature of any MSI / RPP based packages / projects.

Default name:

This is the display name shown in any UI. This value is used to populate the `Feature` table 'Title' column. See the MSDN MSI online documentation for more information.

Default description:

This is the description of the feature shown in any UI. This value is used to populate the `Feature` table 'Description' column. See the MSDN MSI online documentation for more information.

Components

This section deals with the naming of the components for none executable files (`exe`, `ocx`, `dll`, `sys`) in MSI / RPP based packages.

Default name of components without PE-executables:

This is the suffix added to a components that contains non executable files. The **@** character is replaced with the name of the directory in which the resources of the component (in this case files) are installed to.

Create and Remove Entries

This section deals with the creation of empty directories and the removal of files and directories in MSI / RPP based packages.

Default name of CreateFolder entry:

Whenever a `CreateFolder` table row is created, RayPack uses this default value to automatically determine a decent data object reference.

See the MSDN MSI online documentation for more information regarding the `CreateFolder` table.

Default name of RemoveFile entry:

Whenever a `RemoveFile` table row is created, RayPack uses this default value to automatically determine a decent data object reference.

See the MSDN MSI online documentation for more information regarding the `RemoveFile` table.

Drivers

Feature name:

Whenever drivers are added to a packaging project, RayPack automatically creates a new feature for storing driver information. Please make sure to use a name that is informative and suitable for swift orientation within the potentially large amount of features within a project. This makes it easier to identify drivers and install drivers correctly using the DIFxApp (Driver Install Frameworks for Applications) methodology.

MST

Opening

This section defines the behavior of RayPack when opening a file.

- **Scan for matching MST transform when opening MSI files**
  With this option enabled, RayPack activates extra logic when a user opens an MSI file. If RayPack determines that there is exactly one file in the same folder, which can be used as a valid Windows Installer transform, a prompt will be shown offering the user a choice whether the associated MST file should be opened as well.

MST Errors + Validation

These options define which transformation errors should be skipped when the transform file is applied. Additionally, it is possible to define additional criteria which must be met in order to apply the transform. These options are used each time a new MST file is generated by PackDesigner or PackTailor, including command line switches.

The summary of available options can be found in the MSDN documentation knowledge base: https://msdn.microsoft.com/en-us/library/windows/desktop/aa368246(v=vs.85).aspx.

Advanced

User Interface

If this option is activated, then every action that requires the deletion of information from the project will require confirmation. If this option is deselected, the entries will be deleted without having to confirm this. Please use with caution.

ICE Validation

If the identifiers of ICE validation rules are entered here, RayPack ignores them by default when

validations are executed. Rule identifiers have to be entered as a semicolon separated list. This default filter configuration may be overwritten by custom validation settings made manually before a specific custom validation run is executed.

Installation Directory

This option controls how the (default) installation directory is handled by PackDesigner (only applies to Windows Installer based projects).

Resolve at parent (default)
With this option selected, the PackDesigner will create an additional entry for the directory `INSTALLDIR` in the `Directory` table. Its `DefaultDir` value will be a single dot (`.`), meaning that the path is the same as the parents path. This method requires one additional entry in the `Directory` table, but provides an easier method to update and maintain the the project when editing the `Directory` table directly. This is the default setting that is used by PackDesigner.

Change identifier
With this option selected, the PackDesigner will change the identifier of the folder selected as the default installation directory. Any previous entries in the `Directory` table with this identifier will be renamed to avoid conflicts. This method produces cleaner output, but is slightly more difficult to maintain when editing the `Directory` table directly.

COM Information Extraction

The COM extraction related settings determine how RayPack handles COM registration data by default. However, manual COM registration extraction is always possible via the context menu for a specific file within the **Files and Folders** view of the Visual Designer or any components of the **Files** tab within the **Advanced** view.

Scan DLL, EXE, OCX and TLB files for COM information during the import
If this option is active, RayPack scans for COM registration information whenever a file is added to any RayPack packaging project. Extracted information is stored within the `Registry` table of the installer database.

Signing + Tagging

Package Signing

In order to be able to sign packages built by RayPack, there must first be some initial settings in place.

RayPack supports two different methods

- Signing with a certificate stored in a password-protected PFX file
- Signing with a certificate whose private key information is protected by a hardware cryptography module.

PFX File Data

Digital certificate file (PFX)
Please use the **BROWSE** button to define the path to the PFX file that has to be used for package signing. As an alternative, it is possible to type the path manually if logical path references, such as UNC path definitions, are required due to environment architecture reasons.
The certificate file must be available for successful package signing during package build procedures.

Certificate
Please use the **BROWSE** button to define the path to the certificate. As an alternative, it is possible to type the path manually If logical path references, such as UNC path definitions, are required due to environment architecture reasons.
The path to the certificate is optional, it does not have to be given for package signing during build procedures.

Password
Please enter the password connected to the given certificate.
The password must be available for successful package signing during package build procedures.

Use timestamp server
Defines whether a timestamp server has to be used during signing. If this option is checked, an internet connection is required to sign the package. Unchecking this option makes it possible to sign without an active internet connection, but the signature may only be valid as long as the certificate is valid. It is recommended to sign using timestamp server.

Hardware Cryptography Module Data

Certificate

The certificate name, for example "My High-Value Certificate"

Device Guard

Infrastructure-less signing with Device Guard Signing Service (DGSS) has been added in this release. In order to get started, visit the updated **Signing + tagging** tab, and perform a one-time sign-in with AzureAD credentials with a user that has necessary signing permissions configured in the Microsoft Store for Business Portal. After that, the packages can be signed with a certificate, for which the root certificate can be downloaded from Microsoft Store for Business.
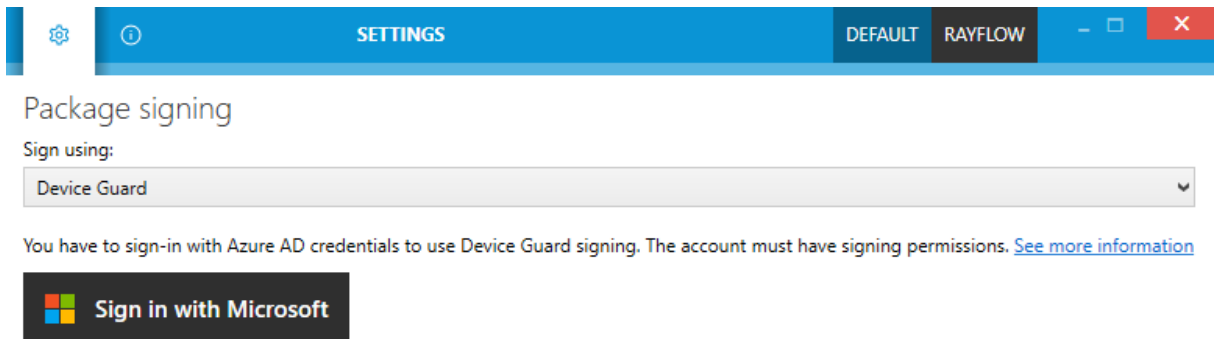
For more information about package signing, refer to the *signtool* documentation:
https://msdn.microsoft.com/en-us/library/windows/desktop/aa388170(v=vs.85).aspx

Software Identification Tag

RayPack provides an easy way to create an ISO/IEC 19770-2:2009 compatible tag for the purpose of software inventory. By default, the tags are generated for projects built (RPP / MSI / MST) and saved (MSI / MST) by *PackDesigner* module.

Create ISO/IEC 19770-2:2009 tags for MSI packages
Specifies whether the software tags are created. Untick the switch to disable creation of software tags.

Require Software Entitlement
Enable this option if the license compliance software should to try to check whether the current user is entitled to have this software installed on his machine.

Tag Creator Name
The name of the tag creator

Tag Creator ID
The registration ID identifying the software creator. It has to follow the *regid* pattern (see below)

Software Creator Name
The name of the tag creator

Software Creator ID

The registration ID identifying the software creator. It has to follow the *regid* pattern (see below)

Software Licensor Name
The name of the software licensor

Software Licensor ID
The registration ID identifying the software licensor. It has to follow the *regid* pattern (see below)

Regid pattern
The registration identifier is a string used to identify the entity. It uses the following format:

```
regid.YYYY-MM.reversedDomainName,division_optional
```

for example:

```
regid.2015-07.reversedDomainName,division_optional
```

> **Note:**
> RayPack validates the input in *regid* fields and displays a red icon next to it if the text is not valid.

Conversion

APP-V 5.X and APP-V 4.6
These tabs define the default options for App-V 4.6 and App-V 5.X options respectively. For more information about available settings, refer to the *Configuring App-V conversion* chapter.

THINAPP

The Build options view tab **THINAPP** contains all options for the build process of these virtual package files:

Compression

Compression method
Packagers may specify whether fast compression should be used or not. Fast compression can affect the start up time of applications, which is why using it is only recommended for those environments, where disk space has high priority.

Isolation

Isolation options specify the level of read and write access the virtualized application has on the physical environment. There are three settings available:
- **Merged**: This allows applications to modify the hosts operating system as long as the data

does not already exist on the host. If changes are assigned to already existing content, they are executed within the application's sandbox.

- **WriteCopy**:This causes all modifications made by the application to be executed within the application's sandbox. The virtualized application has read access to system elements if they are not existing within the sandbox.

- **Full**: This redirects all application operations and queries to the application's sandbox. The virtualized application will not be able to read any resource from the real host environment.

Folder isolation
Select one of the predefined options for the isolation settings regarding files and folders.

Registry isolation
Select one of the predefined options for the isolation settings regarding registry contents.

Sandbox Settings

Remove sandbox on exit
Activate this checkbox to delete the application sandbox as soon as the last child process of the application exits.

Sandbox name
Enter the name of the sandbox

AppSync URL
Define the location that stores an updated version of an application.

Clear sandbox on update
If the checkbox is enabled, the sandbox is assigned to be cleared after an update.

Compatibility and Troubleshooting

Set [AltArchitectureShortcut] flag
Activate the checkbox to solve issues that may occur if a mixed 32 / 64 bit application runs on a 64 bit Windows OS.

Set [WOW64] flag
Activate the checkbox to solve issues that may occur if a 32 bit application runs on a 64 bit Windows OS.

Enable command line
Activate the checkbox to activate the `cmd.exe` debug entry point to be able to solve problems by using the command line.

Enable registry editor
Activate the checkbox to activate the `regedit.exe` entry point to be able to solve problems by using the registry editor.

Thin-App Build Process

Keep ThinApp project

If this option is selected, each time when a ThinApp project is built, the source files and `package.ini` will be preserved in the build folder. Should the project require additional finetuning in the ThinApp editor, the source files can be used to quickly recreate and rebuild the package with new settings.

ThinApp bin directory

Specified the full local path where the ThinApp binaries are located. The files are required to build a ThinApp package (refer to *Prerequisite Software* for more information)

- **Use default Thin-App installation path**
  Instructs RayPack to look for ThinApp binaries in the standard installation location

- **Custom path**
  Instructs RayPack to look for ThinApp binaries in a custom location.

MSIX + UWP

This tab contains settings for the conversion to MSIX (using Desktop Bridge).

- **Accent color**
  Defines a color for accented elements (for example background of tiles). When the automatic color is used, the color is determined from the current accent color. Otherwise it is possible to select one of available brushes.

- **Architecture**
  The platform of the package.

- **Language**
  The language identifier as present in the manifest file.

- **Package signing**
  By default, all APPX packages must be signed before installing. If you plan to sign your package outside of RayPack or later, deactivate this checkbox.

- **MSIX Core**
  This setting defines whether MSIX Core runtimes should be automatically copied to the output folder with each MSIX build. You can also select which type of bundle should be copied (simple binaries, installer file or both).

RayFlow + PackageStore

RayFlow

This is where the settings for the connection with the RayFlow server can be configured.

RayFlow

RayFlow Server URL Address:

Enter the address of a running RayFlow instance which will be used to connect to project, read and exchange data etc.

The URL address of the RayFlow Server is shown here. Click on the [🌐] button to open the link to the server in a web browser.

> **Be aware:**
> The URL address does not have to be available at the time settings are saved. However, the instance must be up and running before any RayFlow related activity can be carried out.

Package Store

This is where the settings for the Package Store integration can be defined.

In order to use the Package Store module, the user must enter his API key into the **API key** field. Alternatively, the value is already filled with a key if the current product license has the Package Store module in.

Other options available:

- **Create package report**
  When enabled, a PDF report for each package is created and saved together with package files. The report contains information about the settings used for customization as well as the information about installation and repair.

- **Use proxy**
  When enabled, the user may provide his own proxy settings to be used to connect to the Package Store.

Deployment

This view contains various settings used by the **Deployment wizard**.

SCCM

These settings are used by default by the SCCM Deployment module.

- **Deploying type**
  The default type of the SCCM Deployment (recommended: application).

- **Server name**
  Name or IP address of the SCCM server.

- **UNC Share Path**
  The folder where deployment package sources are saved to.

- **Authentication**
  Determines whether current user credentials (Windows Authentication) or custom credentials are used. For the second option, it will be necessary to enter them once here.

Intune

These settings are relevant for deployment to *Intune*. You should configure them first before accessing *Intune*-related functionality for the first time.

The following checklists shows necessary prerequisites and actions that must be configured or provided by your Azure AD administrator:

- Registered application (app registration) in Azure AD
- The app must have required permissions for device management
- Admin consent should be granted

Client ID and secret must match the values from the app registration in Azure AD. The domain must match your tenant domain.

Virtual Machines

This view contains settings used by the **PackBot** and **PackTailor**.

By default, the list of machines is empty. Each machine that needs to be used has to be imported first. The view is divided into three sections:

- **Function Buttons (Add, Remove, Search)**
  This panel is used to add a new machine, remove a selection, and search for machines in the list. In order to filter the list, type a few letters into the search field and the list will be filtered automatically. To clear the results, either click X or clear the content of the search box.

- **List of Machines**
  This is a list of all the machines defined for RayPack Studio products. Each machine is represented by an icon representing its type (**VMware Workstation**, **VMWare ESX**, or **Hyper-V**), machine name, and (if configured) the name of the snapshot that is to be used. Select any machine from the list to see its details shown on the right side.

- **Virtual Machine Editor**
  The right panel contains various controls and inputs about the currently selected machine. Later in this chapter their meaning is discussed together with the slight differences between the different machine types.

To Import a Virtual Machine...

> ⚠️ **Warning:**
> It is not possible to import virtual machines if RayPack is already installed on a virtual machine!

1. Press the **+** button to expand the new machine drop-down menu.
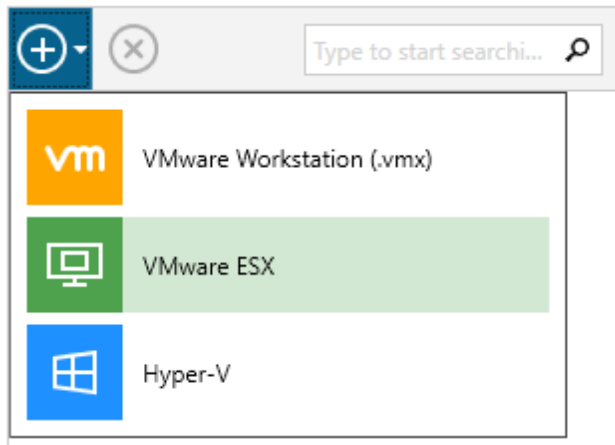
2. Select type of machine to import.
3. The machine will be added to the list. Use the editor to configure the details for the machine.

To Delete a Virtual Machine...

1. Select a machine from the list.
2. Press the **X** button in the functional panel.

Editing Virtual Machines

The virtual machine editor is divided into three tabs. The visibility of tabs depends on the type of the currently selected machine.

Virtual Guest Tab

This tab contains common properties which are valid for all types of supported virtual machines.

- **Display Name**
  This is the name under which the machine is shown in the list. For VMware Workstation and ESX, the name can be an user-friendly string which helps to determine the purpose of the machine (for example *My Repackaging Machine*, etc.). For Hyper-V machines, this value must be equal to the name that is visible in the Hyper-V manager.

- **Machine Type**
  The type of the current selection. This setting is read-only.

- **Path to VMX File** (Only Workstation and ESX)
  This is the full path to `.vmx` container file. For Workstation machines, this must be a full absolute path to a file, for example `C:\Virtual\Machine\Machine.vmx`). For ESX, the name must include a datastore token (name surrounded by square brackets) followed by the relative path of the `.vmx` file. The path can be viewed in the properties dialog directly in a vSphere client. A sample value would be `[DATASTORE]Machine \Machine.vmx`.

Be aware:
Failure to provide valid file paths to the VMX file makes the machine unable to work with. Machines with invalid paths or unsupported file formats will be marked as invalid and the user will not be allowed to use them..

- **Computer Name** (Hyper-V Only)
  This should be the full DNS computer name of a virtual machine.

- **Authentication**
  This setting configures credentials used to connect to a virtual machine. This field is always required, even if auto-logon is enabled on the guest Operating System. In order to configure the credentials, press the button **Configure credentials...** and enter the user name and password. In order to log in as a domain user, use `DOMAIN\USER` syntax.

Be aware:
Passwords are stored in an encrypted form in a text file. This however offers by no mean a state-of-art security. Expect that these values can be decrypted easily and as such never store confidential data on machines with shared access to RayPack configuration files.

- **Snapshot**
  Defines which snapshot is to be used. While it is generally possible to always use the last snapshot (the current one), in a production environment the name of a snapshot should always be specified either by typing the name manually in the textbox below, or by pressing the **...** button and using the selector dialog.

Host Tab (Only VMware ESX and Hyper-V)

This tab contains the properties of the hypervisor server. Because VMware Workstation uses solely local files, the tab is not visible when working with Workstation machines. When this tab is shown, all values presented here are mandatory.

- **Host Address + Port**
  Defines the location and the port of the host. Contact the responsible administrator to find out these values. By default port `5985` is used for Hyper-V and `443` for ESX machines, but they can be overridden individually.

- **Authentication**
  This setting configures the credentials used to connect to the host. This field is mandatory, even if auto-logon is enabled on the guest operating system. In order to configure the credentials, press the button **Configure credentials...** and enter the user name and the password. In order to log in as a domain user, use the `DOMAIN\USER` syntax.

Note:
The credentials for the host are usually not the same as the credentials that are used to connect to the VM.

Be aware:
Passwords are stored in an encrypted form in a text file. This however offers by no mean a state-of-art security. Expect that these values can be decrypted easily and as such never store confidential data on machines with shared access to RayPack configuration files.

Additional Tab

- **Comments**
  This is an additional invisible field that can be used for notes and remarks about the machine.

- **Execute Tasks on a Virtual Machine as Administrator**
  By default, the tasks are executed as Administrator to ensure that the setup files changing machine files and registries are allowed to do this. For certain environments and operations this behavior may not be desired. Unchecking this checkbox makes RayPack start the tasks as the invoker.

Be aware:
This option is only recommended for troubleshooting or for tasks not requiring administrative privileges. Certain <%PRODUCTPACK%> repackaging functions may not work when the task is not running as an administrator.

Snapshot Selector

To start the **Snapshot Selector** first select the **Before starting the machine revert it to the following snapshot** option that is available beneath the Snapshot heading in the **Virtual Machines** tab in the **Settings**. After the option has been selected, click on the **Browse [...]** button to open the **Snapshot Selector**.
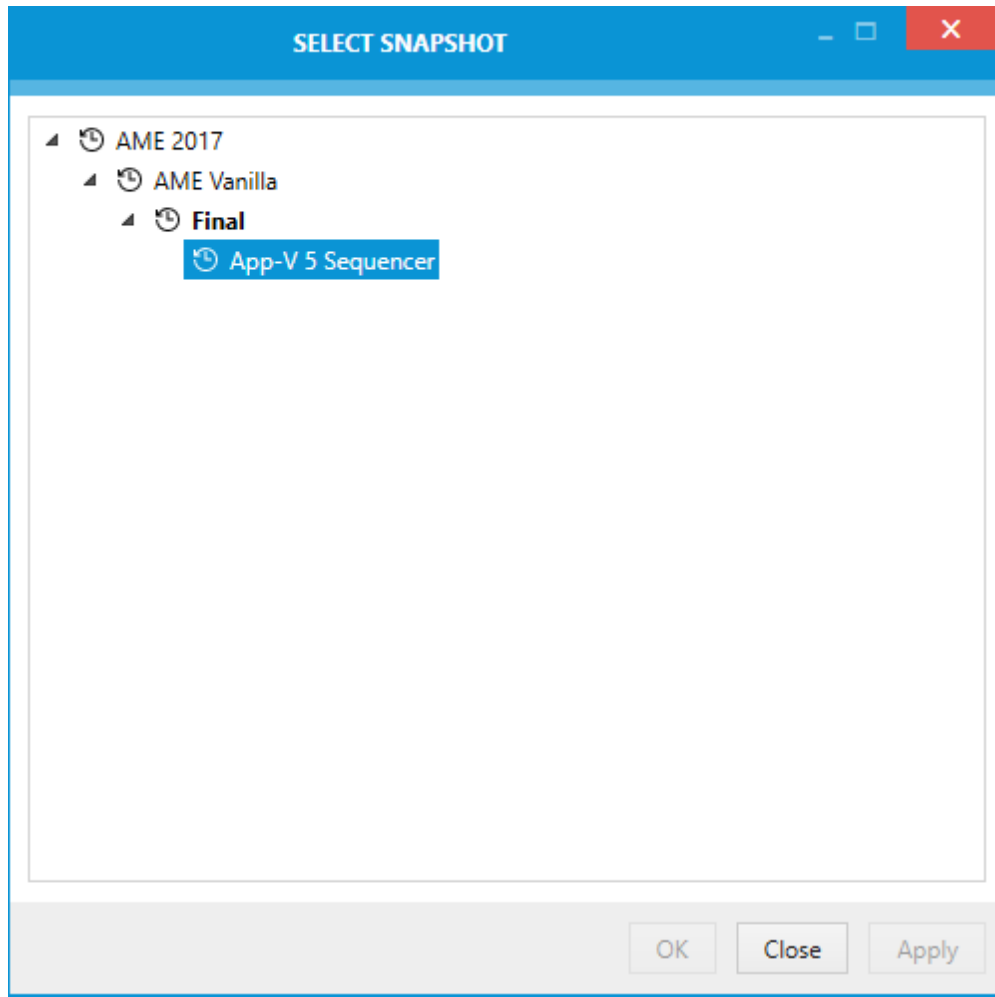
In the **Snapshot Selector** the available snapshots for the virtual machine are shown in a tree structure. Either select a snapshot or close the **Snapshot Selector** without changing the current selection. The following options are available in the **Snapshot Selector**:

- **OK**: This option is used to close the **Snapshot Selector** saving the currently selected status.
- **Close**: This option is used to close the **Snapshot Selector** without saving the changes.
- **Apply**: This option is used to save the currently selected status without closing the **Snapshot Selector**.

Preparing Virtual Machines

Depending on the type of the virtual machine, some extra steps may be required for it to work with RayPack. The following guide outlines the required steps.

General (All Types)

- The guest machine must be able to see the host machine by its DNS name. For example, if the host machine name is MWS0189, then the guest must be able to resolve its name to an IP

address.

- On the host machine, incoming traffic on a specific port must be enabled. The same remote port must be allowed by the firewall configuration on the guest machine. The port range used by default is `48654-48999`, starting from the lowest available. The port range can be configured by changing the configuration files.
- It is highly recommended to disable User Access Control (UAC) on the guest machine. This is especially important for silent operations, so that the setups can be started without user interaction.
- It is highly recommended to enable auto-logon to the machine for the configured user. This ensures that the machine can be automatically powered on and started, without waiting for the user to log-in interactively. The following link describes how to configure it: https://support.microsoft.com/en-us/help/324737/how-to-turn-on-automatic-logon-in-windows
- Even if auto-logon is enabled on the guest machine, the matching credentials must be provided in the RayPack configuration. These are used to execute the programs and commands on the VM in the right context (so that the user is able to see dialogs etc.), but due to technical limitations on virtualization solutions they cannot be used to bypass the initial login/lock screen.
- Although possible, we do not recommend using machines without snapshots. It is recommended to have at least snapshot on the VM, and have it selected in the VM configuration.

VMware Workstation

- VMWare Tools must be installed on the guest machine.
- VIX API must be installed on the host machine (this is already the case if VMware Workstation is installed on your host machine).
  o There is a known issue in VMware Workstation 14 and later, where the necessary COM interfaces are not registered by default. You can fix this problem by applying the steps described in the following Knowledge Base Article: *https://raynetgmbh.zendesk.com/hc/en-us/articles/360000277786-RSC200351-Executing-Virtual-Machine-Operations-on-VMware-Workstation-14*.

VMware vSphere / ESXi

- VMWare Tools must be installed on the guest machine.
- PowerShell 3.0 or higher must be installed on the host.
- PowerCLI in a version that is compatible with the vSphere / ESXi version must be installed on the host machine. See *https://code.vmware.com/web/dp/tool/vmware-powercli/12.0.0* for more information.

	VMware PowerCLI															
	12.0.0	11.5.0	11.4.0	11.3.0	11.2.0	11.1.0	11.0.0	10.2.0	10.1.1	10.1.0	10.0.0	6.5.4	6.5.3	6.5.2	6.5.1	6.5.0
▼ VMware vSphere Hypervisor (ESXi)																
7.0	✓	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
6.7 U3	✓	✓	✓	✓	—	—	—	—	—	—	—	—	—	—	—	—
6.7 U2	✓	✓	✓	✓	✓	—	—	—	—	—	—	—	—	—	—	—
6.7 U1	✓	✓	✓	✓	✓	✓	✓	—	—	—	—	—	—	—	—	—
6.7.0	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	—	—	—	—	—	—
6.5 U3	✓	✓	✓	✓	—	—	—	—	—	—	—	—	—	—	—	—
6.5 U2	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	—	—	—	—	—	—
6.5 U1	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	—	—
6.5.0	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
6.0 U3	—	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
6.0.0 U2	—	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
6.0.0 U1	—	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
6.0.0	—	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
5.5 U3	—	—	—	—	—	—	—	✓	✓	✓	✓	✓	✓	✓	✓	✓
5.5 U2	—	—	—	—	—	—	—	✓	✓	✓	✓	✓	✓	✓	✓	✓
5.5 U1	—	—	—	—	—	—	—	✓	✓	✓	✓	✓	✓	✓	✓	✓
5.5	—	—	—	—	—	—	—	✓	✓	✓	✓	✓	✓	✓	✓	✓

Hyper-V

Configuring Hyper-V requires a few extra steps. A comprehensive guide can be found in the following Support Knowledge Base Article in the Support Panel:

https://raynetgmbh.zendesk.com/hc/en-us/articles/360000308223-RSC200355-How-to-Configure-a-RayPack-Studio-Application-for-Hyper-V.

A short checklist of the prerequisites for Hyper-V machines:

- WINRM must be configured with `TrustedHosts` entries on both guest and host.
- PowerShell 3.0 or higher must be installed on the host.
- RayPack Studio Tools for Hyper-V must be installed on the guest machine and be a part of the base snapshot (so that they start each time when the machine boots after reverting to a selected snapshot).

The following checklist helps to find and fix any possible issues when working with Hyper-V machines:

1. Is PowerShell 3.0 installed (both on the guest and the host machine)?
   a. Check `$PSVersionTable.PSVersion` in PowerShell.

2. Is the machine properly configured in the **Settings** > **Virtual Machines** screen (pay attention to any hardcoded IP addresses which may be dynamically assigned by DHCP).

3. Is RayPack Studio Tools for Hyper-V installed on the Guest machine? Is the process `vm-proxy.exe` from RayPack Studio Tools for Hyper-V running?

4. Is WINRM configured?
   a. Check `winrm qc`.

5. Does WINRM have proper `TrustedHosts` entries on both the VM and the server?
   a. `winrm s winrm/config/client '@{TrustedHosts="RemoteComputer"}'`.
   b. `winrm g winrm/config/client` - shows the current `TrustedHosts` lists.
   c. More information: *https://technet.microsoft.com/en-us/library/ff700227.aspx*.

6. Does WINRM have a connection to the VM and vice-versa?
   a. `- Test-WSMan -ComputerName IP`.

7. Are all necessary ports unblocked on the physical machine?
   a. The default port range is `48654-48999`.

Changing TCP / IP Configuration

In some cases it may be required to use custom port ranges, timeouts, etc. for VM related functionalities.

The following table summarizes the available options:

Setting name	Default value	Description
`TcpIpDefaultPort`	48654-48999	Port range used for TCP/IP communication. Use minus (-) and comma (,) to indicate which ports are valid for incoming communication. Make sure that these ports are not blocked by your firewall. PackBot tries to find the first valid free port and listens for it from lower to higher numbers.
`TcpIpMaxRetry`	3	The maximum number of retries before asserting that the machine is not available.
`TcpIpDefaultReceiveT:`	240000	Reverts to the default value if Windows does not define its own timeouts.
`TcpIpDefaultSendTimeout`	240000	Reverts to the default value if Windows does not define its own timeouts.

The options can be configured by editing the configuration file `RayPack.exe.config`. Each option is defined as a pair of key and value in the `<appSettings />`. For example, to change the default port to 50000 and the maximum number of connection retries, configure the following:

```
[...]
<appSettings>
    <add key="TcpIpDefaultPort" value="50000" />
    <add key="TcpIpMaxRetry" value="5" />
[...]
```

> **Note:**
> If these options are not present in the configuration file out-of-the-box, the defaults from above should be used.

PackRecorder

PackRecorder is RayPack's tool for capturing, repackaging and editing application setups. It uses snapshot technology to create Delta files, which are the basis for further steps of package project generation and application resource manipulation.

The PackRecorder can also be used to "capture" Windows Installer based installations. However, it is not recommended to do so in standard packaging processes. The processing that the PackRecorder carries out when capturing Windows Installer based installation can be set here.

PackRecorder Wizard has two modes of operation, *Expert* mode and *Basic* mode. The mode to be used is set using the option here.

Capturing Applications

This section describes the setup capture process using PackRecorder.

Launching the PackRecorder wizard for setup capturing is available at two central places:

- By clicking the **create a new project** tile from the Home Screen
- By clicking the **FILE** tab of the Main Toolbar

Either way, the **NEW** dialog of the **FILE** menu is displayed. Initializing a new PackRecorder wizard run is achieved by clicking on the **repackaging** button at the right-hand side of this dialog. Please refer to the Common Dialogs section to read more about the FILE menu.
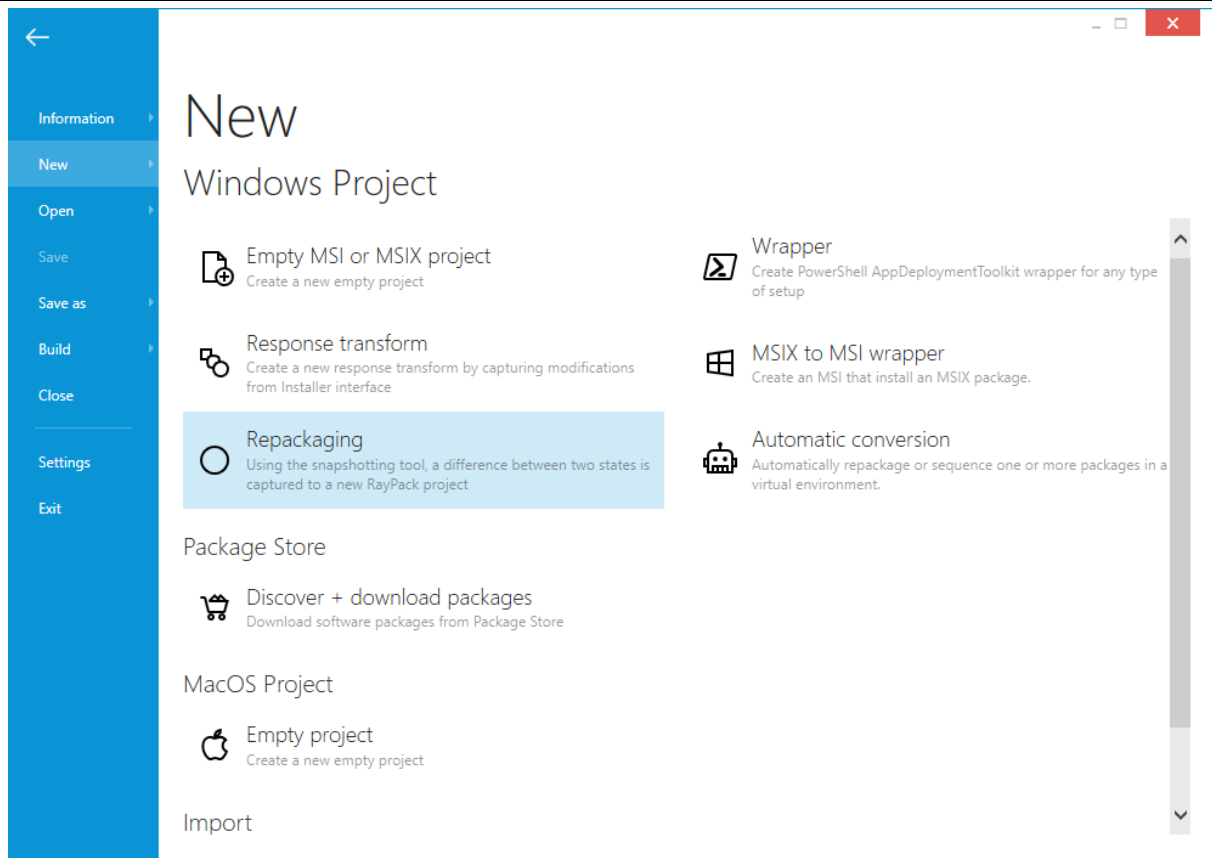
Click on the repackaging button to proceed.

Tip:
It is recommend to ensure that any services or applications that are not required by the application that is to be captured, be disabled or shutdown. This helps immensely in creating a "clean-capture" and requires less work when editing the resulting capture project.

Wizard Modes

Some of the pages described in the following sections may be missing depending on the capturing mode specified in the current profile. By default, RayPack uses the **BASIC mode**, which is limited regarding available options and wizard steps. The **EXPERT mode** triggers the recommended procedure for experienced packagers with advanced option requirements.

To permanently switch between the modes close the wizard and *go to the profile settings*.

A temporary mode switch may be performed on the first page of the wizard. However, the local changes made to the mode setting from within the wizard interface take effect on the current wizard run, not the permanent system setting.

Wizard Navigation

The interface of the PackRecorder wizard offers several ways to navigate between the procedure steps:
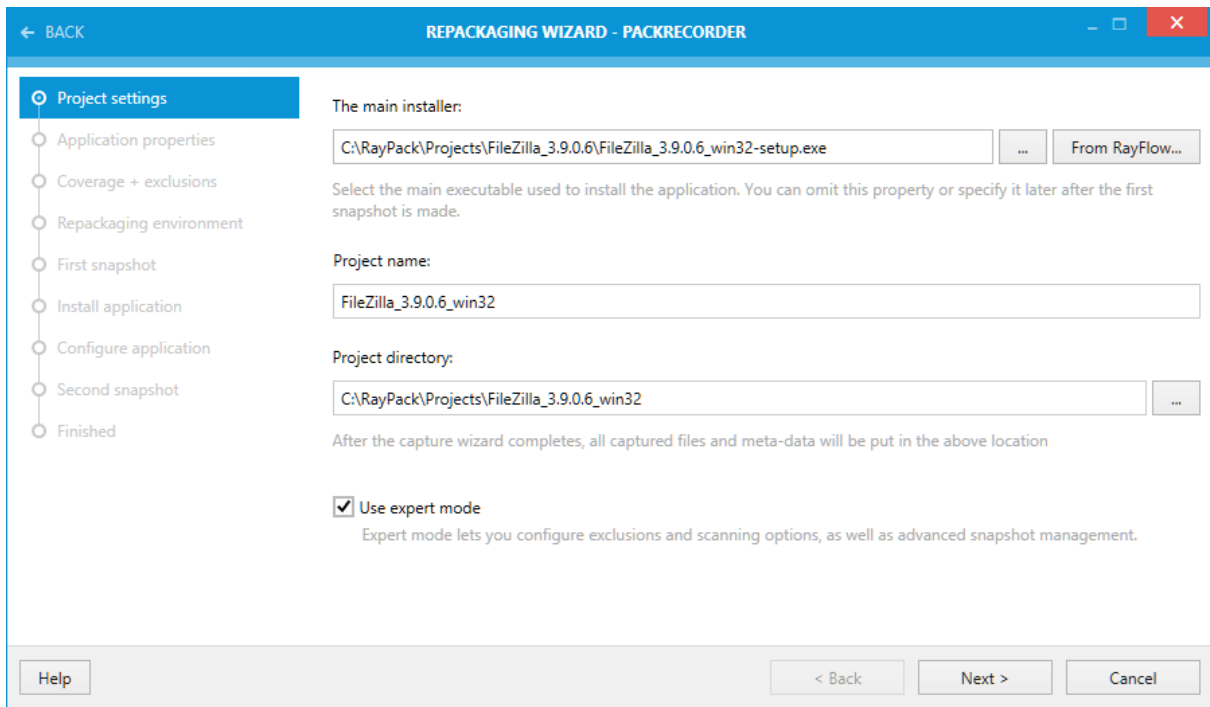
The **< Back** and **Next >** buttons at the bottom of the dialog window allow to move either one wizard step ahead or back. However, the **Next >** button will remain disabled until the mandatory fields of the current step contain valid values.

The **wizard menu** at the left hand side of the dialog window allows to directly jump back to any prior wizard step. Since walking through the whole procedure means to proceed upon prior settings and intermediate results, it is not possible to jump forward by using the wizard step menu. Once an earlier step has been called by using the menu column, it is required to execute the succeeding steps once again.

The **Cancel** button at the bottom of the wizard window is available at any time during wizard execution. Clicking the **Cancel** button aborts the current capture procedure. If snapshots have been saved during the current wizard run, they will persist even after hitting **Cancel**. All other recorded resources will be discarded.

Project Settings

This screen shows the general settings required to create a capture based project. There are two modes available, **BASIC** and **EXPERT** mode.

The Main Installer

This field / entry specifies which executable (installation) is to be executed. This field is optional because later in the wizard an option for choosing an executable file to run will present itself. Naturally this field may be left empty if a "manual" installation (a.k.a., copy and paste installation) should be captured.

A file from RayFlow can be opened by clicking the **From RayFlow...** button. The exact procedure is described in the section *Repackaging Files from RayFlow*. The remaining steps are the same as below.

Project Name

The name of the project. This name is used to create a subdirectory in the main project directory which is defined in the profile. This field is mandatory and requires a valid value. As the project name is typed, the name of the directory created under the default project directory is automatically changed to reflect this.

Project Directory

This is the default root directory into which the project will be saved (appending the name of the project entered in the **Application name** field). The default directory location is contained in the currently active profile, and can be edited *here*. The project directory location can be changed here, but this will only be valid for the current project. This field is a mandatory field and requires a valid value.

Use Expert Mode

The **EXPERT** mode allows to adjust the options applied to the current wizard to run beyond the overall settings defined for the PackRecorder wizard runs within the settings profiles. If the general recorder mode is set to **EXPERT**, the **Use expert mode** checkbox is already activated when this view is loaded. If the global mode setting is defined to be **BASIC**, the checkbox is not active. However, switching the mode state takes effect on the current wizard run, not the permanent setting stored for the RayPack instance. To change the global PackRecorder wizard mode setting, please refer to the *Settings* section.

Repackaging Files from RayFlow

In Order to Select a File for Repackaging from RayFlow:

Be aware:
In order to use this functionality you have to configure your RayFlow connection first. See chapter Settings > RayFlow for more information.

1. In PackRecorder Wizard, click **From RayFlow...** button.

2. A new overlay window will be shown, prompting to select RayFlow phase, task, and a file belonging to a task.
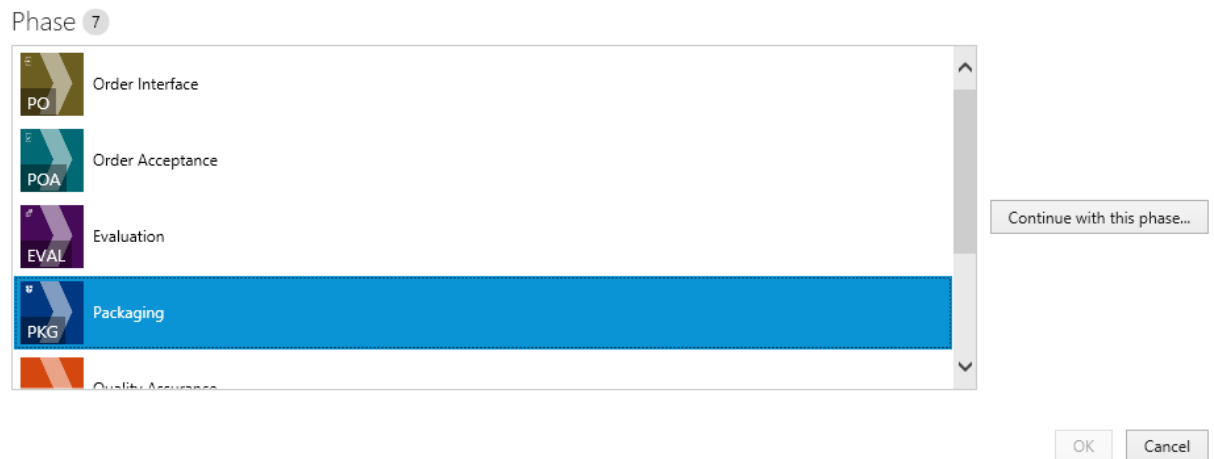
3. Select RayFlow phase which contains the target task for opening. The counter above the list shows the total number of phases that are visible to the current RayFlow user.

4. Click **Continue with this phase...** to proceed to the task selection with the currently selected phase. Press **Cancel** to go back to the backstage menu.

5. Next, select the target task for opening:

The number displayed above the list shows the total number of tasks that are visible to the current RayFlow user. The results can be narrowed by ticking **only mine** option, which hides all packages not belonging to the current RayFlow user. The task browser displays the first 5 data fields from RayFlow to enable easier identification (refer to the RayFlow documentation about setting up the data fields).

6. The results can also be narrowed by searching, using the search box in the top right corner of the list.

7. Once the required task is selected, press **Continue with this task...** to proceed to the file selection. Press **Cancel** to go to the backstage menu. In order to go back to the phases view, click the arrow on the left side of the list or the partially transparent area containing the tiles representing available phases.

8. On the next screen, select the file to open:

9. The number displayed in the top left corner represents the number of files that are present in the currently selected package.
   o The type column represents the source from which the file can be accessed. Possible values are **Attachment** (file has been physically uploaded to RayFlow) and **Link** (the package is available on a shared location and only the path to the root file is saved in RayFlow).
   o **Date** and **Uploaded by** columns denote who and when has uploaded the file
   o The list is sorted by the date, the most recent items are displayed on the top of the list.
   o Only the following file extensions are shown:
     ▪ `.exe`
     ▪ `.cmd`
     ▪ `.vbs`
     ▪ `.msi`
     ▪ `.ps1`
     ▪ `.zip` (the file will be extracted, see section *Handling multiple and supporting files within a single package*)
     ▪ `.7z` (the file will be extracted, see section *Handling multiple and supporting files within a single package*)

10. Press **OK** to open the selected file. Press **Cancel** to go back to the backstage. In order to change RayFlow task or phase, click the arrow on the left side of the list or the partially transparent area containing the list of tasks.

11. If the file has been already downloaded, a confirmation screen is shown prompting to either keep the local file and use it as a source for a tuned package, or download the package content
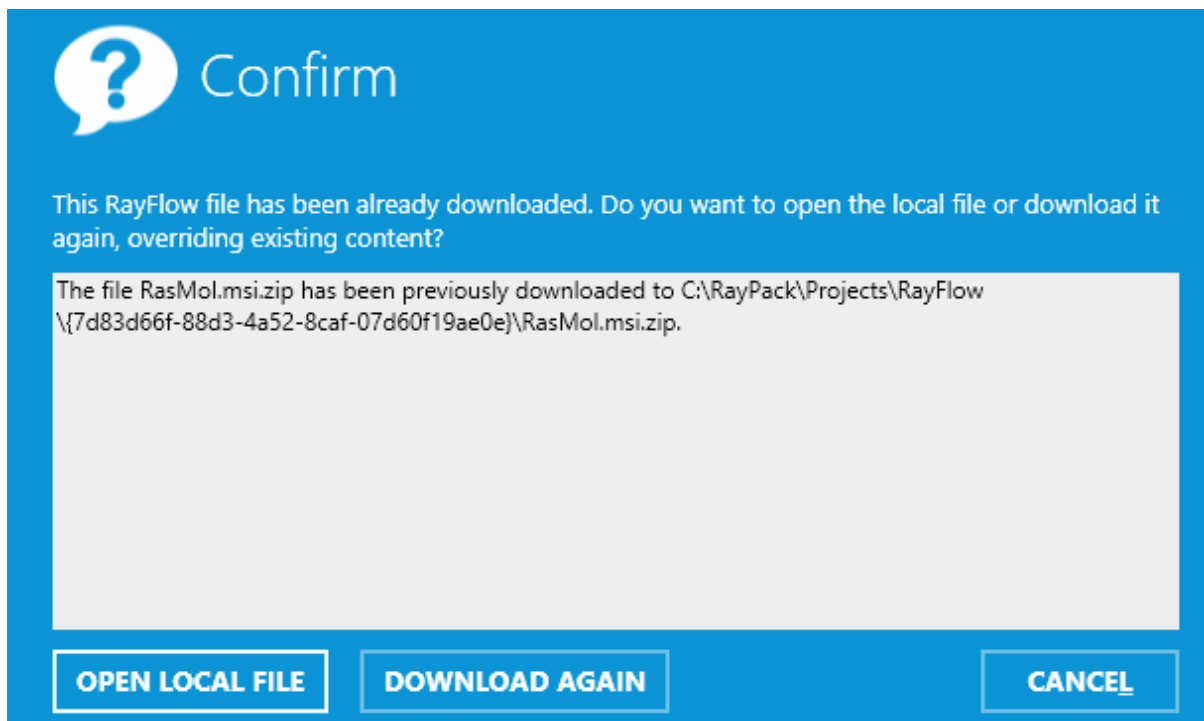
again:

- Press **OPEN LOCAL FILE** to use the already existing content
- Press **DOWNLOAD AGAIN** to download the whole content of the package. Any changes made to the local files will be overriden.

12. If the file is in a ZIP/7Z format and contains several `.msi` files, then the following selection screen may be shown:
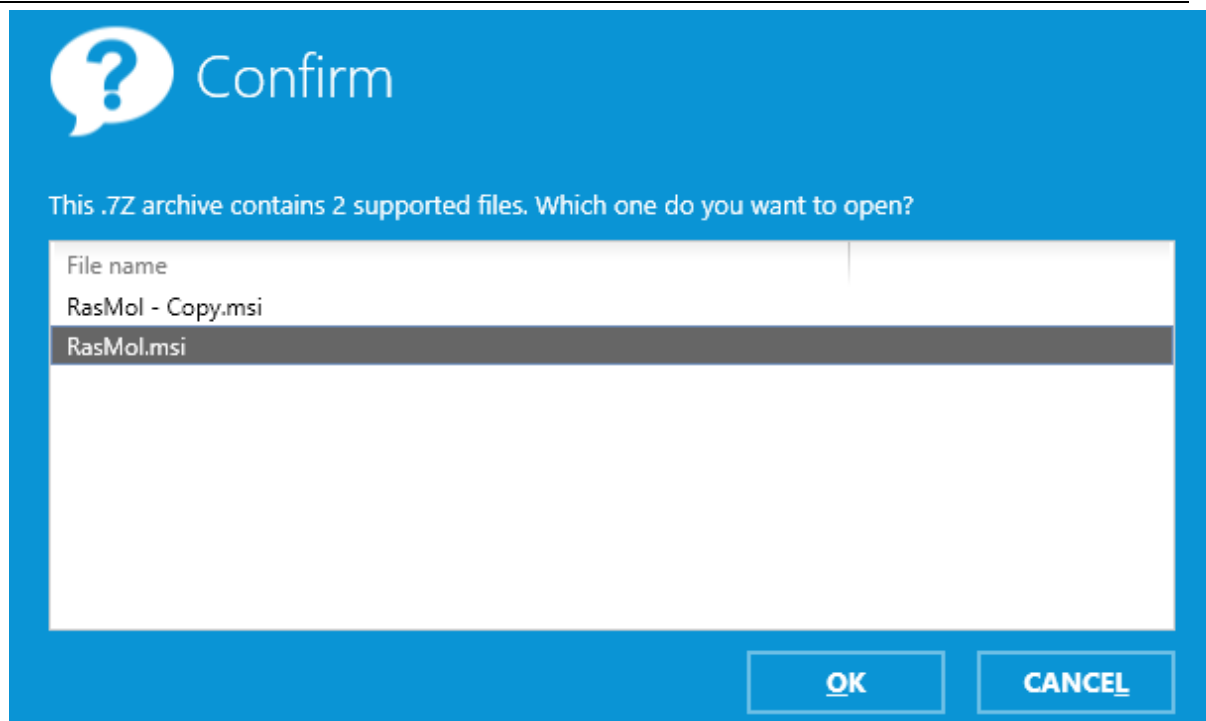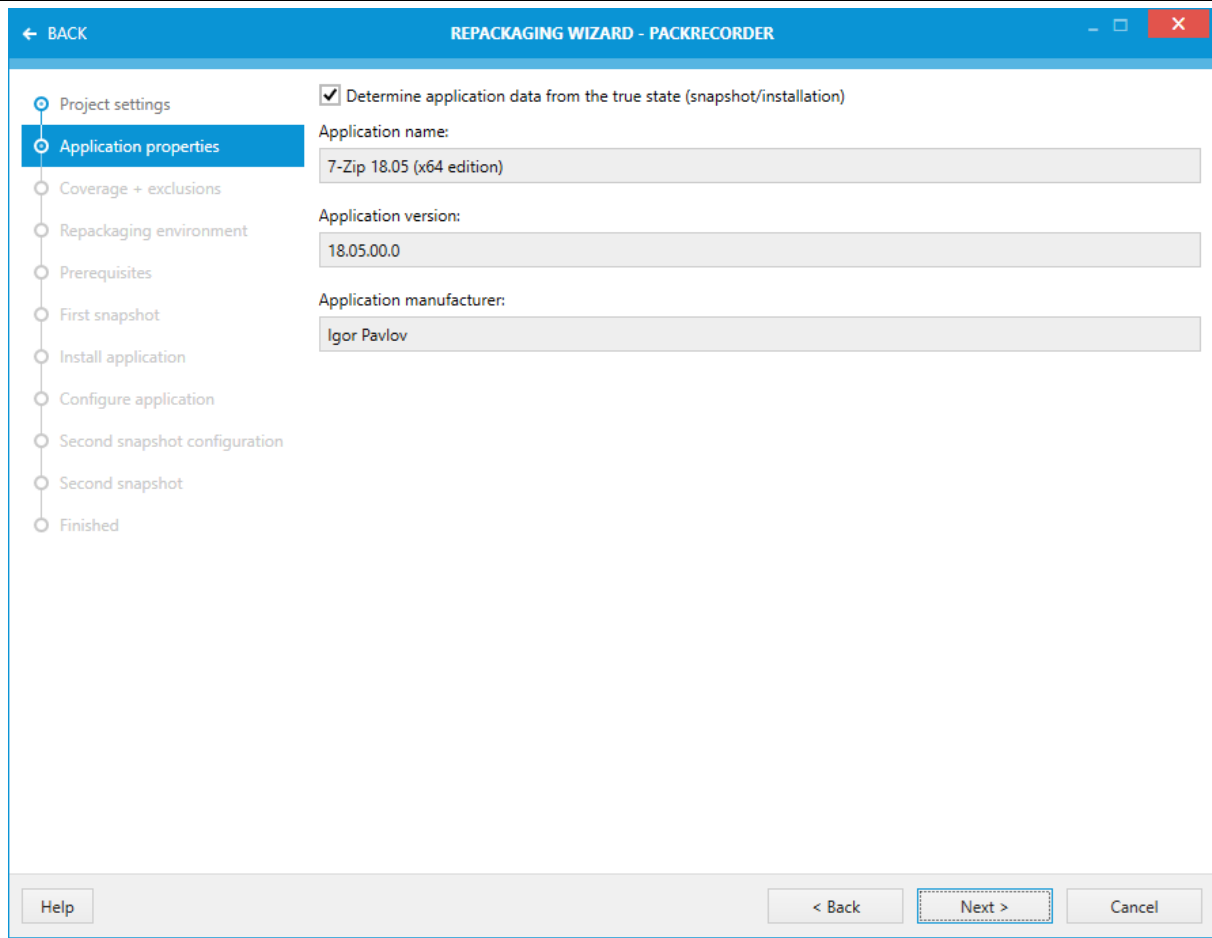
o Press **OK** to use the selected item as a legacy vendor setup to be repackaged.

> **Tip:**
> Source files downloaded from RayFlow are saved in the Projects folder. Refer to chapter Opening files from RayFlow for more information.

Application Properties

The **Application properties** screen allows you to override the way some properties like project name, version, and manufacturer are determined.

By default, these properties are determined automatically from the differences between the two snapshots (by analyzing the installed software, ARP entries, etc.). For some setups, you may want to use custom properties which differ from these automatically gathered characteristics. Another use case would be if the repackaging is done without an actual setup, but only with a manual file copy operation. Then the automatic routine is unable to determine them.

In any of these use cases, by unticking the checkbox **Determine application data from true state (snapshot / installation)** the user is able to enter these details manually, and they will be taken over by the comparison engine and put directly into the output project file.

When ready, press **Next >** to go to the scanning options.

Coverage + Exclusions

This screen shows the options that are available for the adjustment of the current wizard execution parameters. It is only available in the **EXPERT** mode, but not in the **BASIC** mode.
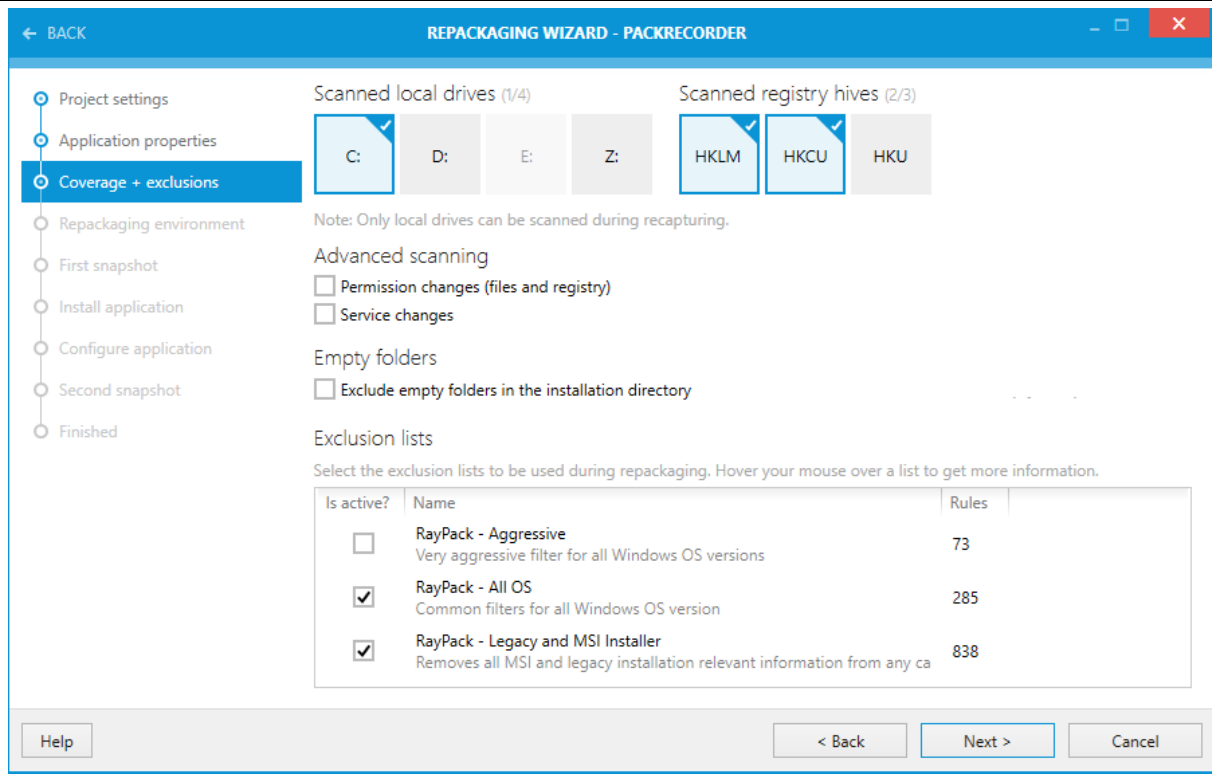
The options for exclusions and scan settings are originally set in the currently active profile, which can be edited here. If any changes are made here, **they will only be applied to the current capture process**.

Scan Settings

By expanding the **Scan** settings, changes can be made to the areas of the system (drives and registry hives) that are explicitly scanned. To select an area for scanning, simply click on the area that is to be represented in order to switch the colour from grey (inactive) to blue (active).

> **Note:**
> Only areas that are represented by blue tiles with a checkmark in the upper right corner will be scanned during the upcoming snapshot procedure.

The current setting will cause the wizard to scan on
drive C:, but ignore any activity on D:, E: and Z:

> **Note:**
> The set of available drives depends on the environment given on the currently used machine. There may be additional drives, such as shared network drives.

The settings for the **Advanced Scanning** manage the recognition of changes performed on services and permissions (on files & registry) during the application installation and configuration. Activate the respective checkbox to scan for the labelled option.

	Note:
	Changes here are only applied for the current capture process and are not permanently saved. Perpetual change can be made in the global profile settings which can be found here.

	Tip:
	The fewer system areas that are observed, the less time it will take to perform the scan procedure. Therefore, it is vital for efficient setup capture routines to make sure that only relevant system parts are reflected. However, if an application is rather unknown to the packager, running an extended capture might be prudent in order to make sure all installation outcomes are covered. It is part of the experience of a professional packager to know which application affects which system area and adjust their work procedures accordingly.

Capturing empty folders

Prior to RayPack 6.3, empty folders were not captured and had to be included manually. From version 6.3, the user can control whether the legacy behavior (ignoring empty folders) or respecting empty folders should be used when building RCP files.

Note that this setting applies to folder that are within your installation directory (so that temporary and system directories are always excluded if they are empty).

Exclusions

Expanding the Exclusions toggle is where changes can be made to the exclusion filters that are applied. The user can select an alternative directory containing exclusion filters, add or remove existing filters.
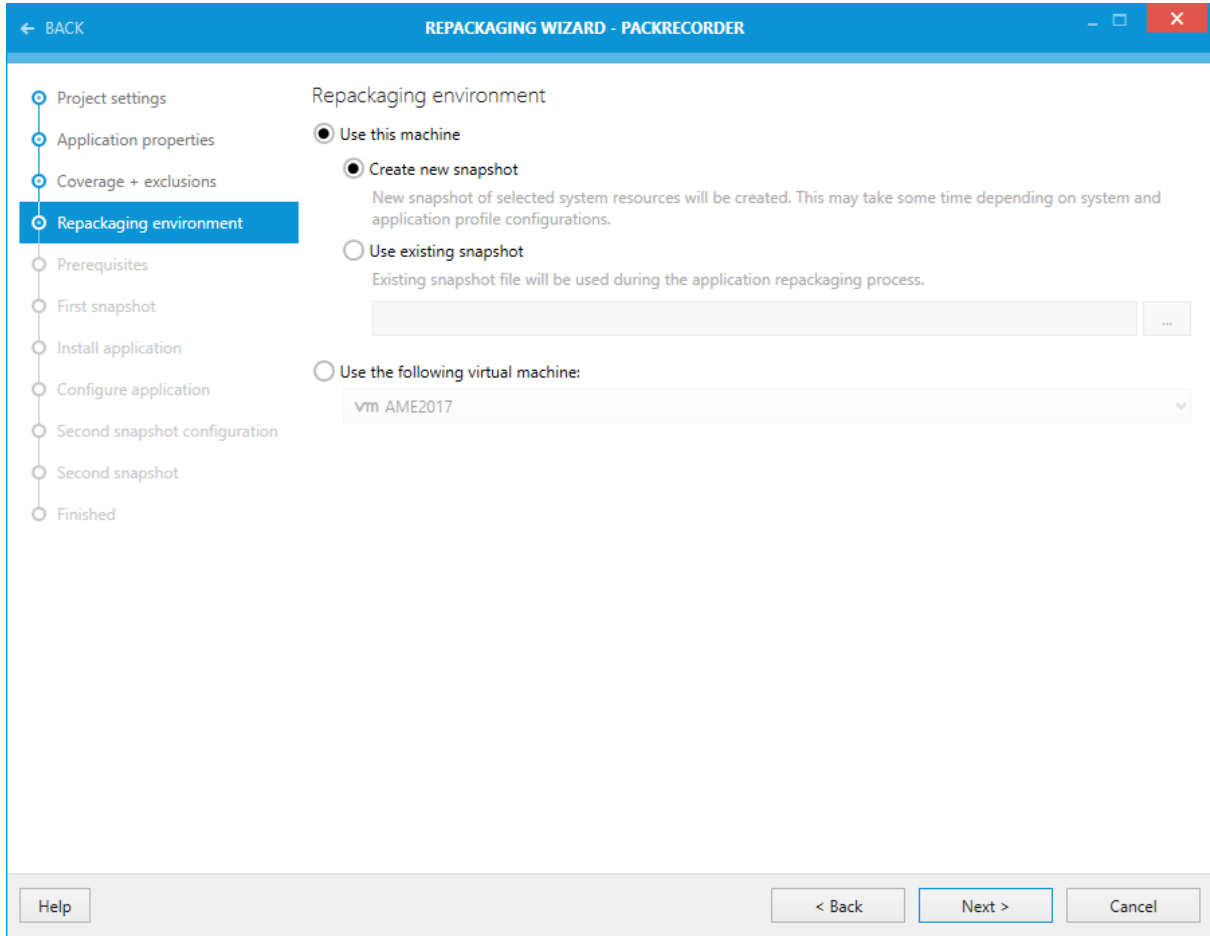
	Note:
	It is not possible to edit the filters from this user interface. This can only be done in the global profile settings *in the Settings section*.

	Tip:
	Even though RayPack comes along with a predefined set of standard exclusion lists, it is highly recommended to add custom lists according to the individual customer or project environment needs. Please keep in mind that high quality exclusion preparation repeatedly saves time for every single setup capture, since it reduces the efforts required for future project clean-up measures.

Repackaging Environment

This screen allows the selection of a previously saved snapshot as the data source for the initial system capture step. It is available in the **EXPERT** mode, but not in the **BASIC** mode.

Use this machine

This option will perform repackaging on the current machine on which RayPack is running. You should use this option only when running the Product on a virtual machine, so that any changes to be captured will not be persisted.

Create New Snapshot

When choosing this option a new snapshot will be created. Please note that this snapshot file will be stored in the location that is defined to store snapshots in the profile. The default name for the first snapshot is `snapshot0.rcs`. Any snapshots that have this name in the location set as the snapshot directory will be overwritten.

Use Existing Snapshot

If a snapshot has been saved during a previous capture process, use the browse button **[...]** to navigate to an `*.rcs` file and select it. When the snapshot file has been successfully parsed, information regarding the snapshot file, as illustrated below, is displayed.
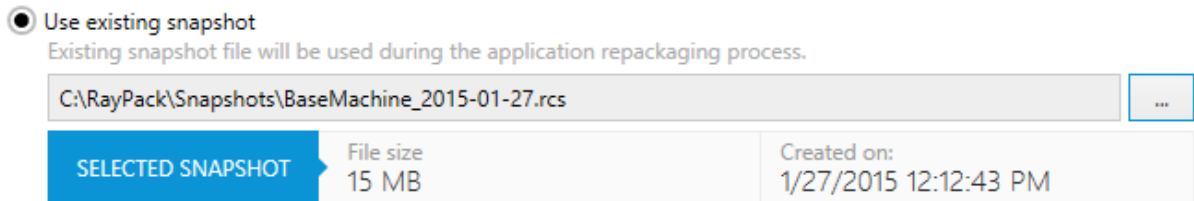
- **File size**
  The size of the snapshot file.

- **Created on**
  The date and time on which the snapshot was created.

Use the following virtual machine

This option will perform repackaging on a selected virtual machine. Available machines and their configuration can be reviewed and edited in the Settings screen. Before pressing **Next >**, make sure that the desired machine is selected in the drop-down menu.

Prerequisites (Local repackaging)

This screen shows the recommendations and hints about the state of the current machine that may affect the repackaging process.

> **Note:**
> This page is only shown when a local repackaging ("Use this machine") was selected in the previous step.
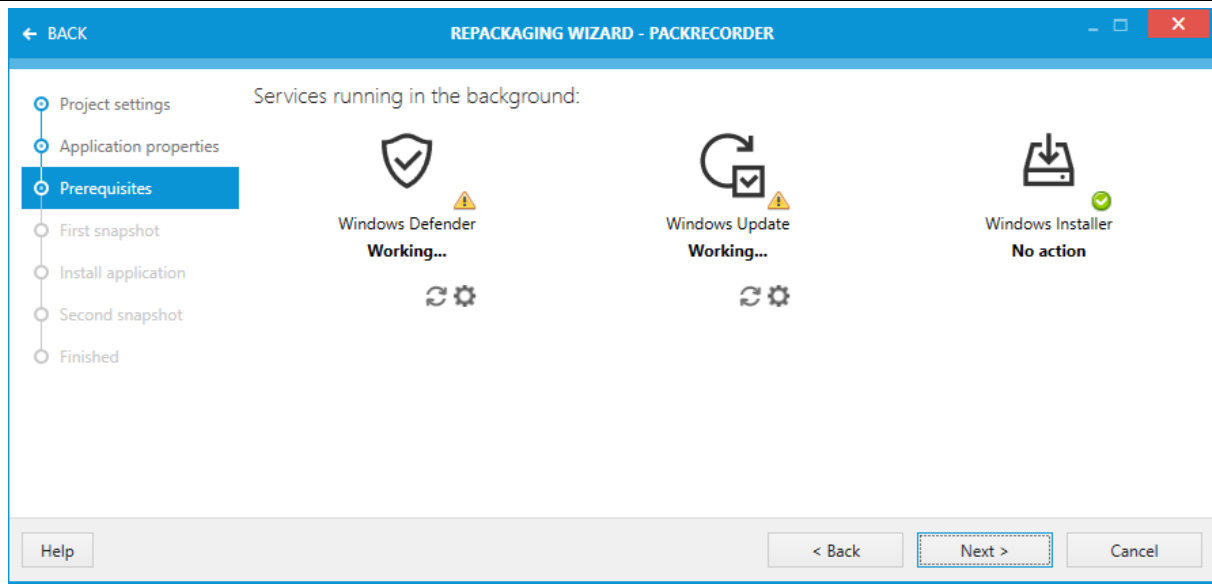
The program checks for the following conditions:

- Whether Windows Defender is enabled (for repackaging disabling Defender is recommended to reduce noise and improve the performance of the repackaging).
- Whether Windows Update is enabled (for repackaging disabling Windows Update is recommended to reduce noise).
- Whether Windows Installer is running (for repackaging the installer should not be running prior to the installation to reduce noise).

If the status says **Working…**, the service is enabled and active. The label **No action** means that the service is not running and / or disabled and no action is required. You can press the gear icon under each icon to go to the **OS Settings**, where the given option can be disabled. Press the **Refresh** button to refresh the view.

These recommendations are optional and can be ignored. When you are ready, press **Next >** to go to the snapshot settings.

First Snapshot

This screen shows the progress of the currently active first snapshot recording of the capture wizard. It is available in **BASIC** and **EXPERT** mode.

It will not appear if a snapshot file (`*.rcs`) has already been selected as the data source for the initial snapshot. Please refer to the previous topic to find out how to re-use RayPack system snapshots.
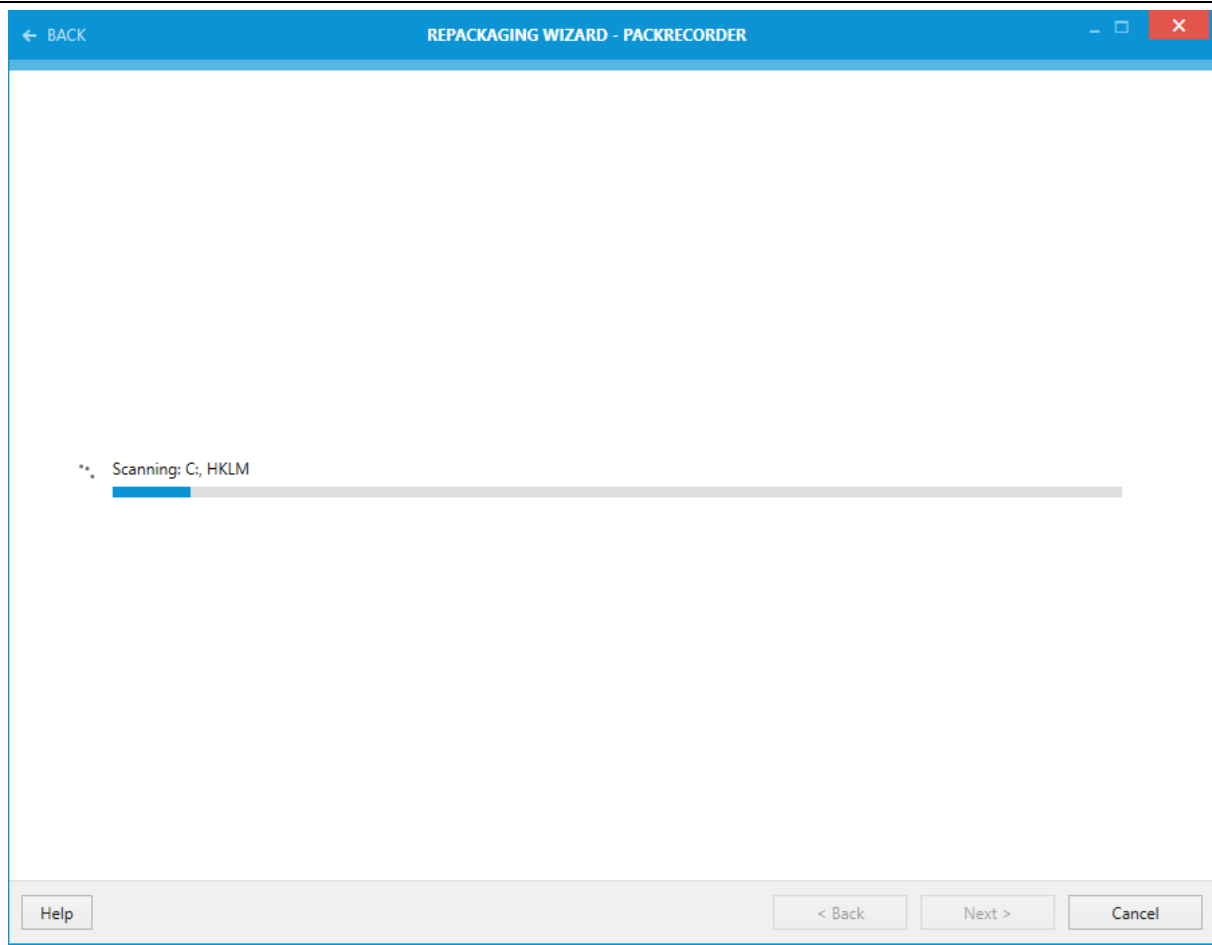
The first (initial) snapshot is taken. Depending on the settings that have been made for the snapshot process, this may take some time. If during the snapshot process the need to abort arises, you can click the **Cancel** button, but be aware that any information gathered during the snapshot process will be reset and lost (unless you are using a previously saved snapshot).
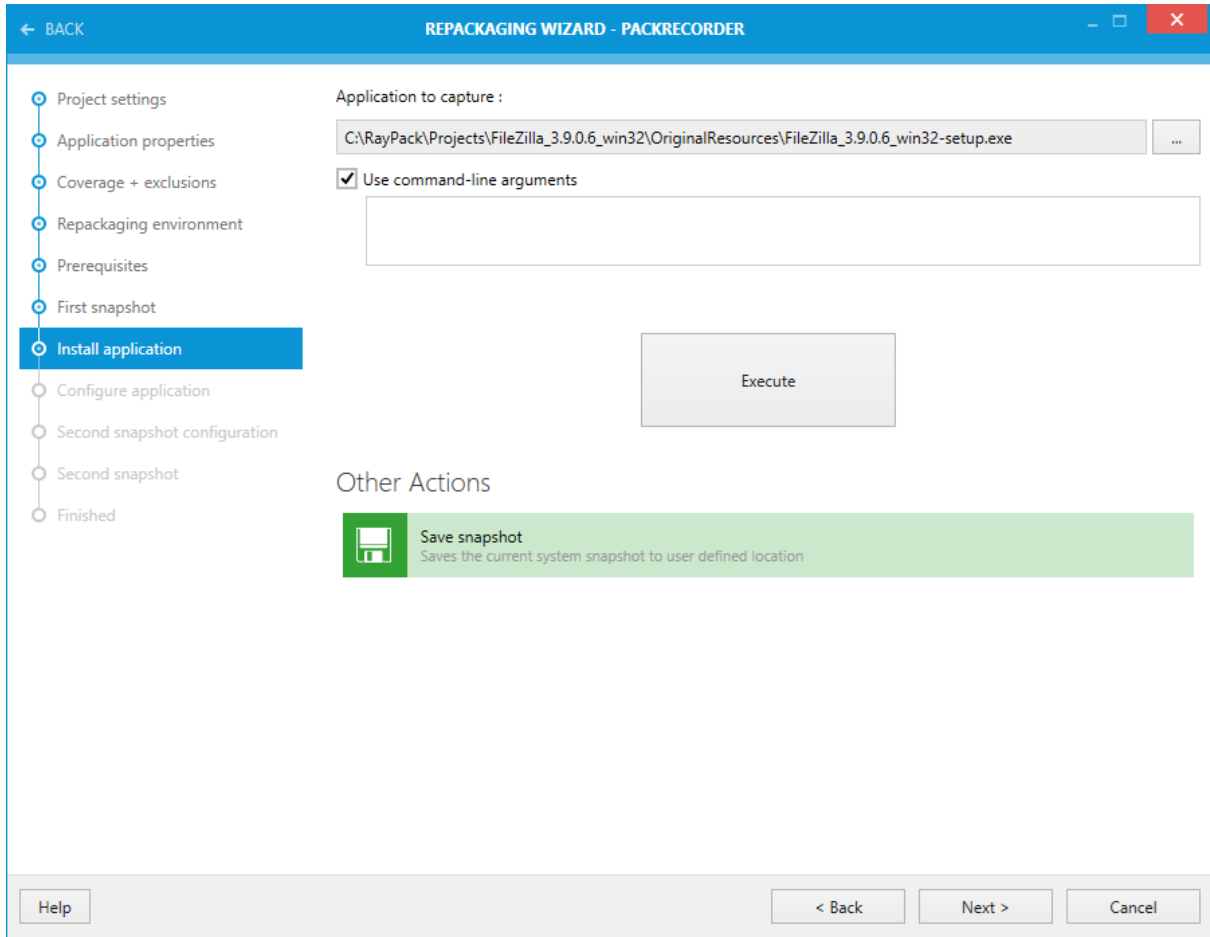
Once the first snapshot has been completed, the wizard automatically moves to the next step.

Install Application (Local repackaging)

This screen shows the options and controls available for application installation. It is available in **BASIC** and **EXPERT** mode.

> **Note:**
> This page is only shown when a local repackaging ("Use this machine") was selected in the previous step.

Application to Capture:

This field may contain the name of the application to capture (usually the installation executable). If the field has been set to The main installer on the project settings page, then this entry is present. If this field is left empty, then any actions that are carried out will be recorded (such as copying files, manipulating registry items, and the like). The **Execute** button has no effect if this field is empty.

Use Command Line Arguments

This checkbox allows the user to set command line arguments that are to be used in conjunction with the Application to capture field. When this check box is selected, any command line arguments in the appearing text input field can be entered. To use the command line arguments, be **sure** that the **Application to capture** field contains a valid file that can be executed, and be **sure** to click the **Execute** button.

EXECUTE

On clicking the **Execute** button, the values that have been entered in the above fields are used to execute an application (usually the installation) passing any command line arguments defined. During this process, the application required for use should be installed.

> **Note:**
> If you specified an application to be executed and pressed the **Next >** button without clicking the **Execute** button beforehand, the setup will be started anyway. This is a one-time behavior by design.

Other Actions (EXPERT Mode Only)

This option enables the saving of the current snapshot to a location of choice. This is especially useful if the snapshot is to be used later.

> **Tip:**
> If the snapshot is saved before installing the application, then a snapshot of a clean machine is readily available and can be used again for any further capture projects.

> **Note:**
> If there is no installation routine to install and configure the application to be packaged (i.e. manual copy to target system), then there is no need to enter anything into the **Application to capture** field. Simply leave this empty and carry out the manual copy / paste installation, create any shortcuts, etc. Once the steps that comprise your "installation" have been completed, simply click the **Next >** button.

Recognition of Vendor MSI Installations

PackRecorder automatically detects vendor MSI installations. Many setups are prepackaged to a single container `.exe` file which acts merely a a wrapper and executes the actual Windows Installer database extracted to a temporary location. If this happens, repackaging to MSI format is usually a bad choice because it breaks the compatibility and upgrade possibilities of older vendor packages. In most cases, the recommended approach is to create a Windows Installer transform.

While staying on the **Install Application** screen, once an MSI installation is detected an information is shown in real-time underneath the **Execute** button:
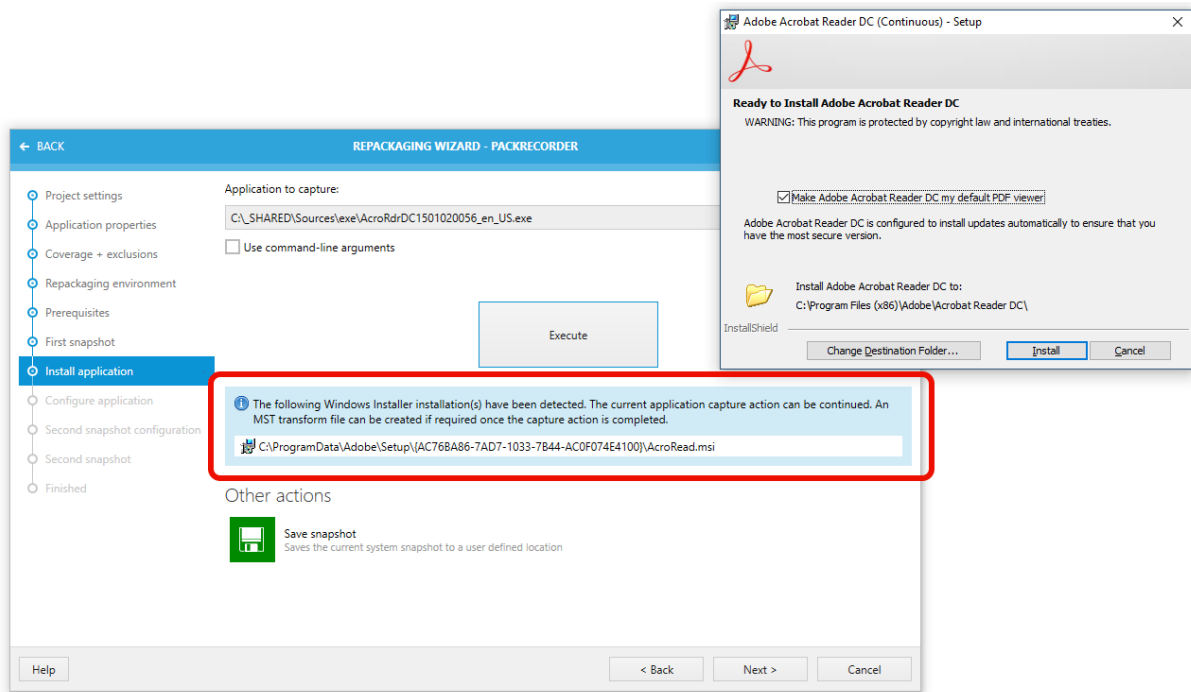
It is not necessary to take any extra actions, at this point the message is only for information. However, once the RCP project is created, the original MSI will be copied to the project folder for a further reference and an option to create an MST transform file against the vendor MSI will be available.

Configure Application (Local repackaging)

This screen allows to configure the recently installed application before the second snapshot is prepared. It is available in the **EXPERT** mode, but not in the **BASIC** mode.

📎	**Note:** This page is only shown when a local repackaging ("Use this machine") was selected in the previous step.

On this page the application should be configured according to its use and requirements. The application should be started to ensure that any settings that are automatically created during the initial start-up of the application are captured. If required, reboot the machine. The PackRecorder wizard will resume at this step as soon as the machine has restarted. The removal of any unwanted files, shortcuts, and/or registry entries can be performed. However, adding files, shortcuts, and/or registry entries is not only possible, but recommended and in some cases, necessary.

Any changes made on the repackaging machine whilst this wizard step is active will be considered during the delta recognition between the first and second snapshot!

Tip:
It is recommended to remove any temporary files and or registry keys here, as this saves time when editing the resulting capture project.

Configure Installer (Repackaging on VM)

This screen allows to run a setup with specific arguments on the target virtual machine.

Note:
This page is only shown when a local repackaging ("Use the following virtual machine") was selected in the previous step.

On this page the application should be configured according to its use and requirements. The application should be started to ensure that any settings that are automatically created during the initial start-up of the application are captured. If required, reboot the machine. The PackRecorder wizard will resume at this step as soon as the machine has restarted. The removal of any unwanted files, shortcuts, and/or registry entries can be performed. However, adding files, shortcuts, and/or registry entries is not only possible, but recommended and in some cases, necessary.

Any changes made on the repackaging machine whilst this wizard step is active will be considered during the delta recognition between the first and second snapshot!

> **Tip:**
> It is recommended to remove any temporary files and or registry keys here, as this saves time when editing the resulting capture project.

Second Snapshot Configuration (Local repackaging)

> **Note:**
> This page is only shown when a local repackaging ("Use this machine") was selected in the previous step.

This screen allows to select a previously already saved snapshot as the data source for the second, comparative system capture step. It is available in the EXPERT mode, but not in the BASIC mode.
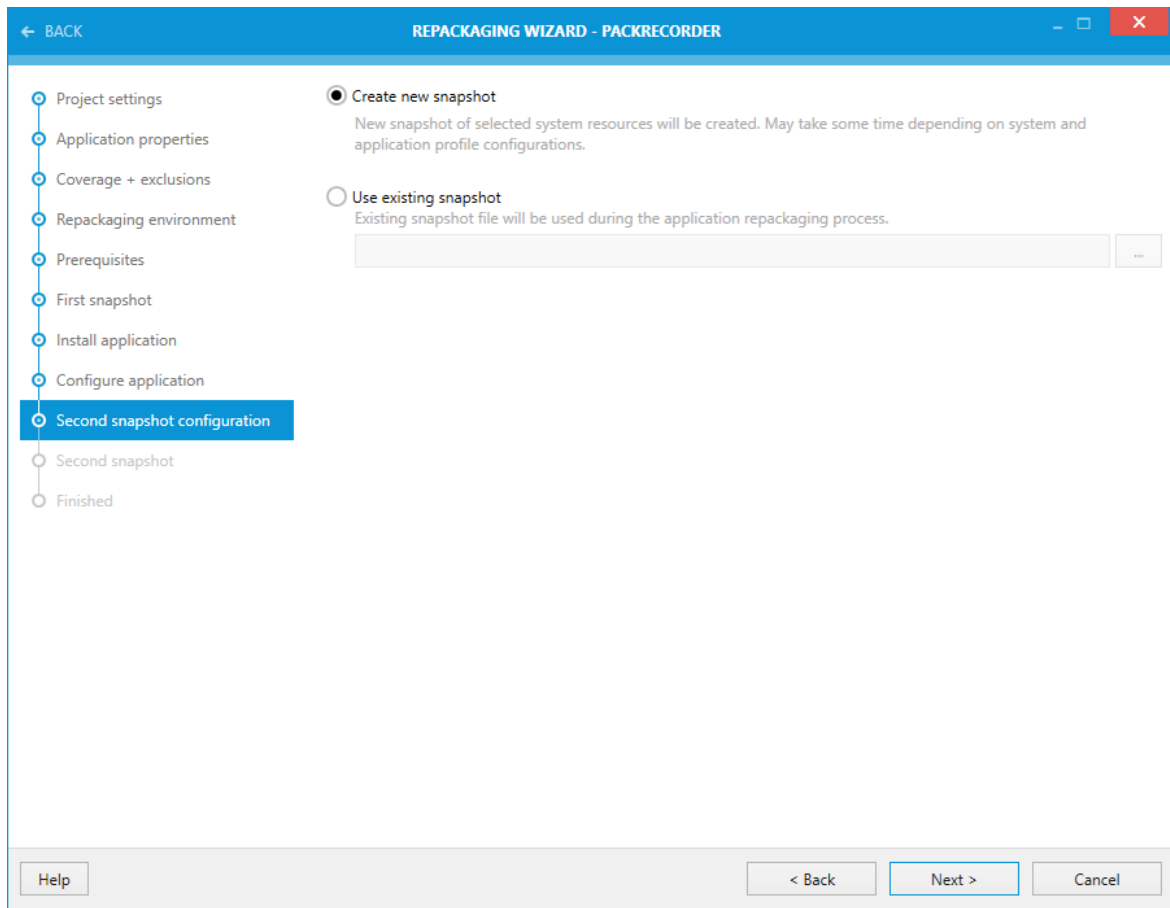
Create New Snapshot

By selecting this option, a new snapshot will be created. Please note that this snapshot file will be stored in the location that is defined to store snapshots in the profile. The default name for the first snapshot is `snapshot1.rcs`. Any snapshots that have this name in the location set as the snapshot directory will be overwritten.

Use Existing Snapshot

If the decision was made to save the snapshot during a previous capture process, use the **BROWSE** button to navigate to an `*.rcs` file and select it. When the snapshot file has been successfully parsed, information regarding the snapshot file is shown.

- **File size**
  The size of the snapshot file.

- **Created on**
  The date and time on which the snapshot was created.

Choosing the NEXT button will initiate the second snapshot **unless the choice was made to use an existing snapshot**. In this case, on the Second Snapshot page navigate to the snapshot that was previously taken. The chosen snapshot will be compared **without taking a second snapshot**.

Second Snapshot

This screen shows the progress of the currently active second snapshot recording of the capture wizard. It is available in **BASIC** and **EXPERT** mode. If the second snapshot configuration was used to pick a previously saved RCS file as the data source for the second snapshot, there will be no further recording in this step. Please refer to the previous topic to find out how to reuse RayPack system snapshots.

However, no matter if the first, second, or both snapshot files have been selected from already existing `*.rcs` files, or recorded anew, this wizard step is also used to determine the delta between the two system states represented by the snapshots provided.
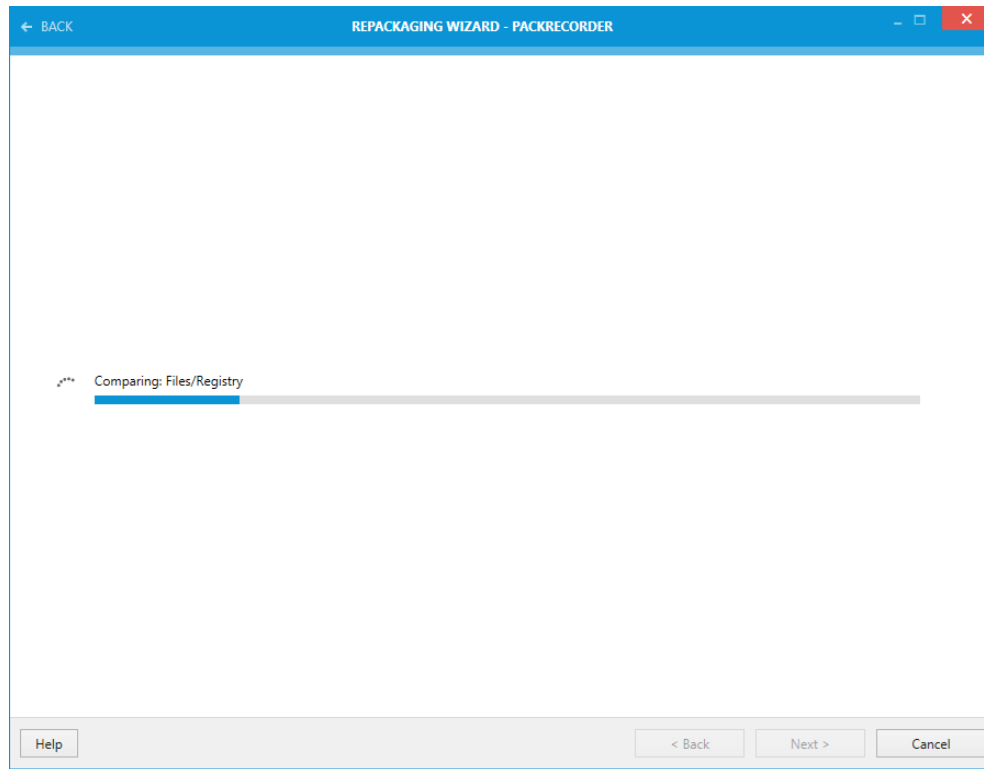
Once the delta calculation is complete, the next page is shown. If a rather complex application has been captured or the work is being performed on a machine with very low performance, this process may take some time.

As soon as the differences between the files are ready to be stored as RCP (RayPack Capture Project) file, the **Finished** page of the wizard is displayed.

Finished
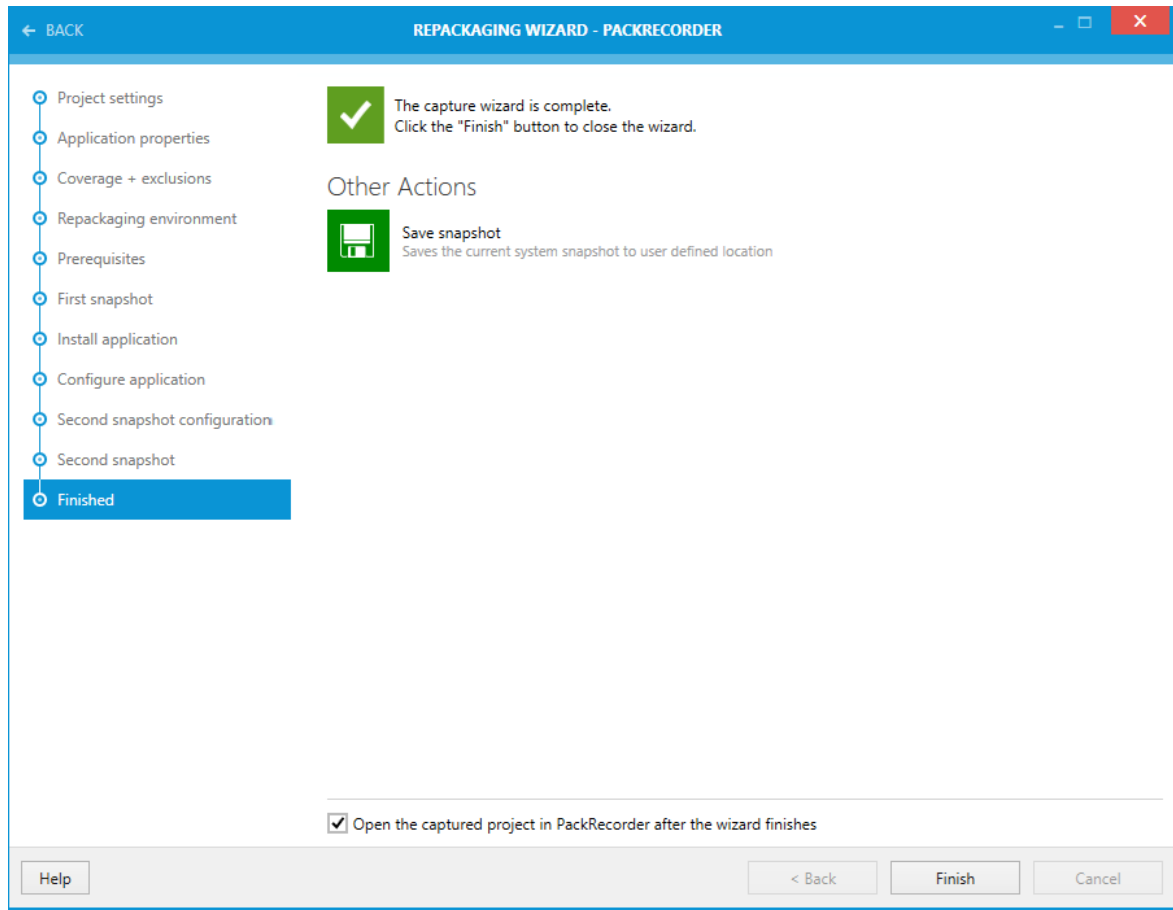
Once the second snapshot has completed, the project and any data relating to it have been saved in the project directory previously defined. Upon clicking the **Finish** button the user is returned to the Home Screen, from where one may open the capture project that has just been created.

The project will be opened automatically if the checkbox is activated on **Open the captured project in PackRecorder after the wizard finishes**. Choosing **< Back** will return to the Second Snapshot page, where the second snapshot process will be repeated.

Other Actions (EXPERT Mode Only)

This option permits the saving of the current snapshot to a location of choice. This is useful if the snapshot will be used in the future.

> **Tip:**
> If the snapshot is saved after installing and configuring the application, then the snapshot of a machine as it has to be after the later target package (e. g. an MSI) has been installed. This can be used to perform quality control checks later when your package has been built.

Editing the Captured Project

The PackRecorder Editor is the tool that allows editing of the information obtained after executing the PackRecorder Wizard (described here). The editor can be started by opening a RayPack Capture Project (`*.rcp`) file via the Home Screen or if RayPack has been installed locally, by double-clicking any RCP file. The PackRecorder editor allows editing of the various files, folders, shortcuts, registry entries and package information before generating the required package format. This section describes in detail the various options and parameters that can be edited and how to generate a package.

The editor interface is divided into three main sections:

- **The Main Toolbar** at the top of the view
  The buttons and tabs provided by the Main Toolbar allow swift access to core functionality of RayPack, such as opening the settings area, or building packages from the currently opened project. Please refer to the Home Screen topic for further details.

- **The tree-view** on the left hand side of the window
  This is the internal editor view navigation. Each content type of RCP files that may be manipulated is represented by one of the items in this navigation tree. In order to maintain clear structures, there is a Your project overview present at the topmost position of the tree. There are additional group items to provide access to views that are closely related to each other. Simply click on one of the items to load the information or manipulation interface into the details pane at the right-hand side of the application window.

- **The information view** (also called details pane) on the right-hand side
  This area contains information and allows the editing of the properties shown. Its contents are dependent on the entry selected from the tree-view navigation column on the left hand side of the PackRecorder Editor. Whenever a project is opened in the editor, the first page that is displayed within the details pane is the Your project overview.
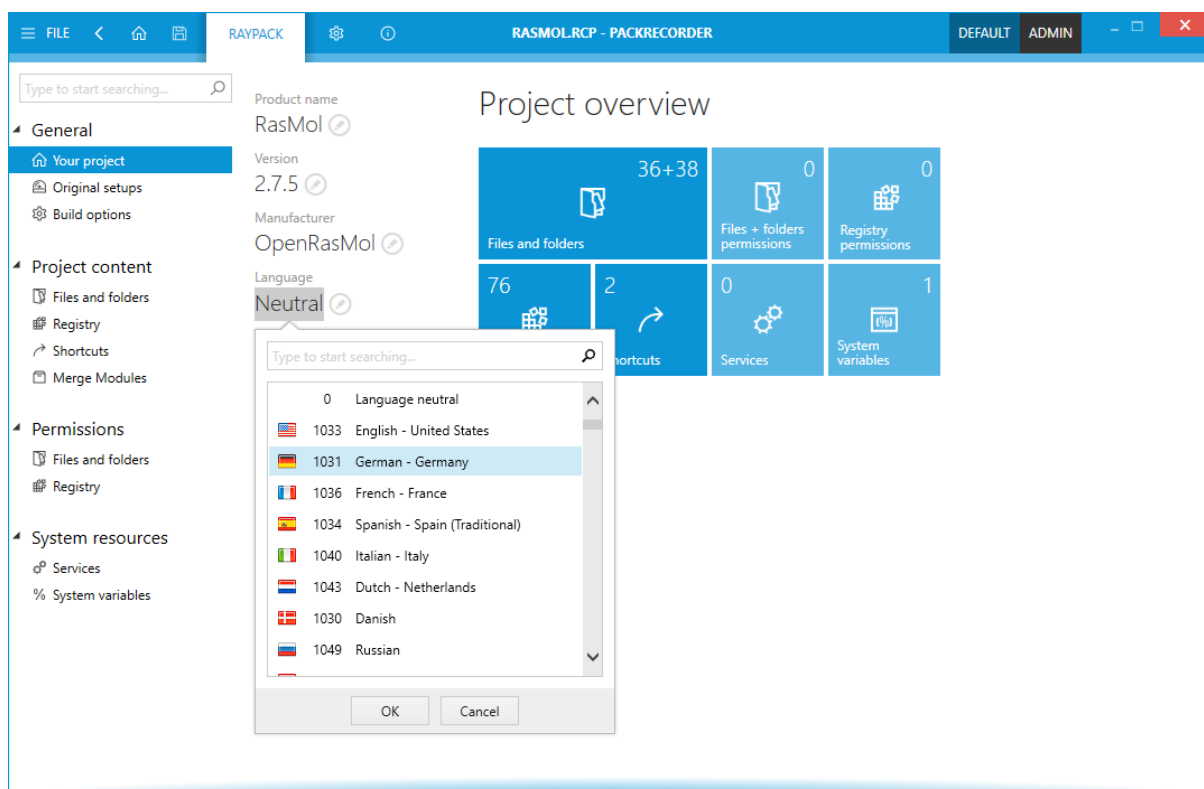
Read on to get details on the specific views of the PackRecorder Editor.

General

This view contains interfaces to manage the project and its build options.
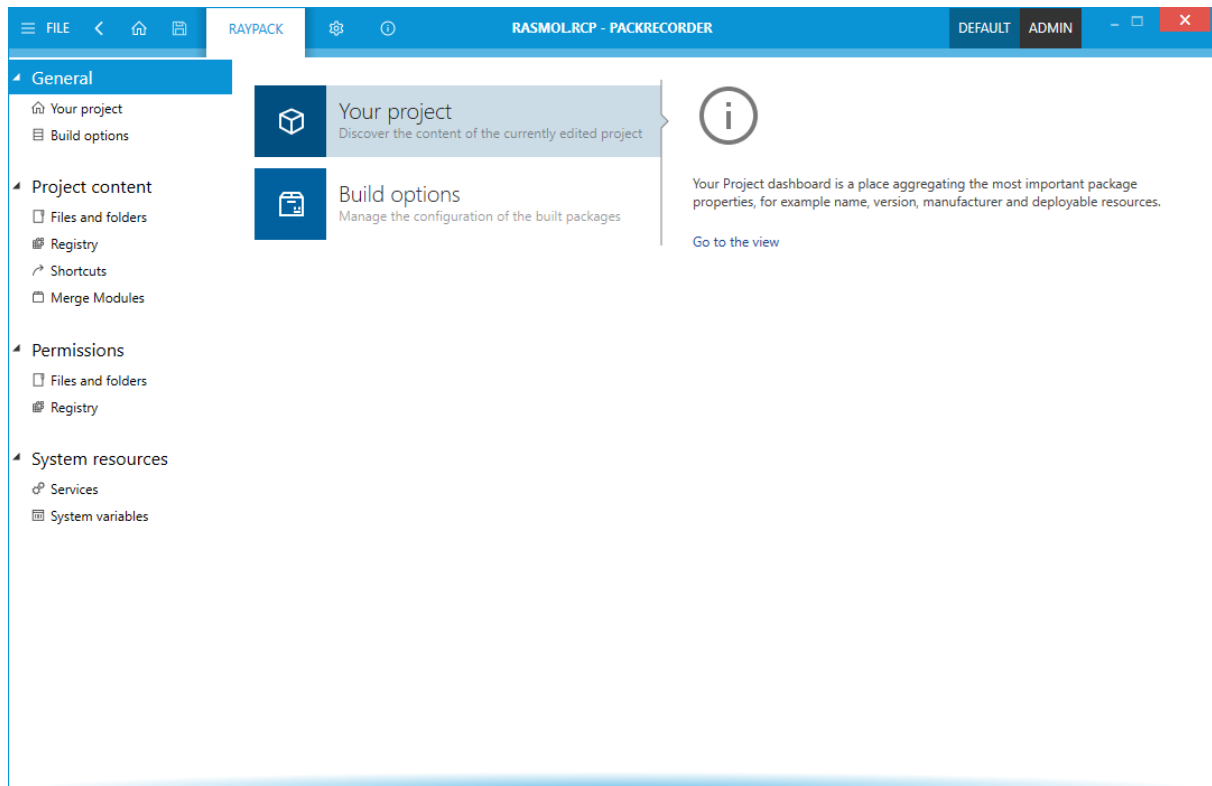
Click on one of the info sections to directly jump to the specific editor view.

Your Project

This view shows the various properties and fields that are relevant to the project. Some of the information contained here is used when building a target project/package.
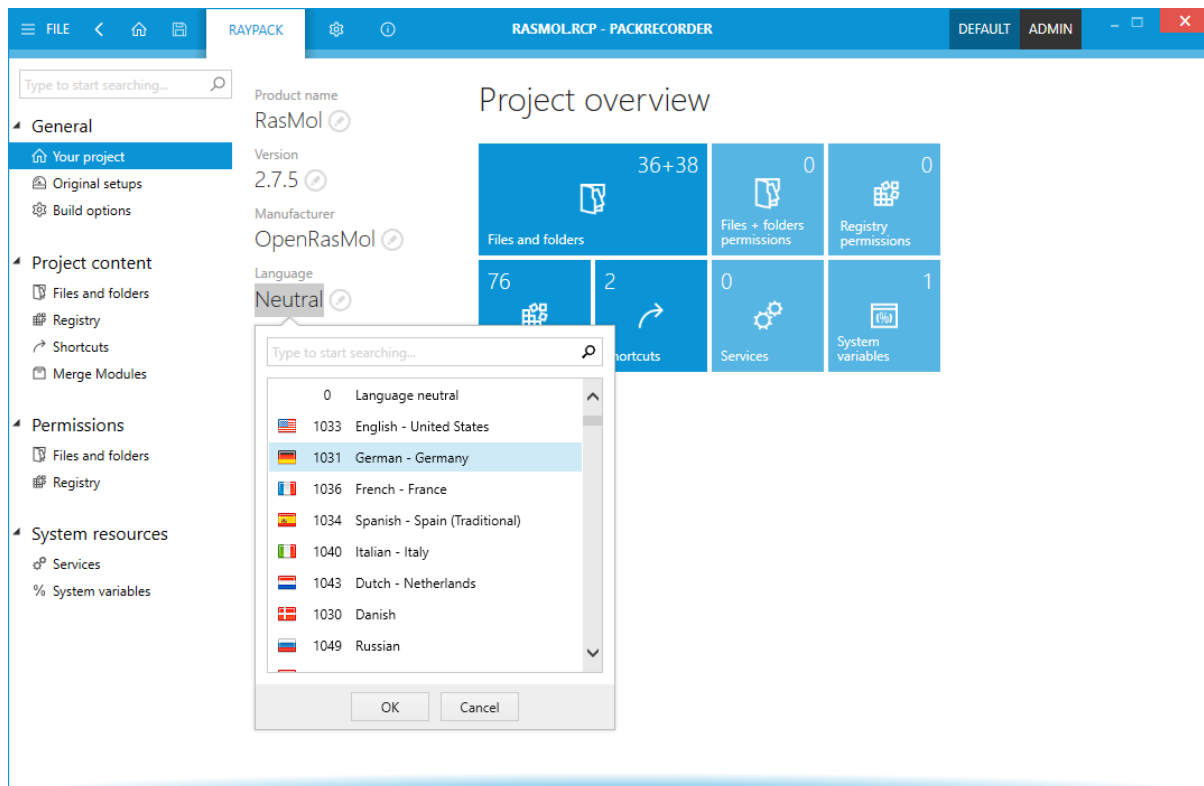
Each tile displayed within the **Project overview** symbolizes a specific package content type, such as files, folders, registry, etc. Clicking a tile directly opens the management view required to manipulate the project values regarding the content type.

The tiles additionally indicate the number of objects currently stored within the project file for a specific type of content.

> **Note:**
> The information shown in the left section, is retrieved from the captured information where possible. If this is not possible, then the field(s) are intentionally left empty.

Depending on the type of package/project generated from the PackRecorder Editor, some or all of the information shown in this view will be used in the resulting package/project. This is heavily dependent on the type of package/project generated.

Any of the properties listed below may be edited by either
- double-clicking on the current value shown below the property label
- left-clicking on the edit icon at the right-hand side of the current content

Once editing has been triggered, the direct value editor interface is activated. Please refer to the Common Dialogs section for details on how to use this editor interface control type.

Product Name

This is the name of the application. This has either been retrieved from the captured application or is based on the name of the project.

Product Version

This is the version of the application. Please note, that this should be a valid version and **must not** consist of any non-numerical characters. Where possible this information has been obtained from the captured application. If this has not been successful, then this field may be empty.

Manufacturer

This is the application vendor. Where possible this information has been obtained from the captured application. If this has not been successful, then this field may be empty.

Language

This is the language of the package, not that of the application. Where possible this information has been obtained from the captured application. If this has not been successful, then this field may be empty.

Each package may either be neutral towards the language settings or be set to a specific language. Please expand the direct value editor and select a language setting option from the given list.

Original Setups

This section contains information about the setups which are used during the repackaging. It automatically collects the following entries:

- The main setup that has been configured directly in the PackRecorder Capture Wizard
- Background MSI installations that were fired during the capturing

It is possible to switch between these setups using the drop down in the upper part of the screen. The following is an example how it may look for a captured MSI installation:
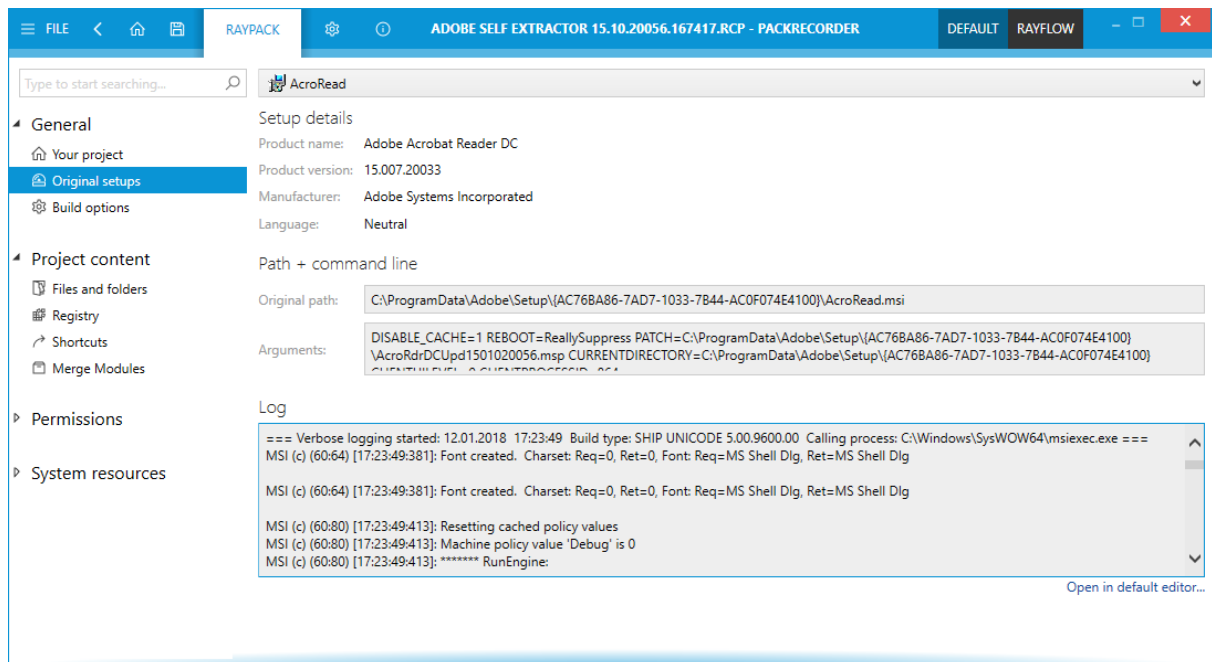
The view contains the following information:

- **Product name**
  The name of the product, taken from property `ProductName` from the vendor MSI.
- **Product version**
  The version of the product, taken from property `ProductVersion` from the vendor MSI.
- **Manufacturer**
  The name of the manufacturer, taken from property `Manufacturer` from the vendor MSI.
- **Product language**
  The language of the product, taken from property `ProductLanguage` from the vendor MSI.
- **Original path**
  This is the original path from which the installation has been started. In most of cases the MSI is started from a temporary folder and removed after the execution, so it may be that the file listed here is already missing at the time the RCP is being edited.
- **Arguments**
  These are the original command line arguments that were used to trigger the installation. This

information can be used to find out which properties, feature states etc. have been requested.

- **Log**
This is the full content of a log file. It is also possible to browse the log file by going to a folder where the `.rcp` resides, and then to the subfolder `<ProjectName>_setup`.

Build Options

This view shows the configuration build settings for creating specific package types based on the information contained within the capture project.
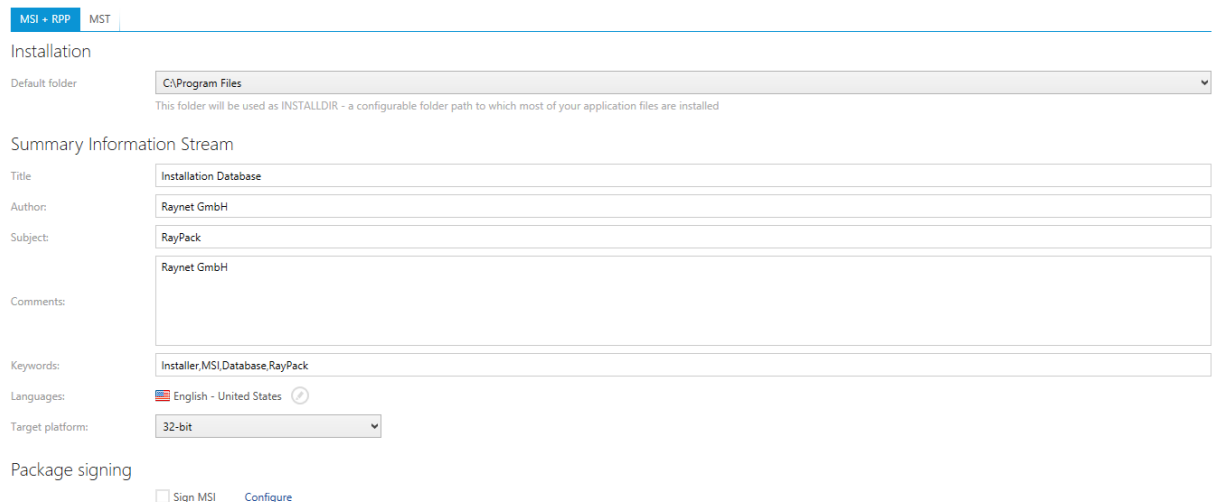
> **Note:**
> Once the build options are defined, users have to click on **FILE** and select **Build** from the options menu to trigger the actual package / project build procedure.

MSI + RPP

This tab contains various meta-data information used when the MSI-based project (including internal `*.rpp` format) are generated.

The screen is divided into tabs. Each tab contains a group of settings used by the specific project exporter.

Default Folder

This is the default installation directory of the application contained within the resulting project / package. Using the drop-down box, the user may choose which directory is the default installation directory. Please note that only the directories that are present in the RCP project can be selected.

Title

A description of this file as an installation package. The description usually includes the phrase "Installation Database", but this is not a requirement.

Author

This is where the name of the manufacturer of this product goes (for example, Adobe, 7-Zip.org etc.).

Subject

This is used for the name of the product installed by this package. This is usually the same name as in the Application name value set in the general settings.

Comments

Usually contains the phrase: "This installer database contains the logic and data required to install Product name". In this example, the creation date and time is used.

Keywords

A list of keywords that may be used by file browsers to do keyword searches for a file (search service and indexing on windows operating systems). The keywords usually include "Installer", as well as, product-specific keywords. In this example, the project name has been used.

Target Platform

The information set in the Target Platform value is used in conjunction with the Languages value to create the Template Summary property. Possible values are:

- **Inherited from template** - The target platform is inherited from the template and not set manually.
- **32-bit** - The package is for 32-bit not 64-bit platforms
- **64-bit running on x64 platform** - The package is only for 64bit platforms (AMD64, **not** Intel Itanium architecture)
- **64-bit running on Intel64 platform** - The project is for 64bit Itanium platforms (**not** AMD64 platforms)

Languages

The Language value sets the languages that are compatible with the resulting MSI project. If this field is left empty, then 0 is automatically assumed (in this case the language 0 means that all languages are supported). This field can contain multiple Language ID's, each separated by a semi-colon such as 1031;1033.

Sign MSI

When an MSI file is built from the current RCP project, it is possible to sign it. To do so, activate

the checkbox Sign MSI. The settings values for package signing (e. g. certificate and password) from the current profile are used to perform the signing.

Note:
Package signing only applies to MSI target packages. If an RPP (or MST) is build, signing does not take effect.

Be aware:
The Sign MSI checkbox is displayed read-only if there are no signing configuration settings available. To enter or adjust the package signing settings, click on the **Configure** link. The required settings editor dialog is displayed in an additional window, ready for immediate interaction.
As soon as the sign settings are saved and the additional application dialog is closed, the Sign MSI checkbox becomes available for activation.

MST

This tab contains settings required for MST generation from capture projects.

MST related build options

Vendor MSI

The sources stored within the `.rcp` file will be compared to the base MSI defined here. The MSI is used to determine the delta for the transform generation.

Transform Options

The following list of project content control options may be used to author transform database table contents:

- **Transform features**
- **Transform properties**

- **Add files**
- **Update files**
- **Add registry entries**
- **Update registry entries**

Activate the checkboxes to include the respective content type into the created transform file.

Output Folders

Depending on the number and types of build projects / packages you have selected, the build process may take some time. Once the build process has completed, the selected projects / packages will be stored in the project sub-directory as described below.

The following sub-directories are reserved / used for the built packages / projects. The following examples assume that the default project directory has been set to `C:\_work_\Projects`, and the project name is `7-Zip_9.20_MUL_1.0.0`

_RPP

This is the directory (`C:\_work_\Projects\7-Zip_9.20_MUL_1.0.0\_RPP`) that will be created / used when a packaging project is built from the capture project (RCP). It contains all of the required files to build a package based on the RPP project format.

_MSI

This is the directory (`C:\_work_\Projects\7-Zip_9.20_MUL_1.0.0\_MSI`) that will be created / used when an MSI package is built from the capture project (RCP). It contains the MSI file (the name is the same as the project, for example, `7-Zip_9.20_MUL_1.0.0.msi`) and depending on the MSI settings used, optionally any support files.

_MST

This is the directory (`C:\_work_\Projects\7-Zip_9.20_MUL_1.0.0\_MST`) that will be created / used when an MSI package is built from the capture project (RCP). It contains the MST file (the name is the same as the project, for example, `7-Zip_9.20_MUL_1.0.0.mst`) and depending on the MST settings used, optionally any support files.

_APPV

This is the directory (`C:\_work_\Projects\7-Zip_9.20_MUL_1.0.0\_APPV`) that will be created / used when an MSI package is built from the capture project (RCP). It contains the virtual package file (the name is the same as the project, for example, `7-Zip_9.20_MUL_1.0.0.appv`, and optionally any support files.

_SWV

This is the directory (`C:\_work_\Projects\7-Zip_9.20_MUL_1.0.0\_SWV`) that will be created / used when an SWV package is built from the capture project (RCP).

_MSIX

This is the directory (`C:\_work_\Projects\7-Zip_9.20_MUL_1.0.0\_MSIX`) that will be created / used when an MSIX package is built from the capture project (RCP).

> ⚠️ **Warning:**
> Any existing project or data files will be overwritten if the project is built a second time.

The user may now continue to open the project / package with the PackDesigner and continue editing and modifying the project / package. If the package is complete, then it is ready to test!

Project Content

This view contains interfaces to manage captured file, registry, and shortcut resources. They represent the actual application content, which will be noticeable for the end-user once the built package is deployed to the target machine during the installation.
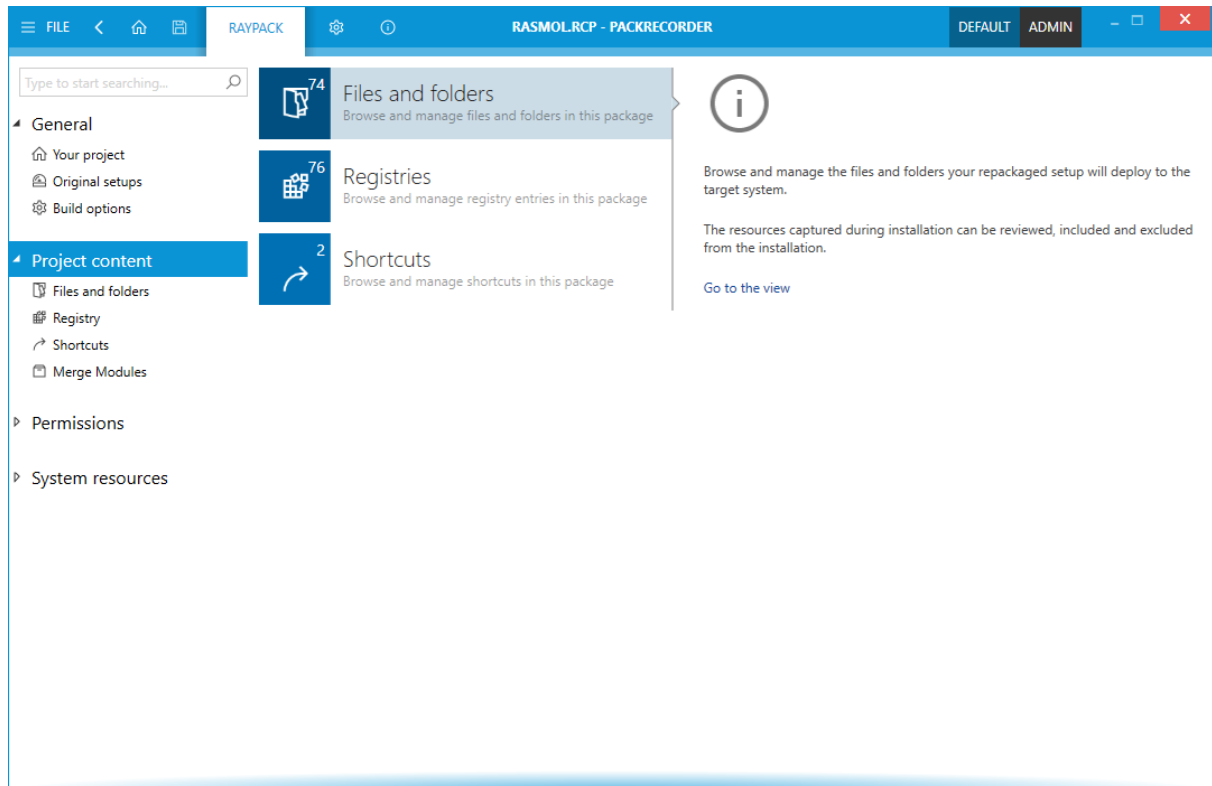
Click on one of the info sections to directly jump to the specific editor view.

Files and Folders

This view shows the files and folders of the capture project. It allows the addition of files, folders, and drives to the target package built from the project, or excludes files, folders, or drives from the target package built of the project. It also permits the renaming of files and folders.

The main view consists of two sections: The directory tree on the left and the file details on the right.
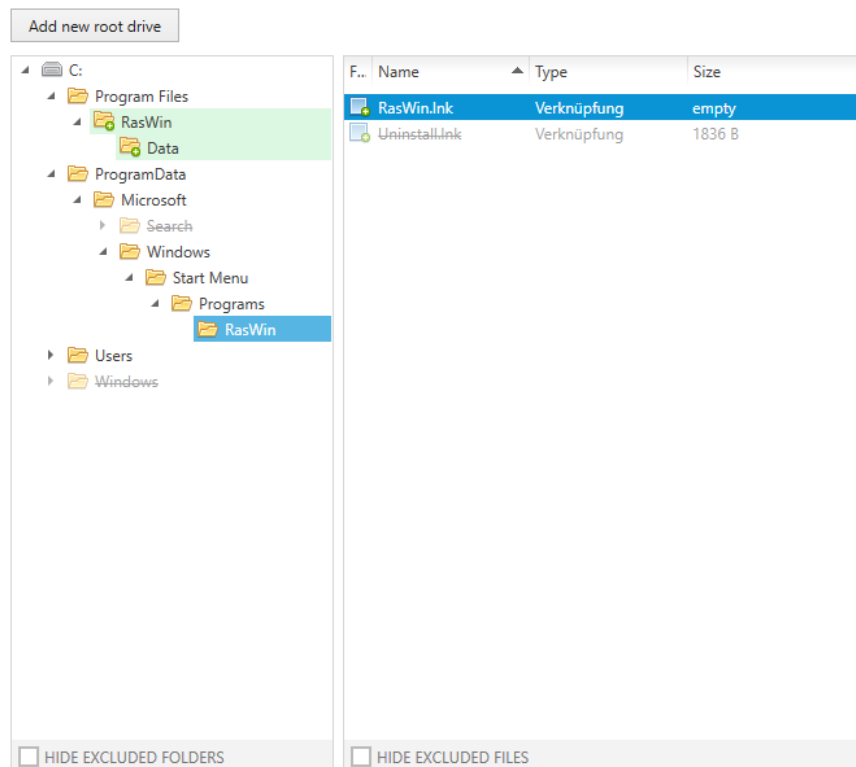
The Directory Tree Area

The directory tree displays all folders that have been recorded as relevant during the capture process.

🖳 The tree contains a root node for each drive that is present. When the Files and Folders view is loaded for the first time, the directory tree is collapsed. Click on the black arrow at the left hand side of the icons to expand the tree. As an alternative, it is also possible to navigate with the arrow keys on the keyboard: Simply select a directory from the tree and move the focus by using the up, down, right, and left arrows on the keyboard.

Each drive contains directories, symbolized by folder icons. The icons are slightly different according to the manipulation that has been applied to the folder during the capture procedure:

A blank folder icon refers to changed content. The directory itself has already been present on the local machine before the capture process has been executed.

A folder icon with a plus symbol refers to a newly created directory.

A folder icon with a minus symbol refers to a directory that has been deleted from the local machine during the capture process.

The name of the directory is displayed at the right-hand side of the icon. If the name is marked with red font color and struck through, it is currently excluded. Excluded resources have been recorded during the capture procedure, but are not relevant for a later target package built from the capture project.

Available Actions in the Directory Tree Area

- **Create subfolder**

  o Right-click a drive or directory and select **Create subfolder** from the context menu
    or
  o left-click a drive or directory and use **Control + D**.

  > **Be aware:**
  > Once a drive or folder has been added to a project it cannot be deleted, only excluded.

- **Import file**

  o Right-click a directory and select **Import** from the context menu
    or
  o left-click a directory and use **Control + F**.

- **Rename folder**

  o Right-click a directory and select **Rename** from the context menu
    or
  o left-click a directory and use **Control + F**.

- **Exclude folder**

  o Right-click an included directory and select **Exclude** from the context menu
    or
  o left-click an included directory and use **Control + E**.

- **Set as install directory**

  o Right-click a directory and select **Set as install directory** to mark that folder as INSTALLDIR.

- **Include folder**

    o Right-click an excluded directory, and select **Exclude** from the context menu
      or
    o left-click an excluded directory, and use **Control + I**.

- **Add a new rule regarding the object to an exclusion list**

    o Right-click a directory, and select **Add to exclusion list...** from the context menu.

- **Hide excluded folders**

    Activate the checkbox at the bottom of the directory tree area to hide any excluded folder
    resource from display. As soon as the checkbox is disabled, the hidden folders are again visible.

The File Details Area

This area displays all files that are stored within the currently selected directory from the tree
view on the left-hand side. Files are symbolized by different icons. The icons are slightly different
according to the manipulation that has been applied to the file during the capture procedure:

A file icon with a pencil refers to changed file content. The file itself has already been
present on the local machine before the capture process has been executed.

A file icon with a plus symbol refers to a newly created file.

A file icon with a minus symbol refers to a file that has been deleted from the local machine
during the capture process.

The name of the file is displayed at the right-hand side of the icon. If the name is marked with red
font color and struck through, it is currently excluded. Excluded resources have been recorded
during the capture procedure, but are not relevant for a later target package built from the
capture project.

Available Actions in the File Details Area

- **Rename file**

    o Right-click a file and select **Rename** from the context menu
      or
    o left-click a file and use **F2**.

- **Exclude file**

    o Right-click an included file and select **Exclude** from the context menu

or
o left-click an included file and use **Control + E**.

- **Include file**

  o Right-click an excluded file and select **Include** from the context menu
  or
  o left-click an excluded file and use **Control + I**.

- **Add a new rule regarding the object to an exclusion list**

  o Right-click a file and select **Add to exclusion list...** from the context menu.

- **Hide excluded files**

  Activate the checkbox at the bottom of the file listing area to hide excluded file resources from display. As soon as the checkbox is disabled, the hidden objects are again visible.

Adding a Root Drive

The **ADD NEW ROOT DRIVE** button at the top of the **Files and Folders** details pane allows the user to add a drive to the capture project. Once the button is clicked, the new root drive will appear. The name of the new root drive is set automatically, according to the already existing set of root drives.

Renaming a Root Drive

In order to rename a root drive, press **F2** when the node is focused and enter the new name.

The newly entered root drive name has to be provided as drive name plus colon, e. g. "`X:`". RayPack automatically prepends the keyword `Drive` and appends a backslash (`\`) to the entered value.

Registry

This view shows the registry keys and values of the capture project. It allows the user to add keys and values to; or exclude keys and values from the target package built from the project. It also enables the renaming of exiting registry keys and values.

The main view consists of two sections: The registry tree on the left and the value details on the right.

The Registry Tree Area

The tree displays all keys that have been recorded as relevant during the capture process.

The tree contains a root node for each hive that is present. When the Registry view is loaded for the first time, the tree is collapsed. Click on the black arrow at the left hand side of the icons to expand the tree. As an alternative, it is also possible to navigate with the arrow keys on the keyboard; simply select a hive or key from the tree and move the focus by using the up, down, right and left arrows on your keyboard.

Each hive contains keys, symbolized by icons. The icons are slightly different according to the manipulation that has been applied to the object during the capture procedure:

   A blank icon refers to changed content. The hive or key itself has already been present on the local machine before the capture process has been executed.

   An icon with a plus symbol refers to a newly created key.

   An icon with a minus symbol refers to a key that has been deleted from the local machine during the capture process.

The name of the key is displayed at the right-hand side of the icon. If the name is marked with red font color and struck through, it is currently excluded. Excluded resources have been recorded during the capture procedure, but are not relevant for a later target package built from the capture project.

Available Actions in the Registry Tree Area

- **Exclude key**

  o Right-click an included key, and select **Exclude** from the context menu
    or
  o Left-click an included key, and use **Control + E**

- **Include key**

  o Right-click an excluded key, and select **Exclude** from the context menu
    or
  o Left-click an excluded key, and use **Control + I**

- **Export**

  Exporting registry keys always includes all contents below the selected item. The target format is a standard conform `.reg` file.

o Right-click a key, and select **Export** from the context menu
or
o Left-click a key, and use **Control + R**

- **Add a new rule regarding the object to an exclusion list**

  o Right-click a key, and select **Add to exclusion list...** from the context menu

- **Hide excluded keys**

  Activate the checkbox at the bottom of the key listing area to hide excluded key resources from display. As soon as the checkbox is disabled, the hidden keys are again visible.

The Value Details Area

This area displays all values that are stored within the currently selected key from the tree view on the left-hand side. Values are symbolized by different icons. The icons are slightly different according to the manipulation that has been applied to the file during the capture procedure:

A file icon with a pencil refers to changed file content. The file itself has already been present on the local machine before the capture process has been executed.

A file icon with a plus symbol refers to a newly created file.

A file icon with a minus symbol refers to a file that has been deleted from the local machine during the capture process.

The name of the file is displayed at the right-hand side of the icon. If the name is marked with red font color and struck through, it is currently excluded. Excluded resources have been recorded during the capture procedure, but are not relevant for a later target package built from the capture project.

Available Actions in the Value Details Area

- **Rename value**

  o Left-click a value, and use **F2**

- **Exclude value**

  o Right-click an included value, and select **Exclude** from the context menu
  or
  o Left-click an included value, and use **Control + E**

- **Include value**

o Right-click an excluded value, and select **Exclude** from the context menu
  or
o Left-click an excluded value, and use **Control + I**

- **Add a new rule regarding the object to an exclusion list**

  o Right-click a value, and select **Add to exclusion list...** from the context menu

- **Hide excluded values**

  Activate the checkbox at the bottom of the value listing area to hide excluded values from display. As soon as the checkbox is disabled, the hidden objects are again visible.

Importing Registry Content

The **Import** button at the top of the Registry details pane allows the importing of registry contents into the capture project. Once the button is clicked, a windows file selector dialog is displayed. Navigate to the desired .reg file, select it, and click Open. The contents will be analyzed and imported into the registry contents of the current projects.

> **Be aware:**
> Importing keys and values that have already been present in the project before, will overwrite old data.
> Once registry items are imported into the project, they cannot be removed again, only excluded.

Shortcuts

This view shows the registry entries of the capture project. It allows the user to import, include or exclude shortcuts. The main view consists of the actual shortcuts, the secondary view to the right contains information about the currently selected item and allows actions to be carried out on the currently selected item. Items colored in red and struck through will be excluded when the capture project is built into the target project / package types.

The items that are initially red and struck through are those that have been explicitly set to be excluded using the *Exclusions Editor* for this profile. The view also shows information about the items in the view, namely Name, Target and Working Directory. If available, the corresponding icon is shown next to the entry.
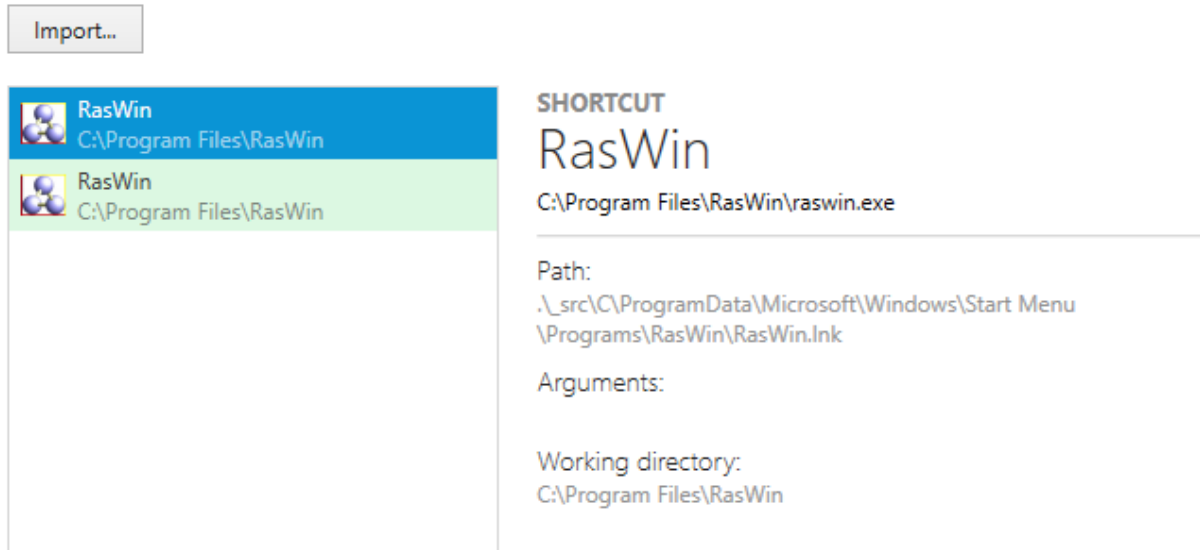
The Shortcuts view shows the captured changes to the shortcuts on the system. Clicking on the column headers allows the sorting of the contents of the view. Furthermore, a context menu is also available for working in the Shortcuts view when an item is selected and clicking the right mouse button.

Available Actions

- **Exclude shortcut**

  o Right-click an included shortcut, and select **Exclude** from the context menu
    or
  o Left-click an included shortcut, and use **Control + E**

- **Include shortcut**

  o Right-click an excluded shortcut, and select **Exclude** from the context menu
    or
  o Left-click an excluded shortcut, and use **Control + I**

Import

The **IMPORT** button allows the user to import a shortcut file (`*.lnk`) into the capture project. For example, if there is a need to add an extra shortcut, choose **Shortcut** from the drop-down, and in the window that is shown, select the destination folder where the shortcut is to be placed.
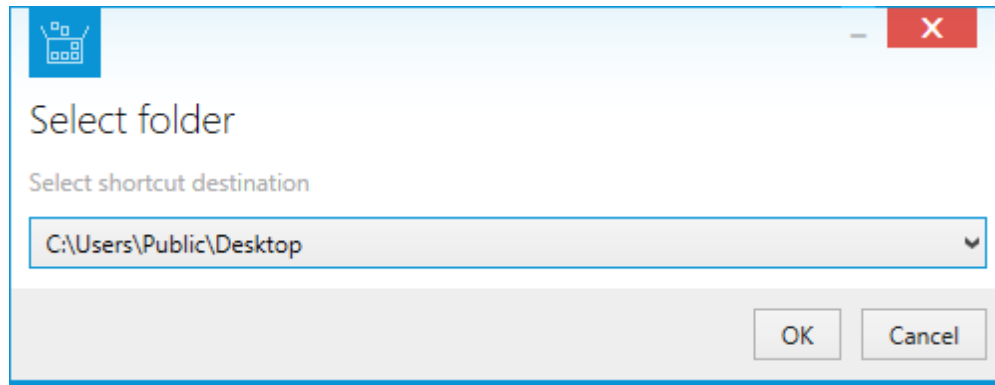
Initially, no selection is present. Click the drop-down icon in the dialog, and from the list of folders, select the folder that is to be the destination folder of the shortcut.

The drop down list displays all folders that are present (and have not been excluded) in the capture project. Select the destination folder and then click the **OK** button. Clicking the **Cancel** button cancels the operation.

Once the **OK** button has been clicked, a decision of what shortcut file to import must be made.
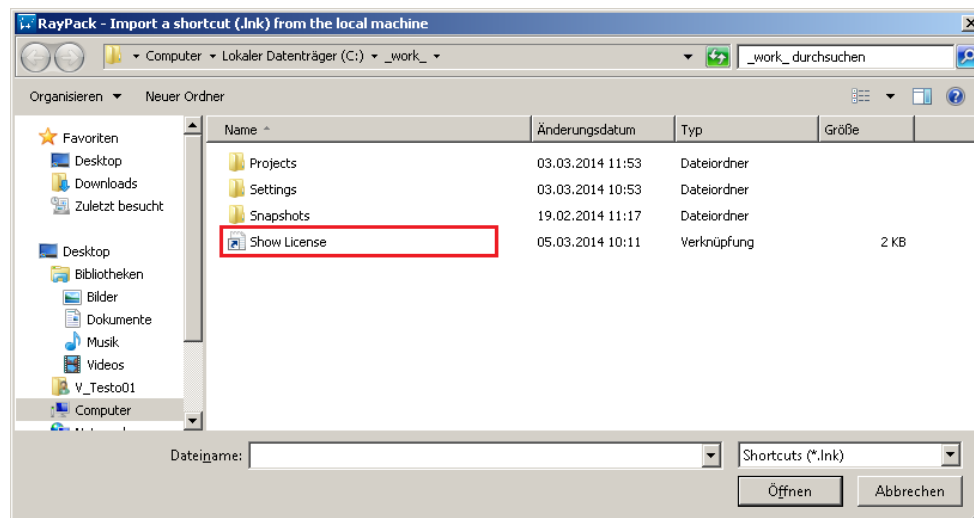
For this example, a shortcut that has been previously created which displays the license text file in the default text editor is chosen. This is then selected and the shortcut is added to the project.

Be aware:
Once a shortcut has been imported into a project, it cannot be deleted, only excluded.

Merge Modules

This view aggregates **Merge Modules** that can be used in output MSI based formats (MSI, RPP, or MST).

By default after repackaging, this section is empty. In order to scan for Merge Modules candidate, press **Scan for replaceable content button.**

After the scanning is done, a list of candidates will be shown, which contains a summary of found modules and the content that matched.

In order to assign one or more Merge Modules, select them using checkboxes and press **Use selected Merge Modules**.

Be aware:
Merge Modules will be applied when you build your project to an MSI, RPP or MST format..

Permissions

This view shows any permission changes of the capture project. Please note that these views will only contain permission changes if the user *originally chose to capture changes to permissions* during the capture process.
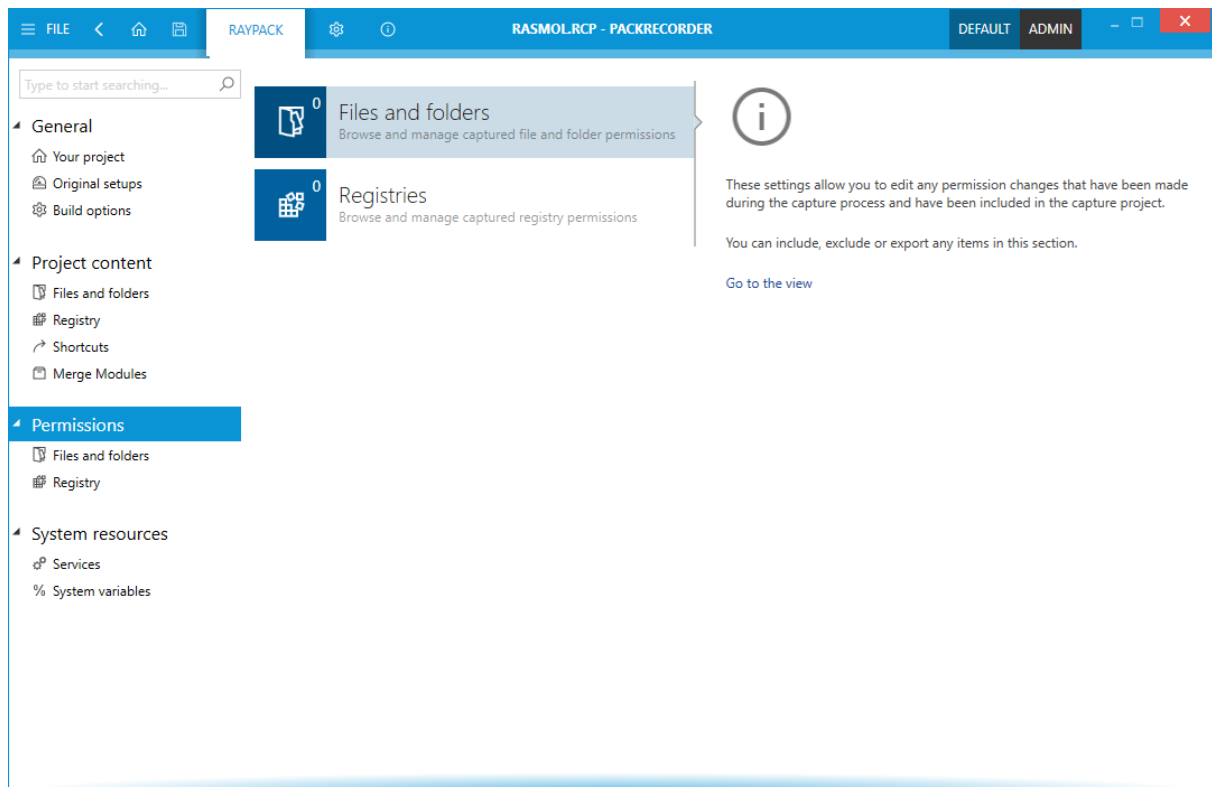
Click on one of the info sections to directly jump to the specific editor view.

Be aware:
These settings will only be applied to RPP, MSI, or MST projects that are built. If the user wishes to include permission changes to other project types, then it must be done by using the methods particular to the specific project type.

Warning:
If there is no intention to use permission changes, it is highly recommend to use only well-known built in accounts for applying permission changes. If accounts that are only present in a specific domain are used, then these permission changes for the specified account may well not be applied, because the account is only known in the corresponding domain.

Files and Folders

This view shows the permission changes to the files and folders of the capture project. For this example, the sub-directory in the captured application (`C:\Program Files\7-Zip\Lang`) was changed is such a way so that Authenticated Users (`SID: S-1-5-11`) have Modify permissions on the folder. This was carried out during the capture process. The view has two columns, Account (the account for which the permission changed has occurred) and the Permissions that have been changed.

The actual SID account is displayed according to the language of the system on which the project has been captured on. However, internally the SDDL notation is used when actually creating a package / project from the capture project and is thus language independent.

> **Be aware:**
> As previously mentioned, these settings will only be applied to RPP, MSI or MST projects that are built. If the user wishes to include permission changes to other project types, then it must be done using the methods particular to the specific project type.

Export

Export can be used to export any (included) permissions that are present to a *security template file*. This is nothing more than a simple inf file which contains the information required by the SecEdit tool to perform changes to permissions on the system. It will export all items in the view, regardless if they are selected or not, provided they are not explicitly marked as excluded. To export included permission changes, click the **EXPORT** button, and choose **SecEdit INF file**

The user will then be prompted to select a name for the file, and a location to save the file.
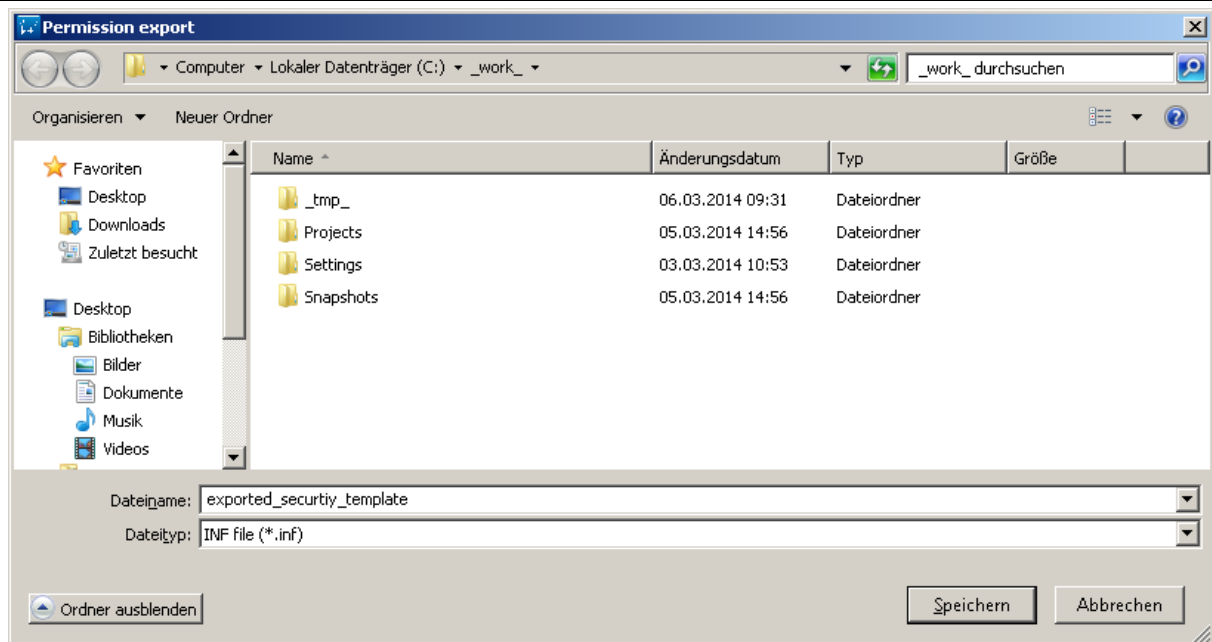
Once the export is completed, the user will have a security template that can be used with the SecEdit tool.
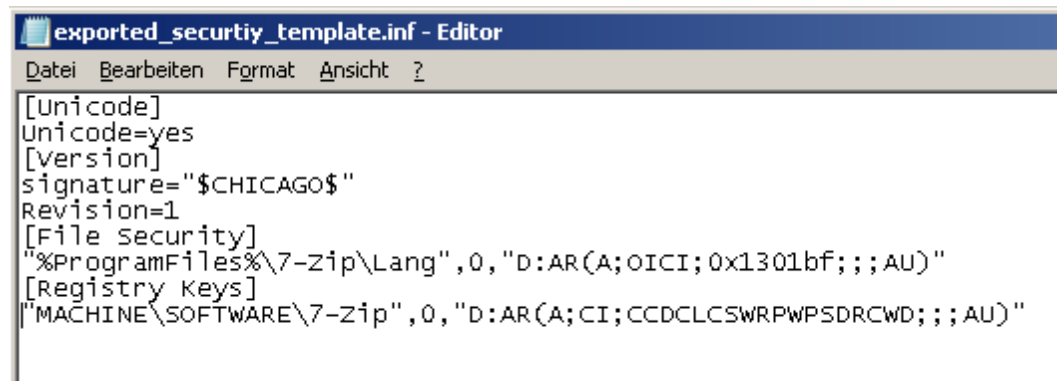
Note:
The resulting exported security template will also contain any Registry permission changes if they are also present and have not been explicitly excluded.

Exclude

If capturing permission changes on the system, the user may be required to manually exclude a specific permission change from the resulting project / package that is to be created from the capture project.

To use the context menu to exclude a permission, select the item that is to be excluded and click the right mouse button, choose the **Exclude** item from the menu. As an alternative, exclusion may be achieved by left-clicking an item and using the key combination **Control + E**. If the

object is marked with red font color and struck through, it is currently excluded.

Notice also that the **INCLUDE** option has become active, as the selected item is marked as excluded and now can be included. The user also select multiple items to be excluded, using the standard windows mechanism for selecting multiple items from a list.

Include

If capturing permission changes on the system, the user may be required to manually include a specific permission change from the resulting project / package that is to be created from the capture project.

To use the context menu to include a permission, select the item that is to be included and click the right mouse button, choose the **Include** item from the menu. As an alternative, inclusion may be achieved by left-clicking an item and using the key combination **Control + I**.

You may also notice that the **EXCLUDE** option has become active, as the selected item is marked as to be included, it can now be excluded. You may also select multiple items to be included, using the standard windows mechanism for selecting multiple items from a list.

Registry

This view shows the permission changes to the files and folders of the capture project. For this example, the registry hive in the captured application (`HKLM\Software\7-Zip`) was changed is such a way so that Authenticated Users (`SID: S-1-5-11`) have extended permissions on the registry key (and sub-keys if present). This was carried out during the capture process. The view has two columns, Account (the account for which the permission changed has occurred) and the Permissions that have been changed.

What might be noticed, is that the actual SID account is displayed according to the language of the system on which the project has been captured on (in this case German), how ever internally the SDDL notation is used when actually creating a package / project from the capture project and is thus language independent.

Be aware:
As previously mentioned, these settings will only be applied to RPP, MSI or MST projects that are built. If the user wishes to include permission changes to other project types, then it must be done using the methods particular to the specific project type.

Export

Export can be used to export any (included) permissions that are present to a *security template file*. This is nothing more than a simple inf file which contains the information required by the SecEdit tool to perform changes to permissions on the system. It will export all items in the view, regardless if they are selected or not, provided they are not explicitly marked as excluded. To export included permission changes, click the **EXPORT** button, and choose **SecEdit INF file**

The user will be prompted to select a name for the file, and a location to save the file.
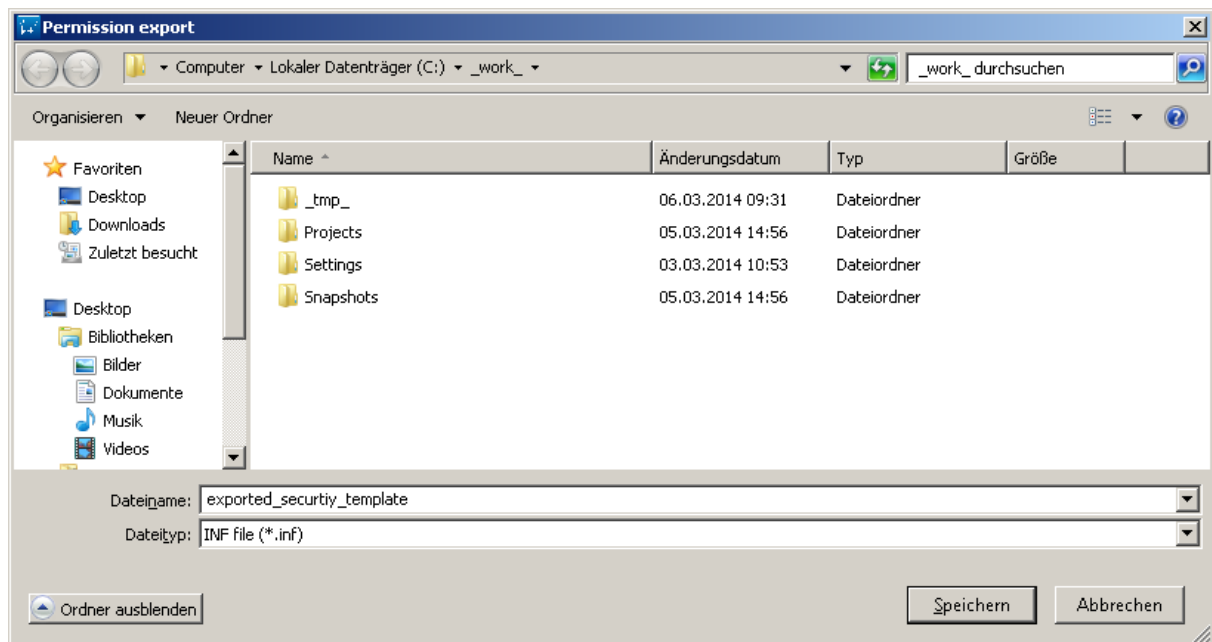
Once the export has been completed, the user will have a security template that can be used with the SecEdit tool.

> **Note:**
> The resulting exported security template will also contain any file or folder permission changes if they are also present and have not been explicitly excluded.

Exclude

When capturing permission changes on the system, the user may be required to manually exclude a specific permission change from the resulting project / package that is to be created from the capture project.

To use the context menu to exclude a permission, select the item that is to be excluded (in this

example, the changes to `HKEY_LOCAL_MACHINE\SOFTWARE\7-Zip`) and click the right mouse button, choose the **Exclude** item from the menu. As an alternative, exclusion may be achieved by left-clicking an item, and using the key combination **Control + E**. If the object is marked with red font color and struck through, it is currently excluded.

It might also be noticed that the **INCLUDE** option has become active, as the selected item is marked as excluded and can now be included. You may also select multiple items to be excluded, using the standard windows mechanism for selecting multiple items from a list.

Include

If capturing permission changes on the system, you may be required to manually include a specific permission change from the resulting project / package that is to be created from the capture project.
To use the context menu to include a permission, select the item that is to be included (in this example, the changes to the registry key `HKEY_LOCAL_MACHINE\SOFTWARE\7-Zip`) and click the right mouse button, choose the **Include** item from the menu. As an alternative, inclusion may be achieved by left-clicking an item, and using the key combination **Control + I**.

One might also notice that the **EXCLUDE** option has become active, as the selected item is marked as to be included and can now be excluded. The user may also select multiple items to be included, using the standard windows mechanism for selecting multiple items from a list.

System Resources

This view shows the settings recorded for services and system variables during the *capture process*. Please note, that these views will only contain changes if the user *originally chose to capture changes to services* during the capture process.
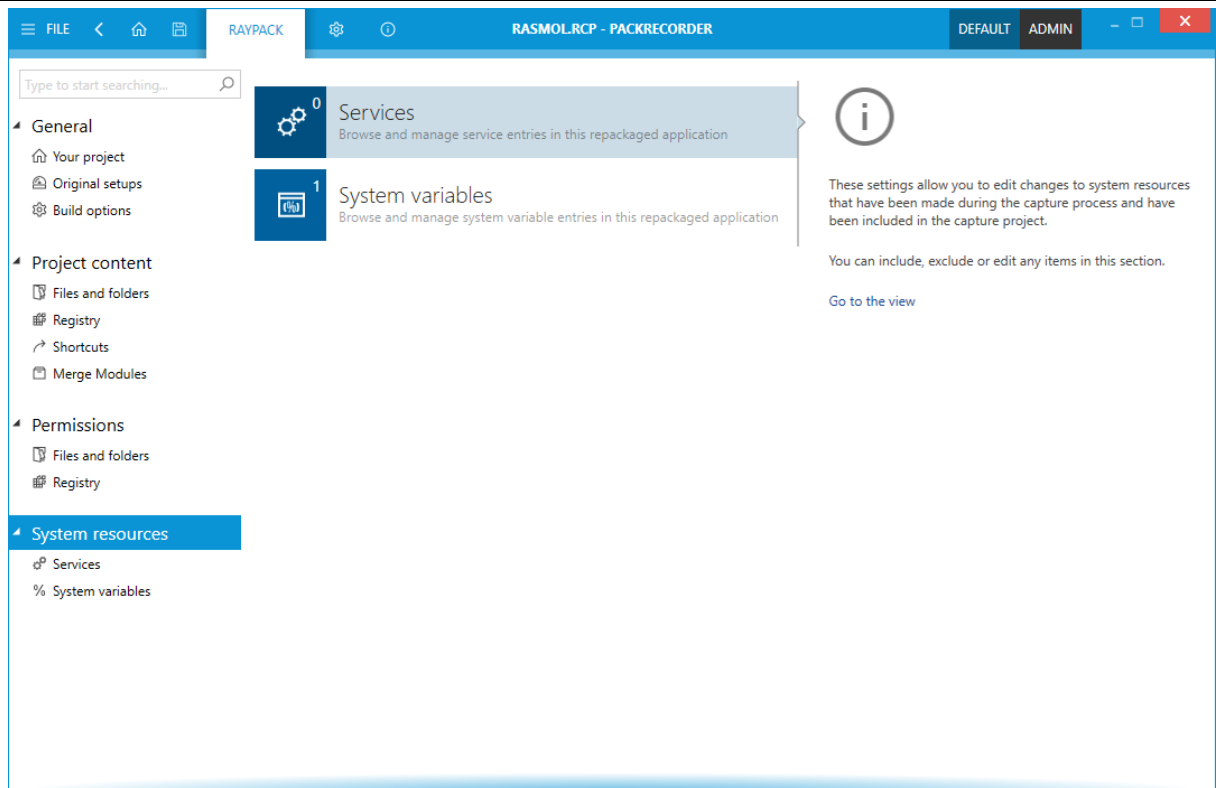
Click on one of the info sections to directly jump to the specific editor view.

Be aware:
These settings will only be applied to RPP, MSI, or MST projects that are built. If changes need to be made to services and or system variables for other project types, then it must be done using the methods particular to the specific project type.

Services

This view shows the configuration of services that are present in the capture project. A brief overview about services can be found on the MSDN website here, and an in-depth description of windows services applications can be found here.

Be aware:
These settings will only be applied to RPP, MSI, or MST projects that are built. If changes need to be made to services for other project types,then it must be done using the methods particular to the specific project type.

Exclude

Although the standard exclusions in RayPack are thorough, it might be required to manually exclude services from the resulting project / package that is to be created from the capture project, depending on your own environment. Excluding an entry is accomplished by selecting the item to be excluded, opening the context menu, and then clicking the **EXCLUDE** option. As

an alternative, exclusion may be achieved by left-clicking an item and using the key combination **Control + E**.

One may also notice that the **INCLUDE** option has become active, as the selected item is marked as excluded and can now be included. The user may also select multiple items to be excluded using the standard windows mechanism for selecting multiple items from a list.

Include

During the capture process, if a service changes or is created on the system, then it may be required to manually include a specific service in the resulting project / package that is to be created from the capture project. Excluding an entry is accomplished by selecting the item to be excluded, opening the context menu, and then clicking the **INCLUDE** option. As an alternative, inclusion may be achieved by left-clicking an item and using the key combination **Control + I**.

One may also notice that the **EXCLUDE** option has become active, as the selected item is marked as included and can now be included. The user may also select multiple items to be excluded using the standard windows mechanism for selecting multiple items from a list.

Information

This view shows the various properties of the service. In this view you may change the various properties to meet your own needs.

Display Name:

The display name to be used by user interface programs to identify the service such as `services.msc`. This string has a maximum length of 256 characters. The name is case-preserved in the service control manager. Display name comparisons are always case-insensitive. This is not to be confused wit the actual name of the service as used by the service control manager.

Description:

The display name to be used by user interface programs to identify the service. This string has a maximum length of 256 characters. The name is case-preserved in the service control manager. Display name comparisons are always case-insensitive.

Executable:

This is the fully qualified path to the service binary file. If the path contains a space, it must be quoted so that it is correctly interpreted. The path can also include arguments for an auto-start service (for example, `"C:\Program Files\UltrasVNC\WinVNC.exe arg1 arg2"`). These arguments are passed to the service entry point (typically the main function).

Start Type:

Here are the service start options. This parameter can be one of the following values.

- **Auto Start** (0x02) - A service started automatically by the service control manager during system startup.
- **Demand Start** (0x03) - A service started by the service control manager when a process calls the StartService function.
- **Disabled** (0x04) - A service that cannot be started. Attempts to start the service resulting in the error code `ERROR_SERVICE_DISABLED`.

The start types Boot Start (0x00) and System Start (0x01) are not selectable, as they are only relevant for drivers and are therefore not available. If these types of service have been captured, then please examine the capture project for drivers that have been installed. These will have to be dealt with separately depending on the type of project that is to be generated.

> **Note:**
> The Auto Start (Delayed) parameter is only valid for Windows Vista or later operating systems.

Error Control:

Here we discuss the severity of the error and action taken, if this service fails to start. This parameter can be one of the following values:

- **Critical** (0x03) - The startup program logs the error in the event log, if possible. If the last-known-good configuration is being started, the startup operation fails. Otherwise, the system is restarted with the last-known good configuration.
- **Ignore** (0x0) - The startup program ignores the error and continues the startup operation.
- **Normal** (0x01) - The startup program logs the error in the event log, but continues the startup operation.

System Variables

This view shows the configuration of system variables that are present in the capture project. This view allows the user to edit, include or exclude system variables. The main view consists of the actual system variables, and the secondary view to the left contains actions to be carried out on the currently selected item. Items colored in red and struck through will be excluded when the capture project is built into the target project / package types.

The items that are initially red and struck through are those that have been explicitly set to be excluded using the Exclusions Editor for this profile. The view also shows information about the items in the view, namely Name, Value and the Type of system variable, Machine or User.

System Variable Types

Variables like `%SystemRoot%` and `%WinDir%` are just plain environment variables. The only difference is where their values come from:

- The system environment variables are predefined and determined by setup.

- The configurable system wide environment variables defined in the `HKEY_LOCAL_MACHINE` `\SYSTEM\CurrentControlSet\Control\Session Manager\Environment` registry key, are effective for any user
- The configurable user specific environment variables defined in the `HKEY_CURRENT_USER` `\Environment` registry key.

The effective process environment is a merge of these three sources.

The latter two registry keys can be edited from the Control Panel -> System applet, Environment Variables button. Beware, making the changes effective **may** require a log off and logon so that the process starts with a fresh copy of the environment instead of a stale one it inherits from its parent process.

Machine

This is the internal descriptor for the environment variable that is valid for all users and processes on the system (system environment variable). The information regarding system environment variables is stored in the registry key `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet` `\Control\Session Manager\Environment`.

User

This is the internal descriptor for the environment variable that is valid for the currently active user and any processes that he / she may start. The information regarding system environment variables is stored in the registry key `HKEY_CURRENT_USER\Environment`.

Exclude

Although the standard exclusions in RayPack are thorough, it may be required to manually exclude changes to system variables from the resulting project / package that is to be created from the capture project, depending on your own environment. Excluding an entry is accomplished by selecting the item to be excluded, opening the context menu, and then clicking the **EXCLUDE** option. As an alternative, exclusion may be achieved by left-clicking an item and using the key combination **Control + E**.

It may also come to your attention that the **INCLUDE** option has become active, as the selected item is marked as excluded and can now be included. The user may also select multiple items to be excluded using the standard windows mechanism for selecting multiple items from a list.

Include

During the capture process, if an environment variable changes on the system, the user may be required to manually include a specific environment variable change in the resulting project / package that is to be created from the capture project. Excluding an entry is accomplished by selecting the item to be excluded, opening the context menu, and then clicking the **INCLUDE** option. As an alternative, inclusion may be achieved by left-clicking an item and using the key combination **Control + I**.

It may also be realized that the **EXCLUDE** option has become active, as the selected item is marked as included and can now be included. The user may also select multiple items to be excluded using the standard windows mechanism for selecting multiple items from a list.

Editing an Environment Variable's Properties

Editing the Value of an Environment Variable

To edit the value of an environment variable, select the environment variable whose value is to be edited, click in the **Value** field and if required, scroll using the left and right arrow keys to the position that is to be edited. Enter / Add the information desired to be edited by clicking anywhere in the view. The changes are then visible.

Editing the Type of an Environment Variable

To edit the **Type** of environment variable, just select the environment variable to be edited, and in the **Type** column, select between Machine and User.

⚠️ **Warning:**
Please be careful when editing environment variables, as not only the captured application may not work, but also the operating system can be detrimentally effected by changes made to environment variables.

Adjusting Exclusion Rules from the Editor Interface

When RayPack creates an RCP file as a result of a system snapshot comparison, it uses pre-configured exclusion rules, bundled as exclusion lists. Each exclusion list may be activated or deactivated individually per capture procedure. Usually, the exclusion lists are maintained via the settings area. However, there are times when reviewing a capture result from within the RCP Editor interface, reveals that a certain rule should be added to an exclusion rule set to optimize the results of upcoming capture procedures.

In order to allow packagers to achieve that with minimal effort, there is an "Add to exclusion list" option available from the context menu of the editor views Project content > Files & Folders (for file and folder objects) and Project content > Registry (for registry key and value objects). The following sections describe how to use the Exclusion list modification wizard that is launched by these context menu options.

> **Note:**
> It is not possible to edit or remove already existing exclusion rules by the interface provided by this editor. Please use the exclusion list editor interface, which is available from the settings area to manage existing exclusion rules (and lists).

> **Be aware:**
> If there are several rules that match a specific capture result object, the last one takes precedence over all others. This means that if there are two rules within an exclusion list, that have regular expressions which fit on a specific object, RayPack will apply them sequentially.
>
> Since there are only two possible states of one property, it is always the exclude or include command of the last rule that is actually applied to the object. Therefore, adding a rule that demands an object to be included may very well void a prior rule that demanded its exclusion.
>
> Therefore, it is required to double-check the full stack of rules within an exclusion list (or better said within all exclusion lists that have been applied during a capture process) to determine why an object has actually been excluded or not. It is not efficient to simply add new rules, since applying exclusion lists during capture procedures takes time, which increases with a growing number of rules.
>
> Please keep this in mind whilst using the function to add to exclusion lists directly from the RCP Editor interface, and make sure to manage the whole set of exclusion lists from time to time in order to provide a tidy and efficient set of rules that achieves the desired exclusion results at the optimal speed.

To Call the Exclusion List Modification Wizard

1. Within the editor interface for RCP projects, call the **Files & Folders** view or the **Registry** view.

2. Use the tree structure on the left to navigate to the object that has to be added to an exclusion list rule set.
   Adjusting the exclusion list settings is available for object of the type **folder**, **file**, **registry key** and **registry value**.

3. **Right-click** the object and select **Add to exclusion list** from the context menu.

4. The **Exclusion list modification wizard** is displayed.

To Add a Rule to an Exclusion List

Work your way through the steps of the wizard to define all required properties for the new exclusion list rule.

By using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, allows to navigate within the already processed steps at any time.
To exit the wizard without creating a new object, use the Cancel button, which is also located at the bottom of the wizard dialog.

Step 1: Regular Expression

Rule type
- Please select **Exclude** to add a new rule that leads to the exclusion of objects from capture results.
- Please select **Include** to add a new rule that leads to the inclusion of objects into capture results.

Regular expression(s)
The lower part of this wizard dialog contains a set of regular expression input validators. The amount of fields actually depends on the type of object that is about to be added to an exclusion list:

- Object type folder: Input field Location
- Object type file: Input fields Name and Location
- Object type registry key: Input field Location
- Object type registry value: Input fields Location, Name, and Value

The regular expression within each of the displayed validator input fields needs to be reviewed since they are an automatically generated regular expression term, which might very well require fine-tuning. For instance, allowing a more general rule that matches a wider set of objects than just the one that was used to trigger this wizard.

When a regular expression is modified, RayPack validates the new state on the fly. The

expression validation result is displayed at the right hand side of the status bar displayed below the expression input field:

- If the expression is **valid**, the result is labeled VALID and visualized with a **checkmark on green background**.
- If the expression is **invalid**, the result is labeled INVALID, and visualized with an **x on red background**.

However, once a rule has been edited to be valid and to match the actual rule needs, users may click on the **Test expression** button at the lower left corner of the regular expression validator input field. An additional input field is displayed, allowing users to enter a test-string. The default value of that input field is the specification of the data object (file, folder, registry key or value) that was used to trigger the wizard.

The test result is displayed within the status indicator bar at the bottom of the test string input field:

- If the string matches the regular expression, MATCHING is displayed on a green background.
- If the string does not match the regular expression, NOT MATCHING is displayed on a red background.

> **Be aware:**
> Even though it is possible to add exclusion rules that test negative, it is not possible to add rules with invalid syntax. Please make sure to repair any expression that is reported to be invalid within a RayPack warning dialog once the Next button is clicked.

Step 2: Exclusion List

Rule description
The description has to provide details about the reason for the rule existence and the affected object scope. Please keep in mind that exclusion lists are usually maintained over several years and may be used by different packagers during that time. It is recommended to enter a decent description to prevent any issues for exclusion list maintenance later on.

Exclusion list
Select one of the exclusion lists that are available from the predefined set. The new rule will be added to this list.

> **Note:**
> It is not possible to add a rule to more than one list at a time. If it is required to add it to several lists; the wizard has to be executed for each list in turn.

Step 3: Summary

Use the summary page to check the correctness of the driver properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the new exclusion list rule.

- If changes are necessary, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the summary page again.

Step 4: Finished

Once the new exclusion list rule has been created, the wizard can be closed by using the **Finish** button at its lower right corner.

> **Be aware:**
> Adding a new rule to an exclusion list does not affect the current project content. Please use the context menu options Exclude / Include to adjust the exclusion settings for an object that resides within the current project. The exclusion lists are evaluated and applied as a one time measure during the capture process. They are not synchronized with the current packaging project content later on.

Configuring Isolation for Virtualization

Folders and registry keys have a special configuration dialog which allows users to configure the desired level of merging when building App-V 4.6/5.x/Thin-App projects. To trigger the settings, right-click a registry key or a folder and go to the **ISOLATION** tab:
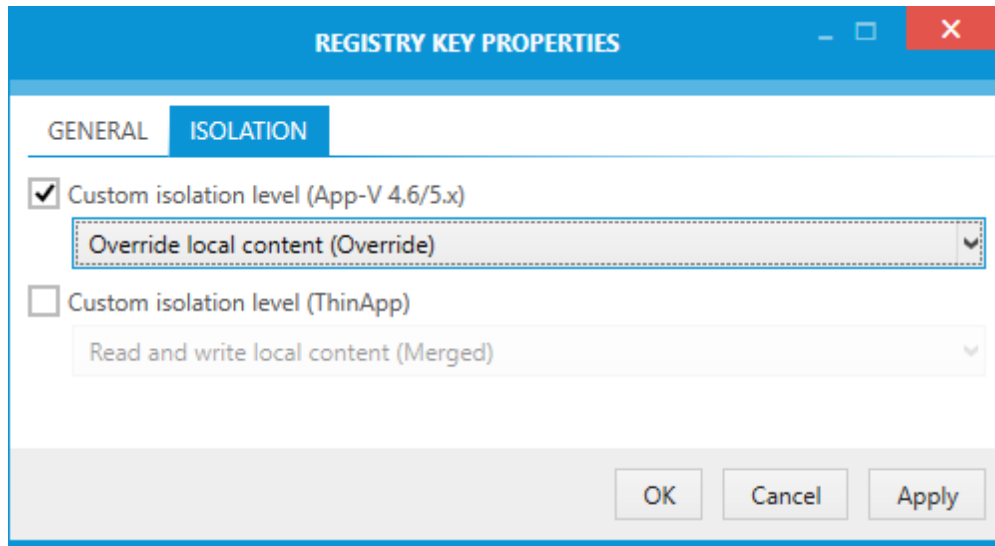
When no custom isolation is used, RayPack tries to use reasonable defaults based on type and location of a resource to provide the best possible results that should work for most of packages.

Building Packages

As soon as a capture project (RCP) has been opened for edition in PackRecorder, it is always possible to build a target package from it.

Always means, that from a technical perspective, the Build option is permanently available. Users may simply use the hot key **F7** to call the Build dialog at any time during a PackRecorder working session. From a logical point of view, packages should not be build from invalid or incomplete resource bundles. Therefore, it is highly recommended to review and adjust project contents first, and only build target packages from them when the interface of the PackRecorder Editor provided sufficient options to ensure that the target package will be fully productive.

In fact, it is recommended to build RPP (RayPack Packaging Project) files first, which may be edited and validated within the substantial PackDesigner Editor interface, before actual target package formats, such as MSI's, are built.

However, **from an RCP project** opened within PackRecorder, packagers may build the following target formats:

- RPP
- MSI
- App-V 4.6[*]

- App-V 5.0*
- ThinApp**
- SWV***
- MSIX
- MSIX app attach (VHD)
- Citrix AppLayering (LAYPKG)
- Intune (win32 package)

_____

* Only when the virtualization pack has been licensed
** Only when the virtualization pack has been licensed, and the ThinApp SDK is installed on the packaging machine. Please refer to the _Release Notes_ for details regarding supported ThinApp SDK versions
*** Only when the virtualization pack has been licensed, and the SWV agent is installed on the packaging machine. Please refer to the _Release Notes_ for details regarding supported SWV agent versions

Be aware:
Building is a different process than simply saving. When a target package is build, all resources are checked for validity and existence, all target format restrictions are evaluated, and finally, the target package files (e. g. `*.msi` and `*.cab` files) are newly generated.

Saving changes made to a file does not include all these steps, but simply saves the updated state of the current project.

WARNING
Building means to determine the target architecture as well. It is not recommended to switch the architecture as part of the transition between RCP and any target package or project type. Doing so may cause issues, since the recorded snapshots base upon changes that affected the original system in its current architecture state (either 32-bit or 64-bit).

Essential modifications that are only present for another system architecture will most likely not be available within the RCP, and therefore missing in any exported format if the target architecture is switched.

RayPack allows users to perform this type of configuration switch, but it is highly recommended to double-check the requirement to do so, and triple-check the usability of the generated target file.

To Build a Target Package

1. Click on the **FILE** button within the Main Toolbar at the top of the PackRecorder Editor application screen, or simply hit **F7** on the keyboard.
   o The same result is achieved by pressing the **Build package...** button on the bottom of the

screen.

2. Select **Build** from the options menu column on the left-hand side

   Make sure the radio button **To disk** is selected.

3. Decide whether the newly created target format object should be displayed in the context of a system explorer instance once the build process has finished. If such an **inspection explorer** is wanted, users have to activate the checkbox **Open the folder after the building is finished**, displayed within the Settings section, at the lower dialog area.

4. Click on the **tile** that represents the desired **target format** (see the *list of available target formats* above)

5. Define the target package **name** and **location**

6. Click **save**

7. Wait for the process to finish.

   Depending on the complexity of the resources that have to be molded into the target package format, the build process may take a while. Please note the process information displayed within the progress dialog: The currently processed resource is displayed by type and path, indicating the area of activity and possible reasons for longer waiting periods (e. g. because a file that has to be deployed with the package is quite large).

To Build a Target Package and Save It to RayFlow

Chapter Saving files in RayFlow describes how to build a package and upload it automatically to the current RayFlow instance.

Building MST Files Against Vendor MSI

It is possible to build RCP projects directly to an MST file format. The option is visible in the format selector as long as the path to vendor MSI is configured in the Build options. This is done automatically if the setup has been:

- Recognized while repackaging with PackRecorder (see chapter *Recognition of Vendor MSI Installations* for more details), or
- the project has been converted by PackBot and an Installer session has been captured as described in the *Detection of Vendor MSI Setups* chapter.

If none of these happened and you still want to create an MST file, then a path to vendor MSI has to be configured per hand in the Build options.

> **Be aware:**
> The MST file format is only visible if the vendor MSI is non-empty. If the format is not visible on the list, make sure to go to the options and add the path to the right place.

Capturing Large Packages

Whilst the capturing and comparison modules are fast on their own, certain settings of the scanning should be fine-tuned when a big package (more than 50 000 files or so) is to be repackaged. This chapter summarizes how to identify the bottlenecks and improve the performance of scanning.

Permissions Scanning

By default, RayPack is scanning the system changes and looking for changes in the file/registry permissions. Scanning of permissions requires additional time and resources, which - depending on the machine, snapshot, and the package - may negatively impact on the performance and snapshot comparison.

If a package is known to contain no custom permissions, it is recommended to disable the unnecessary scanning options. For most of packages, the permissions are not changed by the installer and thus the option can be disabled without any loss of functionality.

Unnecessary Scanned Areas

If it is possible, the list of scanned resources (drives, registry hives) should be limited to absolute minimum. Drives that are not to be affected by the application (usually non-system drives) are increasing the time needed to finish a snapshot, and make them bigger, which results in more time required for the analysis.

For most of packages, a combination of `C:\` drive and `HKEY_LOCAL_MACHINE` and `HKEY_CURRENT_USER` node is sufficient to perform fast and reliable scanning.

Ignoring Resources

Excluded resources are still scanned and copied to the package folder so that it is possible to revert the initial exclusions in the future without the need to repackage again. This feature has a negative impact on the performance. In order to skip analyzing and copying these resources, adjust the ignore list according to section *Ignoring certain resources when repackaging*.

Computer Parameters

Regardless of all settings and fine-tuning of the snapshotting, it is a general rule of thumb that the better the machine, the faster the scanning will be. Since the repackaging is usually done on a virtual machine, it is important to assign as much RAM memory and as many CPU cores as possible. Because scanning requires a lot of I/O operations, it is a good choice to store the virtual

machine files on an SSD drive.

Standalone (Portable) Repackaging

> **Be aware:**
> This chapter describes how to use RayPack on virtual machines to provide a clean repackaging without requiring a separate license for the packaging machine. In most cases, it is better to use PackBot to automate this task, but users may use the following instructions to revert to a fallback in case PackBot is unavailable or the repackaging must be done on a remote machine or on a machine virtualized by anything other than VMWorkstation, ESX, or Hyper-V.

Using RayPack requires a product activation. Whilst the implemented web-based activation procedure works fine for permanent packaging machines, it is not the handiest possible solution for the maintenance of several clean machines for repackaging. Usually, these are various sets of virtual machines which are reset for every new repackaging process. Installing the full RayPack framework on every clean machine would also lead to a high number of required licenses. This is not the intended licensing model Raynet has in mind for RayPack.

The solution to this issue is a special mode in which RayPack provides the basic functionality without checking for a full product license. This means that the product configured in the standalone mode can be used on as many machines as required. The standalone mode does not require an internet connection or access to the activation server. It is fully portable and requires no installation (although it is possible to install the product in the standalone mode).

Limitations of the Standalone Mode

There are several limitations of the functionality. Basically, using a standalone installation allows to perform the whole repackaging process and to create RCP projects.

Some selected features that are not present in the standalone mode are:

- Editing and creating MSI / RPP projects (PackDesigner).
- Creating response transforms (PackTailor).
- Building to deployable formats (MSI, MST, MSP, App-V, ThinApp, SWV, MSIX, VHD, LAYPKG).

In order to create a deployable package (MSI, MSIX, virtual packages, etc.) or use features that are not otherwise present in standalone mode, a full valid RayPack license is required. In a typical scenario, the standalone mode can be used to perform the repackaging on a clean virtual machine, capture changes, and produce the `.rcp` project. Then, the source files have to be copied to a physical machine where the full RayPack is installed in order to customize the package and convert it to required format.

Installing PackRecorder in Standalone Mode

During the installation it is possible to select the **Standalone** installation mode. Such an instance can be used on any machine.

Starting RayPack from Network Shares / Flash Drives

When RayPack is started from a network share, a mapped network drive, or a flash stick and no other license is present on the machine, then the product will automatically start in the **Standalone** mode.

The **Standalone** mode can also be forced by using the `-Remote` command line switch.

Using Portable Repackager

The **Portable Repackager** is a command-line based tool which can create snapshots and produce RCP files on any machine without RayPack license. You can find out more about its usage and syntax in the chapter Portable Repackager.

PackTailor

PackTailor is RayPack's tool for capturing and editing application setups that are based on MSI technology (MSI files). It uses Windows Installer engine to analyze what resources will be installed where on a target system. The parameters used and changed during normal installation are "captured" and saved in a transform file.

PackTailor uses the InstallUISequence of the MSI file. If the MSI does not have a InstallUISequence, then the PackTailor cannot operate correctly. Normally, in a well authored MSI, no changes are made to the underlying system in this sequence. However some MSIs "break" this rule and do not follow it. If this is the case, the underlying operating system will be changed and therefore it is highly recommended, as when using snapshot technology (PackRecorder), to carry out the process on a "clean-machine".

> **Warning:**
> As the resulting transform file is "tailored" for the actual OS that the transform was created on, as a rule of thumb, the resulting transform **will not** be compatible for other operating systems.
> For example, the PackTailor is run on the Windows XP operating system using a vendor MSI installation that is theoretically compatible for both Windows XP and Windows 7. During the configuration, only features, components and resources that are for Windows XP will be marked for installation. This information is stored in the resulting transform file. If the transform file is used to install the MSI on a Windows 7 operating system, it is possible that incompatible features, components and resources will be installed, and the application may not work as designed and/or required.

PackTailor uses a typical RayPack step-by-step "Wizard" like structure to guide the user through the steps of creating a Transform file (`*.mst`) for an existing Windows Installer File (`*.msi`).

Launching the PackTailor wizard for setup capturing is available at two central places:

- By clicking the **create a new project** tile from the Home Screen
- By clicking the **FILE** tab of the Main Toolbar

Either way, the **NEW** dialog of the FILE menu is displayed. Initializing a new PackTailor wizard run is achieved by clicking on the **response transform** button at the right-hand side of this dialog. Please refer to the Common Dialogs section to read more about the FILE menu.
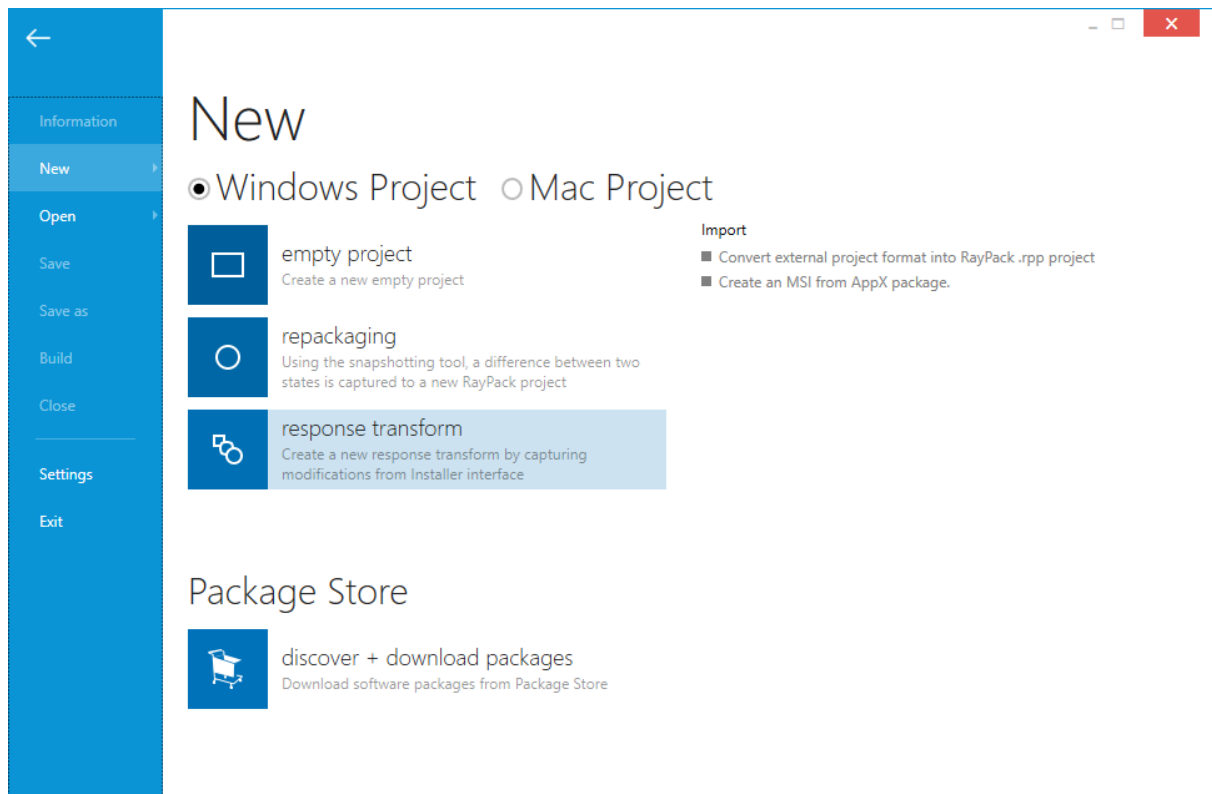
Click on the response transform button to proceed.

Target MSI File

After starting PackTailor, this is the first page that is shown in the Wizard.
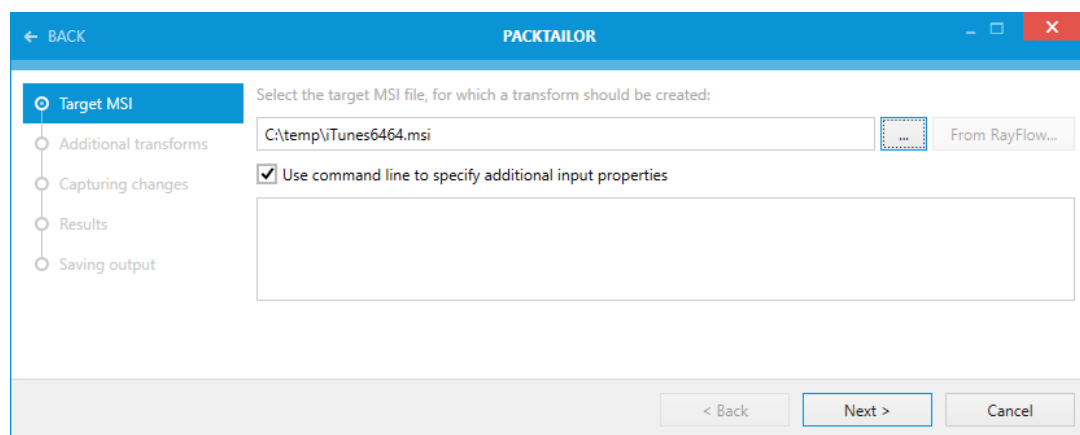
Click the items to navigate to the various sections of the PackTailor process

Select the Target MSI File

Using the browse button **[...]**, select the Windows Installer File (`.msi`) that is to be used to create

a Transform file with PackTailor. The user must choose a valid `.msi` file. Please be aware of the restrictions that some Windows Installer files have as documented *here*.

You can also open a file from RayFlow by pressing the **From RayFlow...** button. The exact procedure is described in section *Tuning files from RayFlow*. The remaining steps are the same as below.

Use the Command Line to Specify Additional Input Properties

Select this option if it is desired to bypass known properties or vendor specific properties that will be used during the creation of the Transform. Please note, that all bypassed properties will be converted to uppercase when used by PackTailor. This is standard behavior (i.e. only public properties can be passed).

Choosing the **Cancel** button will return the user to the **Home Screen**.

Tip:
When creating Transforms for InstallShield based `.msi` files, bypass the Property `ISSETUPDRIVEN=1` using the command line, as this prevents the dreaded "You must run setup.exe..." error message being shown.

Tailoring Files from RayFlow

In Order to Select a Tuned Package from RayFlow:

Be aware:
In order to use this functionality the RayFlow connection needs to be configured first. See chapter *Settings > RayFlow* for more information.

---

1. In the **PackTailor wizard**, click **From RayFlow...** button.

2. A new overlay window will be shown, prompting to select RayFlow phase, task, and a file belonging to a task.
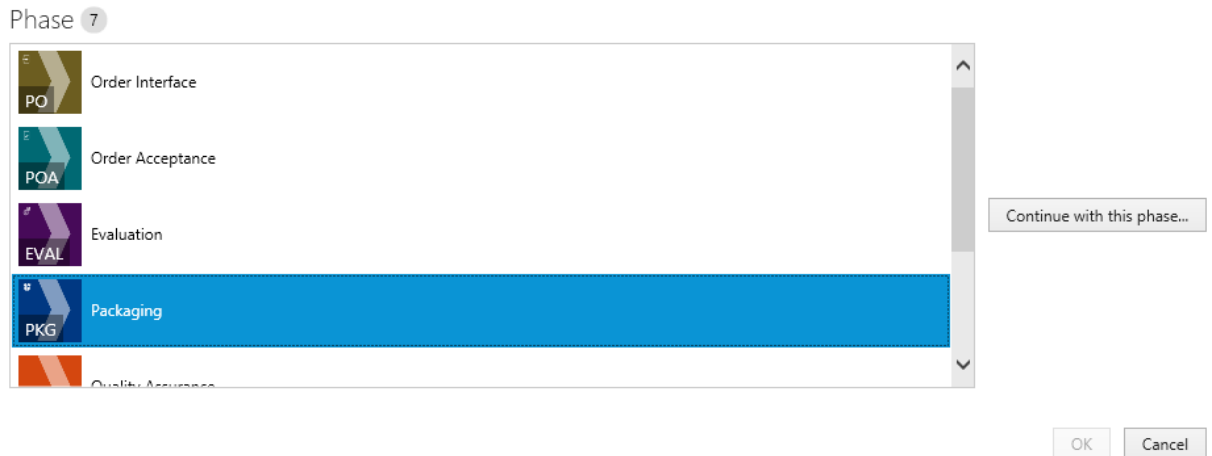
3. Select RayFlow phase which contains the task to open. The counter above the list shows the total number of phases that are visible to the current RayFlow user.

4. Click **Continue with this phase...** to proceed to the task selection with the currently selected phase. Press **Cancel** to go back to the backstage menu.

5. Next, select the task to open:

The number displayed above the list shows the total number of tasks that are visible to the current RayFlow user. The results can be narrowed by ticking **only mine** option, which hides all packages not belonging to the current RayFlow user. The task browser displays the first 5 data fields from RayFlow to enable easier identification (refer to the RayFlow documentation about setting up the data fields).

6. The results can be narrowed by searching, using the search box in the top right corner of the list.

7. Once the required task is selected, press **Continue with this task...** to proceed to the file selection. Press **Cancel** to go to the backstage menu. In order to go back to the phases view, click the arrow on the left side of the list or the partially transparent area containing the tiles representing available phases.

8. On the next screen, select the file to open:

9. The number displayed in the top left corner represents the number of files that are present in the currently selected package.
   o The type column represents the source from which the file can be accessed. Possible values are **Attachment** (file has been physically uploaded to RayFlow) and **Link** (the package is available on a shared location and only the path to the root file is saved in RayFlow).
   o **Date** and **Uploaded by** columns denote who and when has uploaded the file
   o The list is sorted by the date, the most recent items are displayed on the top of the list.
   o Only the following file extensions are shown:
     ▪ `.msi`
     ▪ `.zip` (the file will be extracted, see section *Handling multiple and supporting files within a single package*)
     ▪ `.7z` (the file will be extracted, see section *Handling multiple and supporting files within a single package*)

10. Press **OK** to open the selected file. Press **Cancel** to go back to the backstage. In order to change RayFlow task or phase, click the arrow on the left side of the list or the partially transparent area containing the list of tasks.

11. If the file has been already downloaded, a confirmation screen prompting to either keep the local file and use it as a source for a tuned package, or download the package content again is shown:
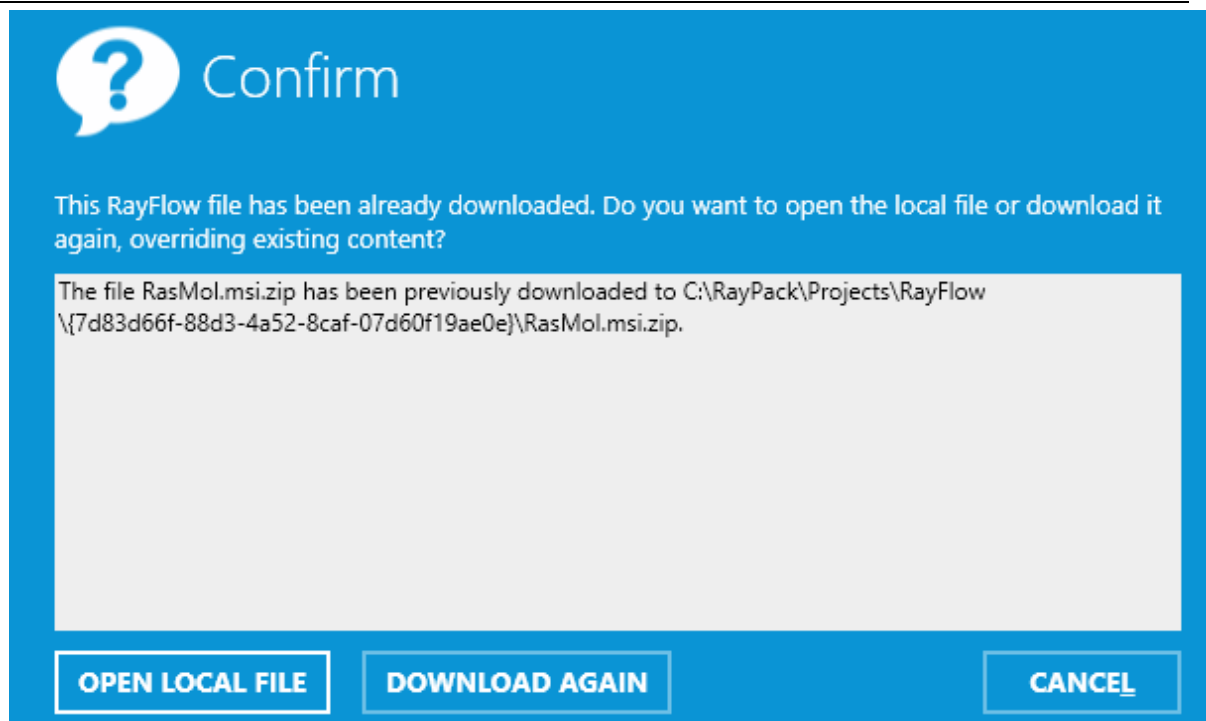
- ○ Press **OPEN LOCAL FILE** to use the already existing content
- ○ Press **DOWNLOAD AGAIN** to download the whole content of the package. Any changes made to the local files will be overridden.

12. If the file is in a ZIP/7Z format and contains several `.msi` files, then the following selection screen may be shown:
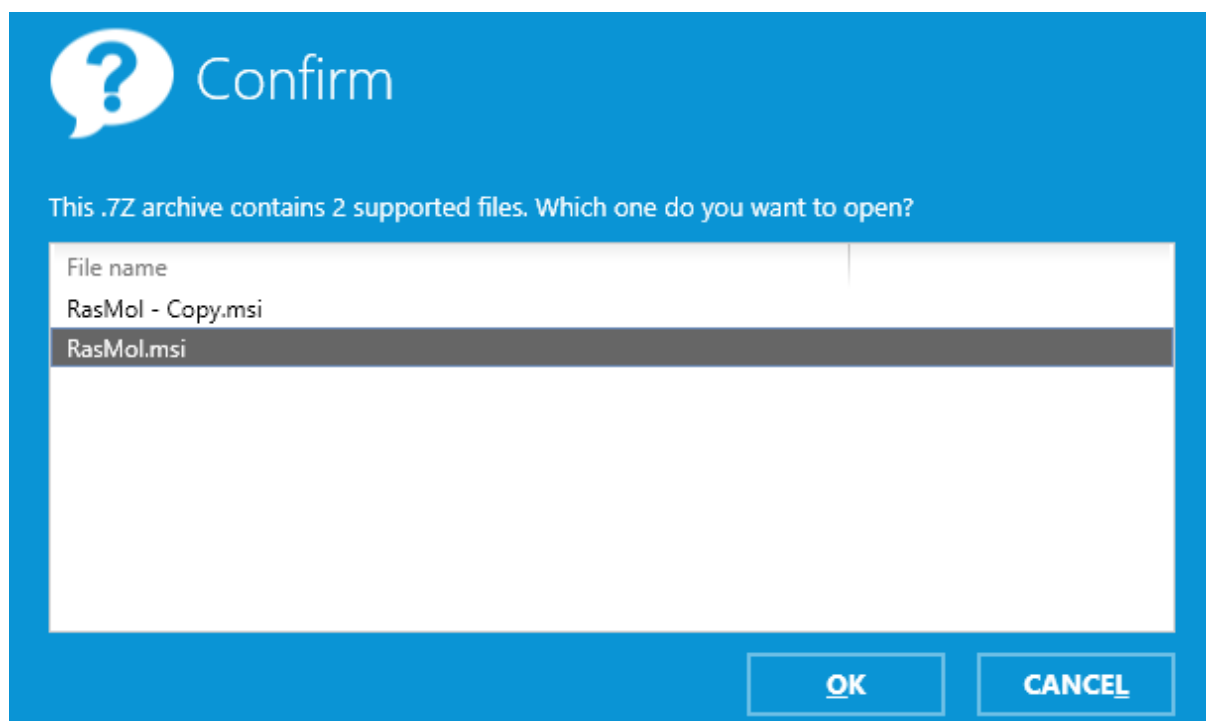
o Press **OK** to use the selected item as a base MSI file.

	Tip:
💡	Tuned transform files downloaded from RayFlow are saved in the Projects folder. Refer to chapter Opening files from RayFlow for more information.

Additional Transforms

This page of the **PackTailor wizard** allows the user to add any additional Transforms.
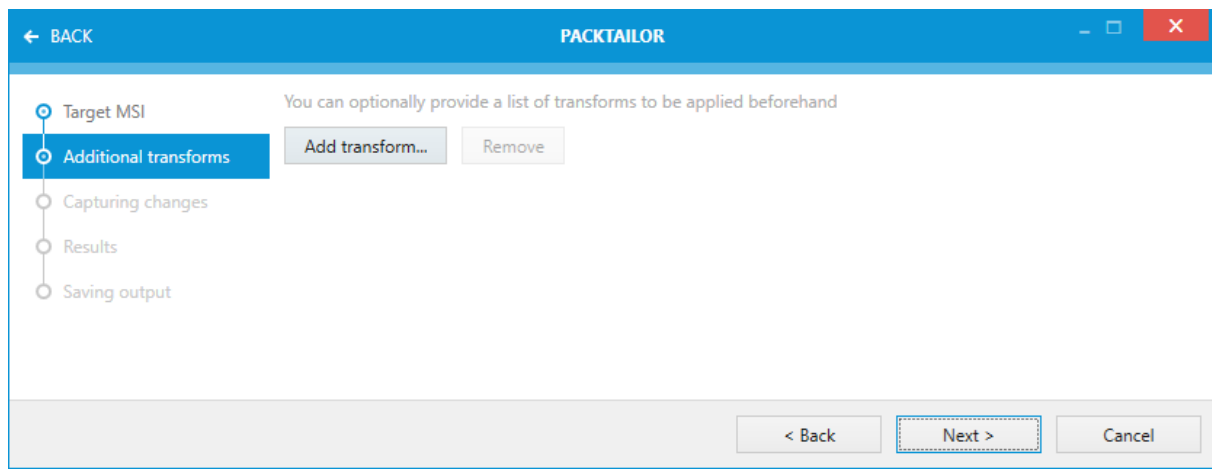
Click the items to navigate to the various sections of the PackTailor process

Adding a Transform

By using the **Add transform...** button, the user can add one or more Transforms that will be applied during the PackTailor process. Some vendor installations supply additional transforms for language or OS specific settings. These can be used here. Once added, they are displayed on this page and will be applied during the PackTailor process.

	Warning:
⚠	If multiple Transforms are added, please be aware that they will be applied in the same order in which they were added to the **Additional transforms** page. At this moment in time, there is no possibility of reordering the sequence in which the Transforms are applied.

Removing a Transform

Using the **Remove** button allows the user to remove any additional Transforms that may have been added. Please note that this button is only active when a Transform that has been added has been selected.

Choosing the **Next** button moves onto the next page of the PackTailor wizard. Choosing the **Back** button, returns to the previous page. Choosing **Cancel**, returns to the **Home Screen**.

Capturing Changes

This page of the **PackTailor wizard** allows to define options that will be used internally by the Windows Installer engine when executing the *selected vendor installation msi* before executing the actual vendor *installation UI Sequence*.

> ⚠️ **WARNING**
> Although RayPack will make no changes to the system, the vendor `.msi` **may** make changes. This is highly dependent on the quality of the MSI that the original vendor has supplied. Even though Microsoft documentation explicitly states that no changes should be made to the system in the UI sequence, this has not stopped vendors from doing exactly that. The result is that often vendor MSIs fail when deployed with anything other that an administrative context. It is therefore advisable to always to carry out the PackTailor process on a clean machine and when finished (if using a virtual machine) return the system to the original state. Also, PackTailor may fail to operate correctly if the vendor MSI does not contain a valid InstallUISequence.
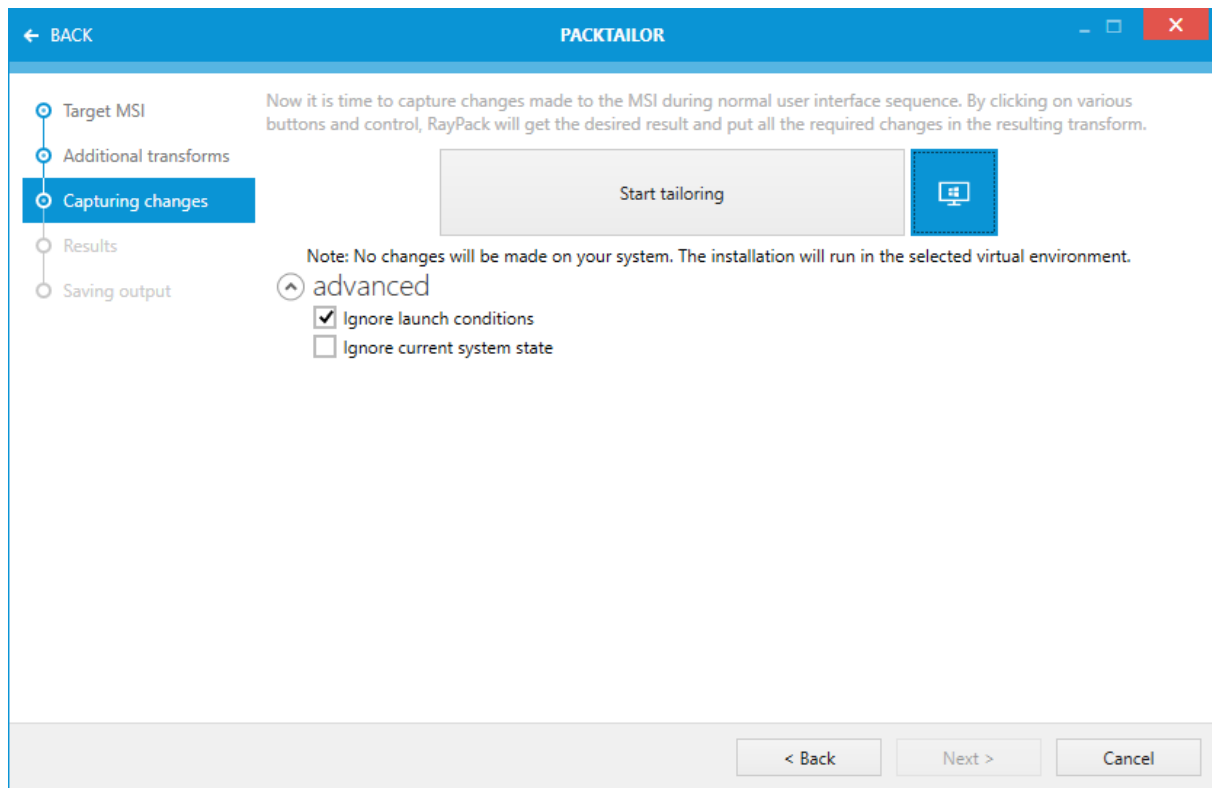
Click the items to navigate to the various sections of the PackTailor process

Start Tailoring

Clicking this button starts the Windows Installer engine, handing over the selected vendor MSI.

During the *installation UI Sequence* the changes that are made will be recorded. The (internal) installer log file and dynamic changes to the `.msi` (in memory) are observed and passed onto the next stage of the wizard. Choosing **< Back** will return to the previous page, choosing **Cancel** will return to the Home Screen.

> **Note:**
> Only the UI sequence is executed and normally with a well authored `.msi` package it makes no changes to the system.

Upon clicking the **Start tailoring** button, the vendor installation is started and the user should configure / execute the installation using the settings that are required to be present in the resulting transform. Once the installation (UI sequence) is complete, the **Results** wizard page is automatically shown.

Before pressing the button, it is possible to define the virtual machine to be used by the tailoring module. To use this option, it is required that at least one virtual machine is predefined in the **Settings** screen. If the button next to the **Start tailoring** is lit (see screenshot above) the machine is armed and ready to use. Otherwise, click the button and select the machine to be used:
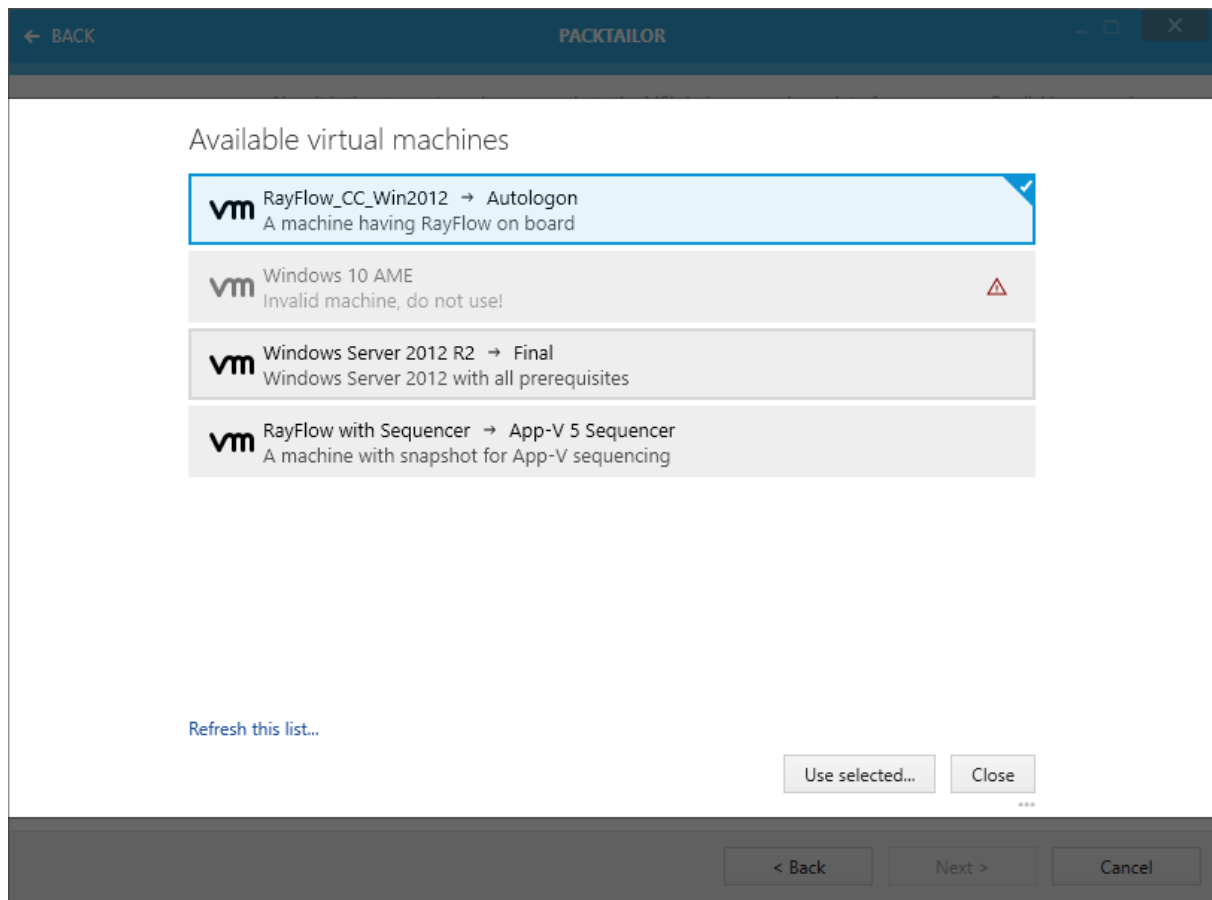
After pressing **Use selected...**, PackTailor tries to communicate with the virtual machine and will show a confirmation prompt when the process is over.

If the path has been entered in UNC format, PackTailor asks whether to copy the file to VM or use it directly:
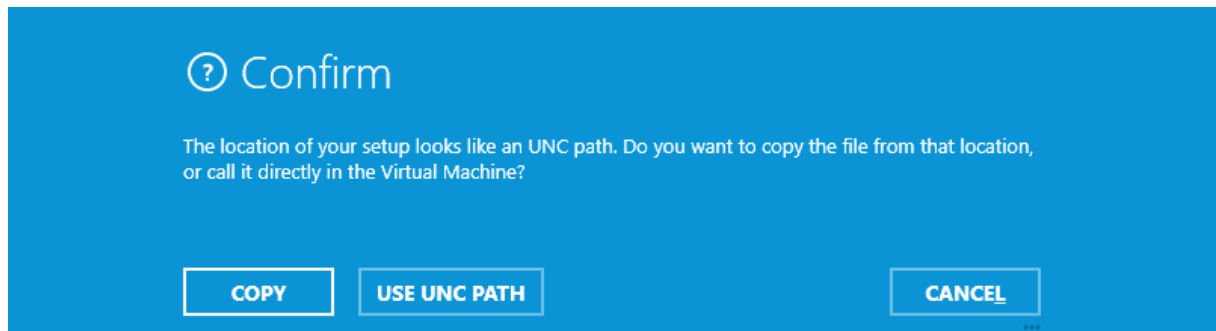
Advanced

These options allow the user to influence the windows installer engine when the vendor `.msi` is executed.

Ignore Launch Conditions

Any conditions that are present in the LaunchCondition table will be (temporarily) removed if this option is activated. This allows the launching of the vendor MSI to create a transform, even if certain conditions are not met.

Ignore Current System State

Activating this option will allow the user to create a transform for a vendor MSI that has already been installed on the system. Instead of receiving the modify / repair user interface, the standard installation sequences user interface will be shown.

Results

This page displays the changes that have been recorded by the PackTailor process. The user can select which changes are to be saved in the resulting transform file. The changes recorded are divided into three categories, **FEATURES**, **PROPERTIES**, and **FOLDERS**.

A number displayed next to each entry informs how many modifications are currently active.
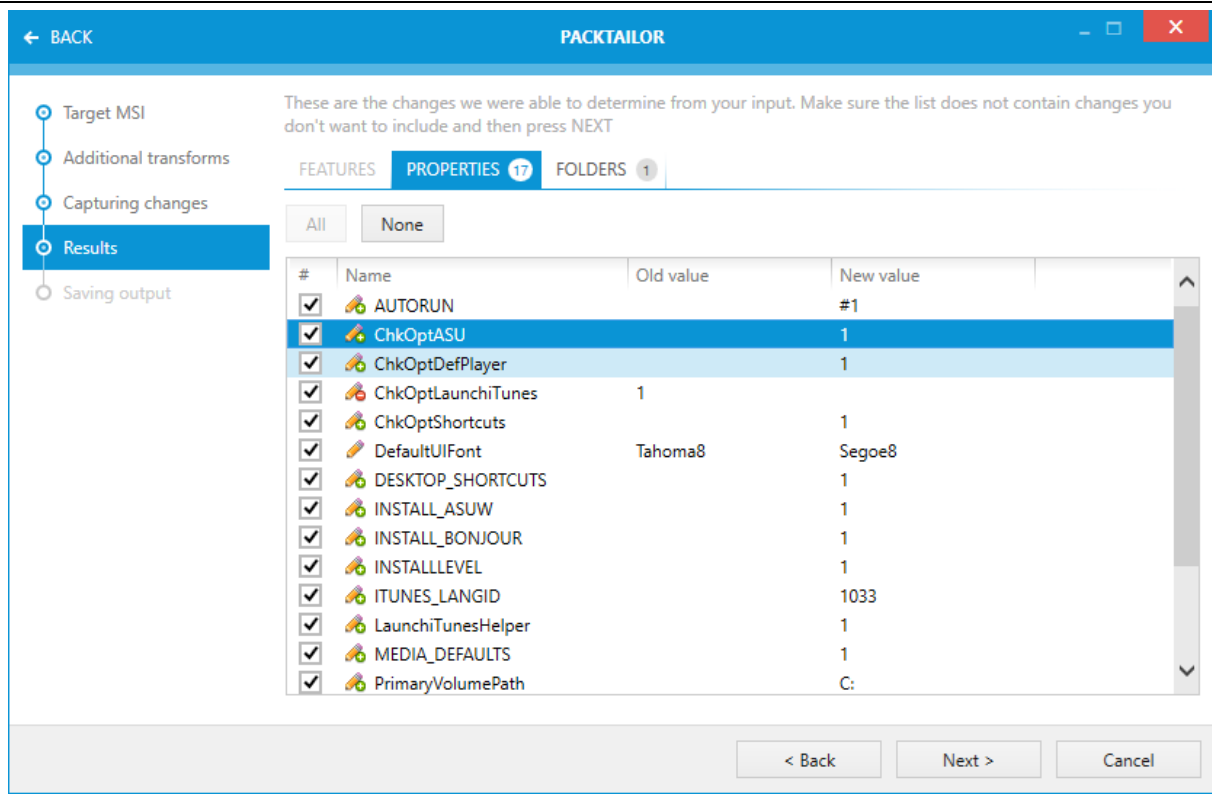
Click the items to navigate to the various sections of the PackTailor process

Features

This tab shows the features whose installation will be modified according to the settings made during the tailoring. The **Old state** column shows the original state of the feature and the **New state** column shows the state that was recorded. Features with **New state Not Install** *are not* to be installed (the 'Level' column of the Feature, in the `Feature` table will be set to '0'). Features with **New state Install** are about to be installed.

When an item in the view is selected, right-clicking it displays the context menu as shown below. This allows the user to quickly select / de-select all of the entries.
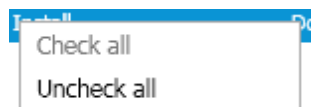
The changes to the **PROPERTIES** and / or changes to **FOLDERS** may be reviewed. Alternatively one can choose the **Next >** button to *continue*. Choosing **< Back** will return to the *previous page*, and choosing **Cancel** will return to the **Home Screen**.

Properties
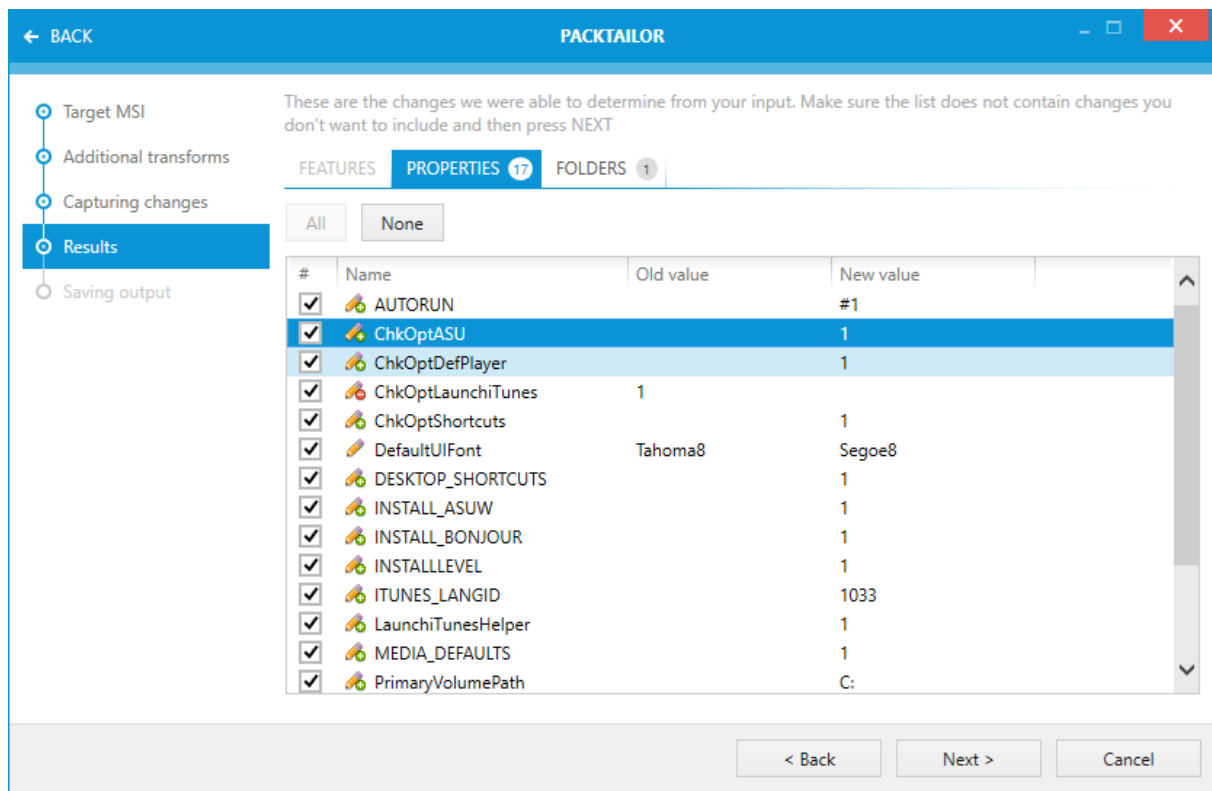
Click the items to navigate to the various sections of the PackTailor process

From the screenshot above, the user can see that the property `AUTORUN` was not initially set in the vendor installation. During the installation it was assigned the value of another property, in this case `#1`.

When an item in the view is selected, right-clicking it displays the context menu as shown above. This allows the user to quickly select / deselect all of the entries.
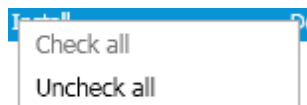
The changes to the **FEATURES** and / or changes to **FOLDERS** may be reviewed. Alternatively on can choose the **Next >** button to *continue.* Choosing **< Back** will return to the *previous page,* and choosing **Cancel** will return to the **Home Screen**.

Folders
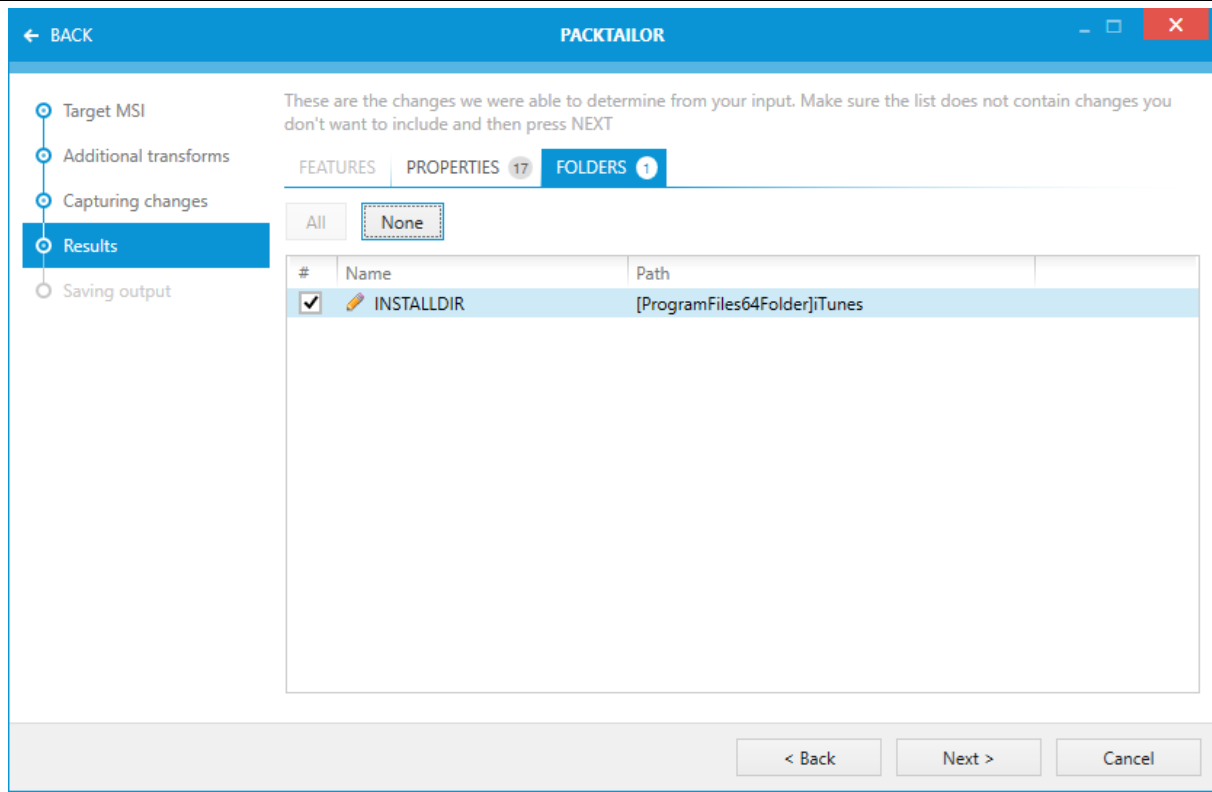
Click the items to navigate to the various sections of the PackTailor process

From the screen shot above, one can see that various directories have been set and or dynamically created. Those entries that remain selected will be added to the `Directory` table of the resulting transform where possible. If this is not possible, then SetProperty (Type 51) custom actions will be used.

Tip:
Depending on the internal structure and use of `CustomActions` inside the tailored package, the section **FOLDERS** may show some changes that were not triggered by the direct manipulation via user interface. Various actions running in the background may change the folder paths as well.

When an item in the view is selected, right-clicking it displays the context menu as shown above. This allows the user to quickly select / de-select all of the entries.

> **Tip:**
> By default, folders resolving to user locations (for example `[AppDataFolder]` and its children) are unchecked. If it is certain that the captured change is relevant and is a critical piece of your installation, then make sure to tick the appropriate checkbox before continuing.
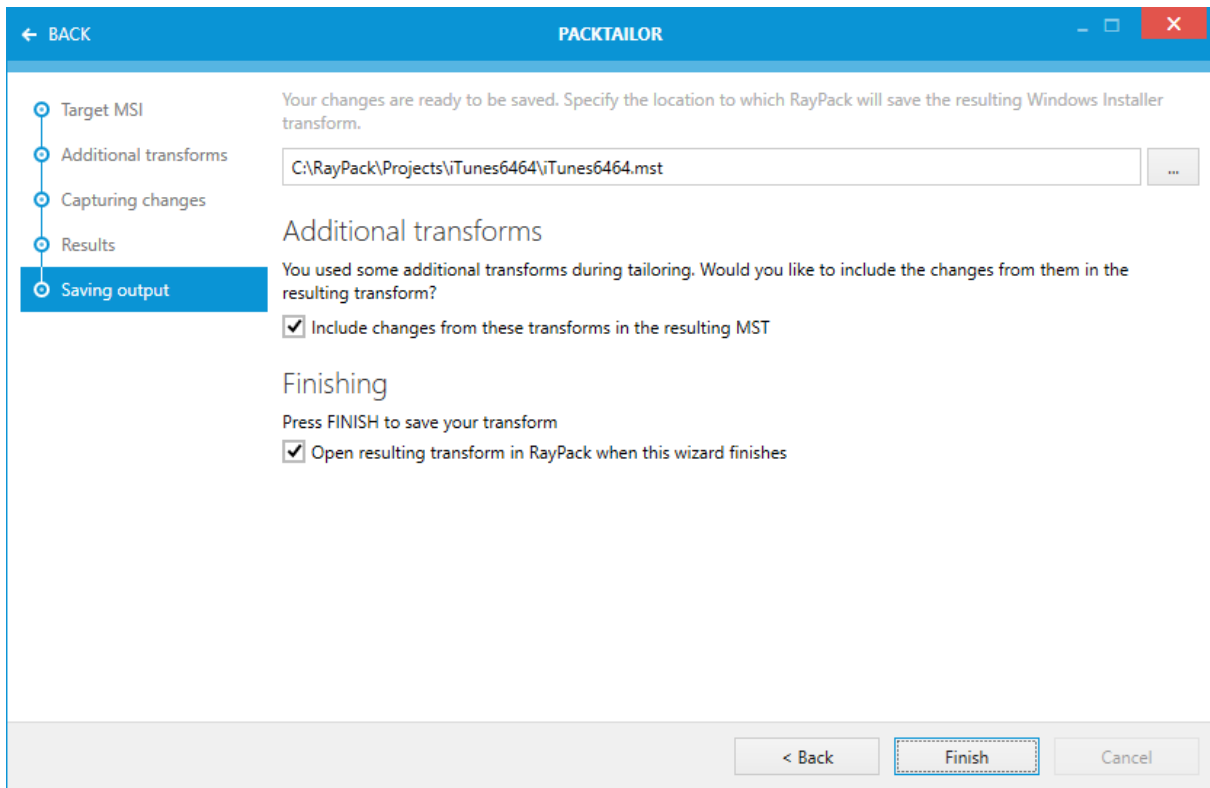
The changes to the **FEATURES** and / or changes to **PROPERTIES** can be viewed to see if they are satisfactory or the user can choose the **Next >** button to *continue*. Choosing **< Back** will return to the *previous page*, and choosing **Cancel** will return to the **Home Screen**.

> **Note:**
> Only certain folder changes are scanned by PackTailor. `INSTALLDIR` and target directories of MSI features are examples of properties that are included in the scan.

Saving Output

This page allows the user to choose the name and location of the transform that is to be created.

Additional Transforms

This dialog section is only displayed if additional transforms have been applied manually to the tailoring process. Activate the checkbox to combine the added transform files with the newly created transform rules to one complete transform. If the checkbox is not active, PackTailor creates a transform that includes the changes recognized during the wizard run, based on the combination of vendor MSI and manually added transforms.

Finishing

When choosing the location and name of the transform file, the `.mst` extension is automatically added. There will be the option of opening the transform that is created automatically in the PackDesigner once the PackTailor wizard has completed (activating the **Open resulting transform in RayPack when this wizard finishes** checkbox). Choose the **Finish** button to complete the wizard, and PackTailor will generate the transform file. Choosing the **< Back** button will return to the previous page.

Be aware:

If the MST template option has been activated, the result MST file will contain the template changes next to the matches from the response transform. This behavior is automatic and cannot be disabled directly from the *PackTailor* wizard. To change the template settings, go to the Settings > Projects screen.

PackDesigner

PackDesigner enables IT professionals to create, manipulate, and validate software packages. In the scope of this release (6.5) the project editor version for Windows target operating systems is enhanced. The PackDesigner component provides two different modes for editing packaging projects, the Visual Designer Mode the Advanced Mode.

The PackDesigner is the tool which enables the editing of the information stored within MSI, MST, or RPP files. The editor can be started by opening a RayPack Packaging Project (`*.rpp`) file via the Home Screen, or if RayPack has been installed locally, by double-clicking any RPP file.

The PackDesigner allows the user to edit the various files, folders, shortcuts, registry entries and package information before generating the required traditional or virtual target package format. This section describes in detail the various options and parameters that can be edited and how to generate a package.

When an MSI package or a combination of MSI and MST is opened in the PackDesigner, the contents of these target packages may directly be edited.

MSI / MST / RPP Based Projects

This sections deals with editing and creating projects and packages based on the Windows Installer Technology or Raynet's own RPP project format. The component to handle these package / project types is PackDesigner. According to the level of experience and individual task requirements, users have two view modes for accessing data and performing manipulations:

- Visual Designer Mode
- Advanced Mode

Visual Designer Mode

Inspired by the many years of experience in the packaging business, Raynet has developed an innovative editor concept, which goes far beyond plain value manipulation functionality.
Its unique user interface provides a variety of improvements to ease the access, the overview, and the handling of data. Visual Designer, embedded into RayPack's flat Windows 8 inspired style, uses a whole set of unified presentation and control elements to support users with an elaborate, responsive, and comfortable interface.

Preparing project / package properties for building MSI, MST, or MSP files becomes a convenient click-through experience. The Visual Designer serves an astonishing intuitive mixture of guided wizards, dynamic dialogues and best practice instructions. Whenever users create or edit a Windows Installer based project, the Visual Designer is their default working environment.

Advanced Mode

For all IT professionals who prefer direct database manipulation, there is the Advanced Mode. The integrated table editor is not only the most swift and direct access method to the core of any MSI related package / project, but offers some likewise innovative and handy features for a maximizing user experience, convenience and database integrity.

Additional advanced tasks, such as creating custom actions, managing the package sequences or adjusting upgrade related package properties, are also available in the views combined in PackDesigner's Advanced Mode.

Creating New Projects

MSI / RPP

New empty MSI/RPP files can be created by accessing the FILE > New menu. Further details are described in the section Create a New Project.

Transform (.mst)

Tip:
In order to create a response transform, use PackTailor as described here: PackTailor

---

There is no direct option to create a new blank transform (this is not allowed by the Windows Installer technology). Instead, follow these steps to create a new transform:

1. Open the MSI file to transform.

2. Do the required changes (or just a part of them, important is to apply all changes before the transform is generated).

3. Press **FILE > Save** as and select **Windows Installer transform** as a target format

> **Be aware:**
> Do not press **Save** button, because until the transform is created it saves changes back to the MSI file.

4. Select where to save the transform and confirm the selected path by pressing **Save** in the **Save File** dialog.

Other Formats

Patch (.msp)

There is no direct option to create a new patch file. The patch must be always built based upon an already existing MSI or pair of MSIs.

In order to create a patch, simply open the updated version of the MSI and follow the steps as described in chapter Building Microsoft Patches.

Virtual Packages (App-V, ThinApp, SWV)

There is no direct option to create a new virtual package. A virtual package can be built based upon an already existing MSI.

In order to create a virtual package, simply open the required MSI, RPP or MST file, and follow the steps as described in chapter Building Packages.

Visual Designer Mode

The Visual Designer mode editor interface is divided into the following main sections:

- **The Main Toolbar** at the top of the view
  The buttons and tabs provided by the Main Toolbar allow swift access to core functionality of RayPack, such as opening the settings area, or building packages from the currently opened project. Please refer to the Home Screen topic for further details.

- **The tree-view** on the left hand side of the window
  This is the internal editor view navigation. Each content type of project files that may be manipulated is represented by one of the items in this navigation tree. In order to maintain clear structures, there is a **Your project** overview present at the topmost position of the tree. There are additional group items to provide access to views that are closely related to each other. Simply click on one of the items to load the information or manipulation interface into the details pane at the right-hand side of the application window.

- **The information view** (also called details pane) on the right-hand side

This area contains information and allows the editing of the properties shown. Its contents are dependent on the entry selected from the tree-view navigation column on the left hand side of the PackRecorder Editor. Whenever a project is opened in the editor, the first page that is displayed within the details pane is the **Project overview**.

- **The action bar** at the bottom of the application window
  This area does not only provide the buttons required to switch between the Visual Designer and Advanced Mode.

  The action bar comes along in two display states: Expanded and collapsed. In the collapsed status, there are only button icons while the expanded mode displays additional text information about the functionality related to each button provided within the action bar. To switch between the expanded and collapsed display states, users have to click on the dot icons at the right-hand side of the action bar.

Read on to get details on the specific views of the PackDesigner Editor in the Visual Designer Mode.

General

The General view allows you to make changes to the Application and Summary information data, as well as package prerequisites and built and bootstrapping options.

> **Note:**
> If any of the items listed above is not available within your RayPack installation, check for the actual set of features covered by your license. To do so, visit the About section and expand the license details with the full listing of licensing options.

Project Overview

This view provides a summary of the contents of the currently opened project or package file.

Project Overview
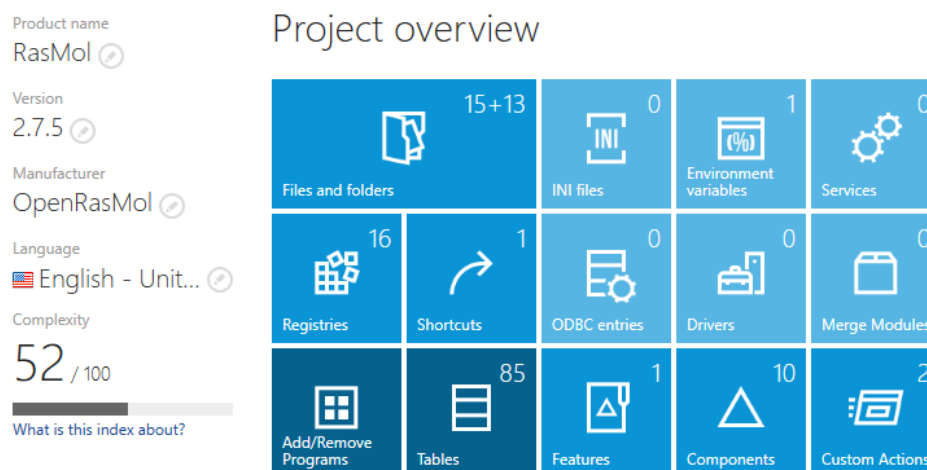
Each tile displayed within the **Project overview** symbolizes a specific package content type, such as files, folders, or add/remove programs information. Clicking a tile directly opens the management view required to manipulate the project values regarding the content type.

Where applicable, the tiles additionally indicate the number of objects currently stored within the project file for a specific type of content. Grey tiles represent content types which are not part of the project, while colored tile backgrounds indicate active content types.

> **Note:**
> Creating a new, empty project leads to the preparation of a project file that already includes some standard content objects, such as default system folders or Installer database tables. Which specific objects are injected on generation is determined by the default template for packaging projects. View the help section regarding template files for more information.

The modifiable properties within this area are **Product name**, **Manufacturer**, and **Version**. Please refer to the section about the Direct Value Editor for an explanation of how to manipulate these properties.

Complexity

Here we discuss the complexity of the package/project. This number is calculated based on a heuristic algorithm, which takes the various characteristics of the package/product into account (structure of the Features Components, properties of the components, custom logic and actions, etc.). The higher the number, the more complex a package/project is. The number is normalized

in the range between 0 (easy) and 100 (very complex).

Packagers benefit from the Complexity information by being able to estimate the error-proneness of required project manipulations due to inter-dependencies or simply due to large amounts of objects which need to be managed whilst working towards the desired result package.

Generate Report

This function allows the user to generate a report of the contents of the package/project. One may select which items are to be included in the report, as well as the file format (PDF, HTML or DOCX) of the report.
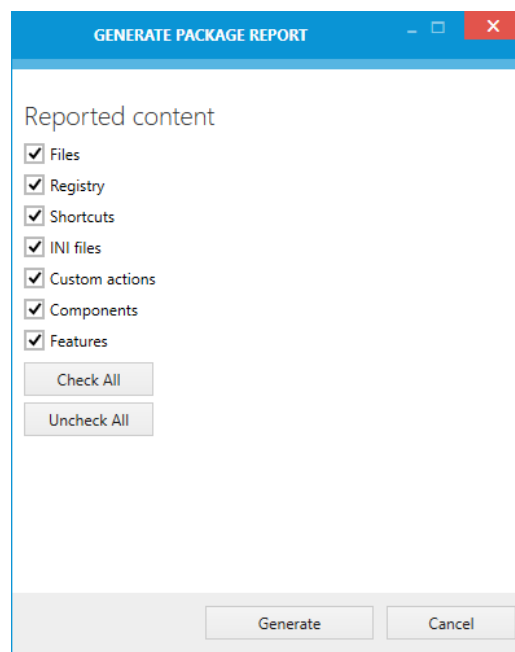
Follow the steps given below to generate a package report file:

1. From the **Your project** view, click the **Generate Report** button
   The Generate Report Dialog opens

2. Click the **Browse** button to select the target destination and format of the exported file.
   Please ensure that sufficient permissions are available to save the report in the chosen location.

3. **Confirm** those settings with the **OK** button displayed within the system browser dialog.

4. Select the desired **report contents** by activating the **Checkbox** at the left-hand side of the content type title.
   All content types are optional and can be combined arbitrarily.

5. Click the **Generate** button at the bottom of the Generate Report Dialog to start the export

process.
A progress indicator dialog is visible while RayPack is generating the report file.
As soon as the progress dialog closes, the report is available at the location specified at the beginning.

> **Be aware:**
> Depending on the performance of the packaging machine, exporting full report files from large packages might take a while.

Application

The Application view deals with information regarding the target package which is visible after program installation. Packagers manipulate the following basic properties within this view:
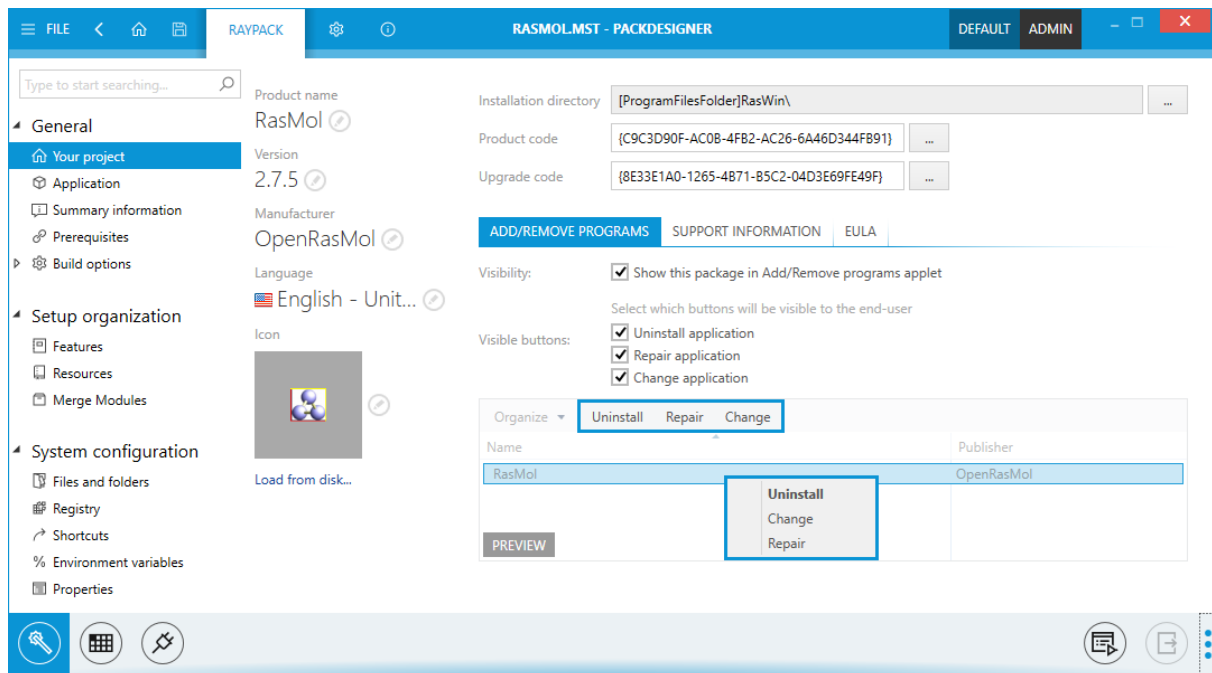
Common Application Properties

Product name

The name of the application is entered here and stored within the Installer database table `Property`. The product name must be given as a non-empty value.

Manufacturer

The name of the manufacturer of the application is entered here and stored within the Installer database table `Property`. The manufacturer must be given as a non-empty value.

Version

The version of the application is a string which is stored as **ProductVersion** within the `Property` table of the Installer database. It is recommended to use the standard notation and domain for the value:

```
[major 0-255].[minor 0-255].[build 0-65535]
```

See http://msdn.microsoft.com/en-us/library/aa370859%28v=vs.85%29.aspx for more information.

Language

The value entered here is stored in the Installer database table `Property` as `ProductLanguage`. Editing the language is fairly easy, as RayPack offers a predefined list of available language settings with checkboxes for one-click activation.

The default setting for the property `ProductLanguage` itself is 1033, which is the decimal declaration for English. The `LANGID` stored as internal property value specifies the language the installer should use for any strings in the user interface that are not authored into the database. When authoring a package as language-neutral, the `ProductLanguage` value is to 0.

See http://msdn.microsoft.com/en-us/library/aa369771%28v=vs.85%29.aspx for an overview of `LANGID`s assigned by Microsoft.

Icon

Each package may be visually augmented with an **icon**. Packagers may either pick one of the already existing icon resources from the current packaging project or add a new icon by loading it from the local disk. If an icon has been added, a preview is shown as visible icon in the collapsed icon resource selector interface.

To pick an existing icon resource:
1. When users expand the icon resource selector interface with a click on the downwards arrow icon next to the standard icon.
   The list of icon resources stored within the packaging project is displayed.

2. Select the desired icon from the list on the left-hand side of the selector interface, and - if required - navigate to the icon index that should be used. Use the arrows pointing to the left and to the right to change the currently active icon index value.

3. Click somewhere outside the icon resource selector interface to save the current setting for the package icon.

To load a new icon resource from disk:
1. When users click on the **load from disk** link, below the preview panel for the current package icon state is displayed.

2. Within the system browser for icon selection, the desired icon resource has to be selected.

3. By clicking on the **Open** button, the icon is loaded into the project and selected to be the icon for the package.

> **Be aware:**
> When the package icon is changed, the formerly used icon resource is not removed from the packaging project. Nonetheless, when a new icon is loaded into the project, a new, permanent icon resource is added to the packaging project. Please make sure to manage icon resources via the controls of the Resources view if required beyond the

needs of the current package object manipulation.

Installation directory

The path to the installation directory determines where the application will reside once it is installed on the target device. The default value of this Installer property is `[ProgramFilesFolder]ProductName\`.
`[ProgramFilesFolder]` is a predefined folder wild-card and will be resolved at runtime to match the actual target device Program Files Folder, which depends on the operating system running on that device. Take a look at the Predefined folders section within the Files and folders view chapter for more information.

In order to keep the later target package as generic as possible, it is highly recommended not to use an absolute path value (such as `C:\Program Files (x86)\RayPack\`) as Installation directory.
The value for the installation directory has to be changed by using the Select a folder dialog; manual input is not supported in this view. To manually edit this value, use the `Directory` table within the Advanced mode view Tables.

Product code

The product code is a **GUID** that is the principal identification of an application or product. It is stored within the Installer table `Property`. If significant changes are made to a product, then the product code should also be changed to reflect this. The value must vary for different versions and languages of a product. Additionally, 32-bit and 64-bit versions of an application's package must be assigned different product codes.

Each product code has to be unique. Therefore, even though it might be possible to manually enter a product code value, it is recommended to use the automatic generation utility by clicking on the button right of the input field. Thanks to the algorithm used by RayPack, generated product code **GUID**s are unique.

The valid format for a **GUID** is `{XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX}` where `X` is a hex digit (`0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F`). Letters in product code **GUID**s must be uppercase.

Upgrade code

The upgrade code is a **GUID** that is required to manage upgrade paths between different versions of the same software application. The formal restrictions are the same as for the product code, while the upgrade code is per definition, not a unique value. Please refer to the Upgrade section for further details about upgrading and upgrade code handling.

Add/remove programs tab

Visibility

Activating this checkbox allows users to see the application within the ARP list of installed

programs. Applications are visible by default. Internally, the visibility status is stored as **ARPSYSTEMCOMPONENT** within the `Property` table of the Installer database. Deactivating the visibility checkbox is equal to setting the property value to 1.

Visible buttons

If an application is visible within the ARP section at all, the available options for user interaction can be scaled. The available settings can be manipulated independently:

- **Uninstall**
  Users may uninstall the program from the device.
  The value is stored as **ARPNOREMOVE** within the Installer database `Property` table.
- **Repair**
  Users may repair damaged installations.
  The value is stored as **ARPNOREPAIR** within the Installer database `Property` table.
- **Change**
  Users may change the set of installed features for a program.
  The value is stored as **ARPNOREMODIFY** within the Installer database `Property` table.

Below the controls for visibility and visible buttons, a preview area shows how the ARP section on the target device will look like for the product according to the current settings. If the application is invisible within the ARP section, the preview section is not visible as well.

Support information tab

The given support information is displayed when a user selects the installed program within the ARP tool. It is intended to help the user if questions regarding updates or support needs arise. Further details regarding ARP control via Windows Installer is available at http://msdn.microsoft.com/en-us/library/aa368032%28v=vs.85%29.aspx.

Product update URL

The link is stored as **ARPURLUPDATEINFO** within the Installer `Property` table.

Contact person

This value is stored as **ARPCONTACT** within the Installer `Property` table.

Phone number

This value is stored as **ARPHELPTELEPHONE** within the Installer `Property` table.

Help URL

The link is stored as **ARPHELPLINK** within the Installer `Property` table.

Comments

This value is stored as **ARPCOMMENTS** within the Installer `Property` table.

EULA tab

This tab contains information about the End User License Agreement. The content of the EULA can be exported to an `.rtf` file, or imported from `.rtf` file.

> **Note:**
> The visibility of this tab depends on whether the current project contains a Memo control that contains a license text. See User Interface section for more details.

Summary Information

The package view mainly provides controls for the manipulation of data stored within the Summary Information Stream of an Installer package.

Basic information

These information are visible in properties of the Windows Installer package

Title: Installation Database

Author: Raynet GmbH

Subject: RayPack

Comments: Raynet GmbH

Keywords: Installer,MSI,Database,RayPack

Language: English - United States

Code page: 1252

Requirements

Modyfing these values will affect on which systems the package may be installed.

Target platform: 32-bit   Installer version: 3.0 (schema: 300)

Privileges:
- ⦿ Elevated privileges can be required to install this package
- ◯ Elevated privileges are not required to install this package

This setting is available starting with Windows Installer version 4.0 and Windows Vista or Windows Server 2008.

Package identification

Package code: {7BA33BAF-548B-40E7-90B0-3838A9BB5F18}

The package code is a GUID identifying a particular Windows Installer package. The package code associates an .msi file with an application or product and can also be used for the verification of sources. Nonidentical .msi files should not have the same package code.

Source image

Sources compression: Compressed   ☐ Use short names

INFO These settings are read-only. If you want to make changes to the compression of the files, please rebuild the database.

Common Summary Information Stream Properties

Title

Usually this value is used to specify the type of Installer database, which might be one of "Installer database", "Patch", "Transform", or the like.

Subject

The name of the product which is installed by the package – most likely identical to the **ProductName** as given within the Installer database Property table.

Author

The name of the package manufacturer – most likely identical to the **Manufacturer** as given within the Installer database Property table.

Keywords

A comma separated list of phrases that are used for keyword searches, e. g., file browsers. It should contain both hints to the file type (e. g., "Installer"), as well as, to properties specific to the application installed by the package (e. g., "image processing", or "database management system").

Language

Editing the language is fairly easy, as RayPack offers a pre-defined list of available language settings with checkboxes for one-click activation. Click on the edit button at the right-hand side of the current language value and select the required new option. The summary information stream may contain several language definitions and it is therefore possible to activate multiple languages in the selector dialog.

Code Page

The numeric value of the ANSI code page used to display the Summary Information. The default value used by RayPack projects is 1252, as it covers English and most Western European language requirements.

Comments

The comment is a formalized phrase, which is recommended to be "This installer database contains the logic and data required to install `[ProductName]`." for Installer databases.

Requirements

Target Platform

Target platform and language are combined to the **Template** Summary stored within the summary information stream. Platform and language ID have to be separated by a semicolon. If more than one language ID is given, then this list has to be added with comma separated:

```
[platform property];[language id][,language id][,...]
```

The default platform setting for RayPack packaging projects is "32bit", which is translated to "Intel" within the original stream property value.

	Note:
📎	Predefined folder browser uses this value to determine whether 64-bit folders (for example `ProgramFiles64Folder`) are shown.

Installer Version

The default Installer version for RayPack packaging projects is 2.0, which resembles a value of 200 stored as **Page Count** property within the Summary Information stream. It declares which minimum installer version is required by an installation package.

Privileges

This property allows to determine whether or not the package requires elevated rights during installation. From a technical perspective, choosing that no elevated rights are required sets bit 3 of the word count property stored within the summary information stream.

	Note:
📎	Please note that this option is not supported by all Installer versions and operating systems, and refer to MSDN for further details.

Package Identification

Package Code

The package code is a **GUID** identifying a particular Windows Installer package. The value stored within the Revision Number property of the Summary Information Stream associates an .msi file with an application or product. It can also be used for the verification of sources.

Each package code has to be unique. Therefore, even though it is possible to manually enter a package code value, it is recommended to use the automatic generation utility by clicking on the button right of the input field. Thanks to the algorithm used by RayPack, generated package code GUIDs are unique.

The valid format for a GUID is {XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX} where X is a hex digit (`0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F`). Letters in package code GUIDs must be uppercase.

> **Note:**
> The Advanced section is only available for MSI / MST packages. It is not displayed when an `.rpp` project is opened for manipulation.

Some advanced Summary Information Stream properties, such as the bitwise defined **Word Count** setting, are not editable via the Visual Designer interface. RayPack uses default values for this property, which are available for (read only) reference within this view.

More details regarding the Word Count property are available online: http://msdn.microsoft.com/en-us/library/aa372870%28v=vs.85%29.aspx.

Prerequisites

The *Prerequisites* view provides controls for the manipulation of prerequisites and dependencies of the currently edited project. The difference between Merge Modules and the prerequisites is that Merge Modules are deployed as `.msm` files which are actually merged into the MSI project, while dependencies / prerequisites are usually distributed as standalone installations (for example VC++ Redistributables are provided in form of an executable installer).

Manufacturer	Name	Version
Microsoft Corporation	Visual C++ 2010 Runtime Libraries (64-bit)	10.0.30319.1
Microsoft Corporation	Visual C++ 2010 Runtime Libraries (32-bit)	10.0.30319.1
Microsoft Corporation	Visual C++ 2012 Runtime Libraries (64-bit)	11.0.50727.1
Microsoft Corporation	Visual C++ 2012 Runtime Libraries (32-bit)	11.0.50727.1

This view displays the list of items that were identified as prerequisites of the current project. Because prerequisites are stand-alone, in order to prepare a package that actually installs necessary piece of software before the main installation it is necessary to BUILD the project.

> **Note:**
> Prerequisites will not be bundled with the MSI package unless the MSI is rebuilt. Pressing FILE > SAVE as does not trigger the bootstrapper build, and therefore creates no wrapper to install them.

> **Be aware:**
> The list does not show nested dependencies. If any prerequisite has its own prerequisites, it will be resolved and compiled together with nested resources at build-time.

Removing a Prerequisite from the Package

To remove a prerequisite from the list, highlight the item to be removed, and from its context menu press *Delete*. The prerequisite will be removed automatically.

Reordering Prerequisites

When the package is installed from the bootstrapper (see the *Built options sections*) the prerequisites are installed in the order as displayed in the *Prerequisites* view. The higher a prerequisite is on the list the sooner it gets executed. In order to reorder the items, highlighted the prerequisite to be moved, and then using the drag and drop technique drag it to the desired place.

Adding Prerequisites

Click on the button *Add prerequisite...* to show the predefined prerequisites browser. See section *Adding prerequisite* for more information or *Defining prerequisites for bootstrapper* for a reference of prerequisite syntax.

Adding Prerequisites

The predefined prerequisites browser shows the list of defined prerequisites.
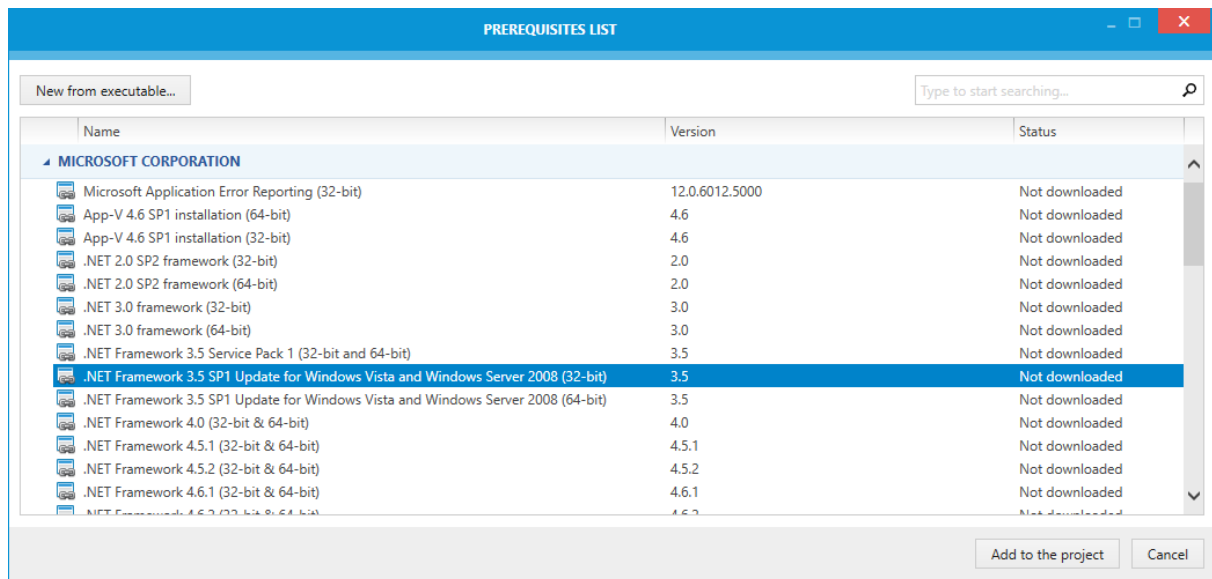
The list is grouped by the manufacturer name, and shows the name of the prerequisite, its version and the status:

- **Available**
  Denotes that the prerequisite is defined and its sources have been already downloaded to the PackPoint location.

- **Not downloaded**
  Denotes that the prerequisite is defined but its sources are either missing, are not valid or require to be re-downloaded.

The list can be filtered and sorted. The selected prerequisites can be added to the project by pressing the *Add to the project* button.

Adding a New Prerequisite from a Local Drive

To add a new prerequisite from an executable file available on local drive, click on *New from executable...* A dialog will open, prompting for a path to the executable file to be used as a prerequisite. Select the required resource and press *Open* to prepare the prerequisite definition.

> **Note:**
> The selected executable will be copied to PackPoint location into *Prerequisites* subdirectory. Basic properties (name, version and manufacturer) are automatically extracted from the selected executable and used as prerequisites's metadata.

Downloading Prerequisites

To keep the RayPack installer size relatively small and to comply with individual license terms, the default installation of RayPack does not contain installation sources of any prerequisites. Each item delivered out-of-the-box contains only meta-data information informing RayPack i.a. from where the necessary sources can be downloaded.

When a prerequisite is to be added to the project, RayPack checks whether the necessary files are present in PackPoint. In case any resource is missing or is incomplete, it will be downloaded from the vendor website.
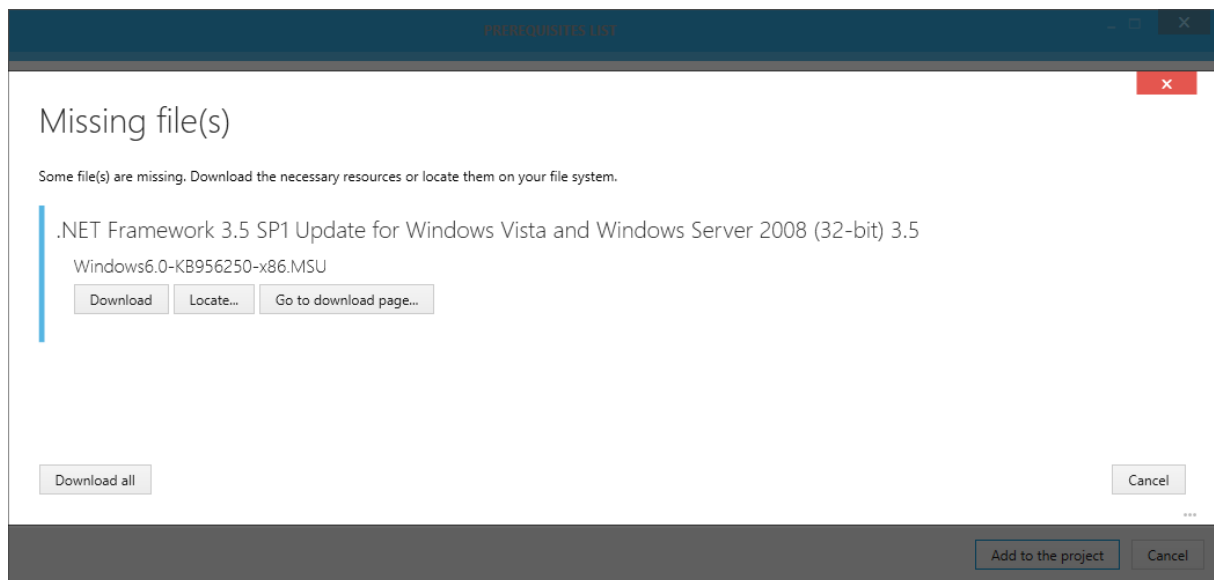
The window stays on-screen until all necessary prerequisites are downloaded, or until the operation is canceled by pressing the *Cancel* button.

Click **Download** to start the download procedure from vendor's website. Depending on the speed of the internet connection, it make take a few moments before all necessary resources are downloaded.

Click **Locate...** to manually select the required resource from your local drive.

Click **Go to download page...** to open a download page of selected resource.

> 📎 **Note:**
> Some buttons may be disabled for prerequisites defining no download link or vendor webpage.

Click **Cancel** to cancel and return back to the list.

Click **Download all** to automatically download all prerequisites. This option is active only if at least one download link is available. Prerequisites without download links have to be resolved manually.

> 📎 **Note:**
> The resources located using either **Locate...** or **Go to download page...** functions are always copied to PackPoint. This is to ensure that the PackPoint library is internally consistent and working for all users. Specifying links to non-standard locations is only available via manual edit of prerequisite definitions.

Nested Prerequisites

If selected prerequisites have nested prerequisite dependencies, the *Missing file(s)* screen will contain all required resources resolved by recursive traversing of prerequisite definitions. For example, when adding *Visual C++ Redistributables,* RayPack prompts for *Windows Installer* package as well. After they are downloaded, only the selected prerequisites will be visible in the *Prerequisites* screen. Their dependencies (once downloaded) will be automatically resolved and compiled during building.

Downloading Prerequisites on Demand

Prerequisites can be also downloaded on demand, without adding them to the project. In order to download a prerequisite:

1. Select the item(s) to be downloaded
2. Right click to open a context menu
3. Press **Download** to start the download operation

Once a prerequisite is downloaded, it will never prompt for a download again, unless its resources are deleted.

Build Options

This view shows the configuration build settings for creating specific package types. The settings are applied each time the Build functionality from the File menu is accessed.

Please use the tab labels to open the options view for the required target package type:

- Windows Installer for MSI files
- Patch for MSP files
- Thin-App and App-V for virtual target package formats

Additionally, **Build Options** provide a subview to access internal path variables, used when building files and streams.

Path Variables

This view shows internal path variables, used by RayPack to:

- resolve source files when building MSI files from RPP projects.
- resolve newly added source files when saving current MSI/MST projects.
- resolve source streams (binary and icon files) if defined so in the Resources view.

To Add a New Variable

1. Click **Add...** button to create a new entry
2. Right click the new entry and select **Rename** option
   - Optionally, press **F2** or simply double click to start editing.
3. Enter a new name for the variable
4. Double click the axiomatically generated value and enter the required path.

	Note:
🖇	Name of the variable must adhere to the following requirements: • It must contain at least one character • Only latin letters, digits and underscore character are allowed • The name must start with a letter or underscore

To Remove a Variable

1. Select the property to be removed
2. Click **Remove selected** to remove it
    - Optionally, press the **Delete** button to delete the currently highlighted row.

To Rename or Change a Variable

1. Select the property to be edited
2. Press **F2** to start editing the current cell
    - Optionally, double click the name or the value to start editing it.

> **Note:**
> Name of the variable must adhere to the following requirements:
> - It must contain at least one character
> - Only latin letters, digits, and underscore characters are allowed
> - The name must start with a letter or an underscore

Windows Installer

The Build options view tab windows installer contains all options for the build process of MSI files:

> **Tip:**
> The screen is available for both RPP projects and also for any MSI opened from your hard drive (including vendor installations and admin installations). Therefore, a vendor MSI may be easily recompiled and their compression layout adjusted to your particular needs.

The settings available in the *Build options* screen are identical to the ones present in the settings screen. By default, when a new project is created the values here will reflect the combination of two sources:

1. The profile settings
2. The template settings

Changing the options in the *Build options* screen does not affect any of the areas previously mentioned. Depending on the type of project being edited, the settings may be persisted and stored per-project (RPP files) or active for the current session (MSI / MST files).

Patch

The Build options view tab PATCH contains all options for the build process of MSP files:

Patch Properties

Display name
The name of the patch as it will be displayed in Windows system property dialogs.

Description
The description of the patch as it will be displayed in Windows system property dialogs.

Allow the patch to be removed
If the option is selected, a patch will be removable by user from the Add/Remove programs panel. The actual availability of the removal option depends on several factors (see the Patching section).

Always include whole files
By default, only a binary difference between patched files will be included in the resulting MSP file. By using this option it is possible to override this behavior. Enabling this option will speed up the patching process, but may also increase the size of the resulting MSP file.

Affected Products

This setting can be used to specify the range of valid product codes that will be targeted by the resulting patch. By default (when the **Automatically determine the affected ProductCodes** option is selected) RayPack will determine the product code from the base patched image. To specify a custom range of affected product codes, the **Patch the following ProductCodes** option has to be used. If you want to automatically include the `ProductCode` of the base image, specify asterisk (`*`) as a ProductCode plus all other required `ProductCodes` if necessary.

ThinApp

This tab defines specification of the conversion process. For information about available Thin-App settings, refer to the *Conversion* chapter (section THIN-APP).

The settings available in the **Build options** screen are identical to the ones present in the **settings** screen. By default, when a new project is created the values here will reflect the combination of two sources:

1. The profile settings
2. The template settings

Changing the options in the **Build options** screen does not affect any of the areas previously mentioned. Depending on the type of project being edited, the settings may be persisted and stored per-project (RPP files) or active for the current session (MSI / MST files).

App-V 4.6 / 5.X

These tabs define specification of the conversion process. For information about available App-V 4.6 and App-V 5.0 settings, refer to the *Configuring App-V conversion* chapter.

The settings available in the **Build options** screen are identical to the ones present in the settings screen. By default, when a new project is created the values here will reflect the combination of two sources:

1. The profile settings
2. The template settings

Changing the options in the **Build options** screen does not affect any of the areas previously mentioned. Depending on the type of project being edited, the settings may be persisted and stored per-project (RPP files) or active for the current session (MSI / MST files).

Setup Organization

The Setup organization view allows you to make changes to the Features, Resources, and Merge module settings.

> **Note:**
> If any of the items listed above is not available within your RayPack installation, check for the actual set of features covered by your license. To do so, visit the About section and expand the license details with the full listing of licensing options.

Features

RayPack includes two views for Feature manipulation. The Features view within the Visual Designer mode allows to manage feature settings as long as they are affecting the user interaction during installation runtime. More complex background settings, such as component management are manipulated via the Advanced mode.
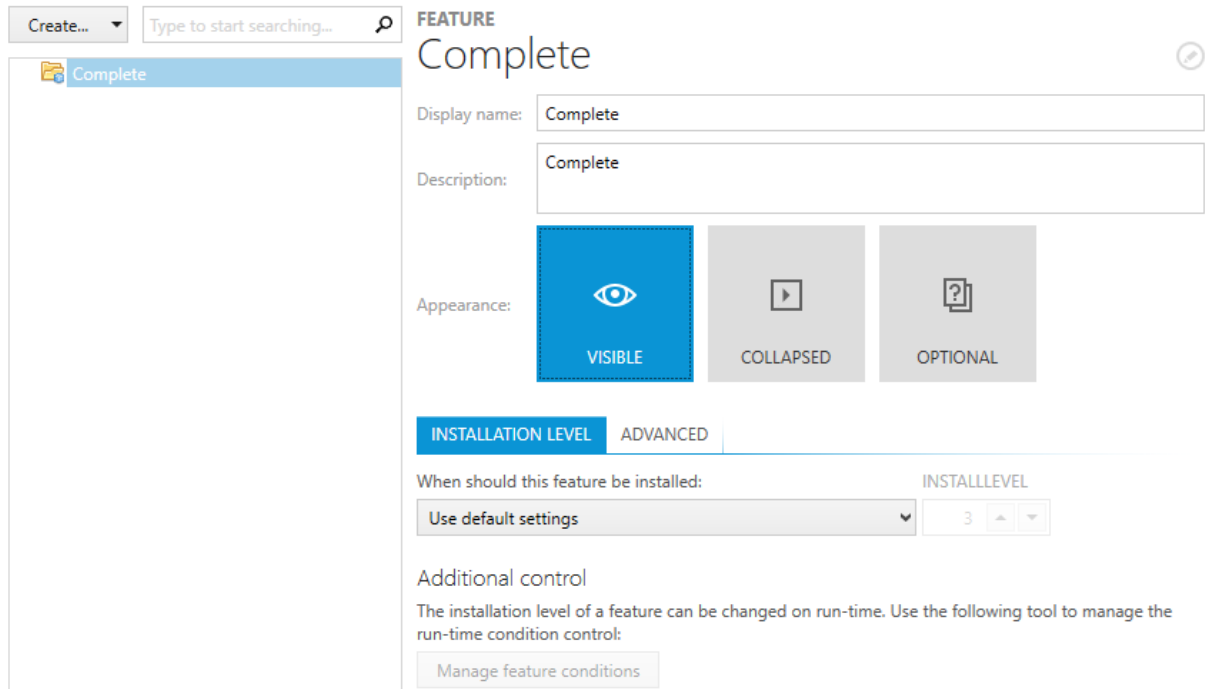
The main view of the Visual Designer area Features consists of the actual features and their nesting on the left. The secondary view to the right contains information about the currently selected item and allows actions to be carried out on it.

The Feature Item Context Menu

A right-click on a feature from the tree view on the left allows to directly **rename** or **delete** it by selecting the respective option from the context menu. Additionally, users may initiate the **extension with a new child feature** from the context menu.

By selecting **Go to row** there is the option to call the Installer database row within the Feature table, which actually references the currently focused feature. By doing so, the Visual Designer mode is left and the Table view of the Advanced mode is loaded with the Feature table visible inside the content area.

Since feature trees may become quite substantial, the context menu also contains options to **collapse** and **expand** all child nodes of the currently right-clicked feature item.

Feature Nesting

A valid Installer package has to contain at least one (root) feature. Additional features may be

added either at the same level as the first one or added as a child feature to an existing one. The tree structure of features is arranged alphabetically by the property "Feature", which is the displayed label of a feature within the tree view. However, it is always possible to change the default order by simply applying drag and drop and moving features (with all their child nodes) to any other location within the tree structure.

Core functionality within the Features view:

- Add a root feature
- Add a child feature
- Rename a feature
- Edit a feature
- Remove a feature

Add a Feature

Add a Root Feature

To add a new root feature, users should click on the **Create** button above the tree view of already added features, and select the option **New root feature** from the menu.

Add a Child Feature

To add a child feature to an already existing (root or child) feature, users should right-click the desired parent feature and select the option **New feature** from the context menu. Additionally, clicking the Insert key on the keyboard also adds a new sub-feature to the currently selected item.

New Feature Properties

Any new feature is generated automatically with the feature property value "New feature" and title property value "NewFeature". The feature column value of features has to be unique. Therefore, if another feature is already using the default value, a unique index extension is automatically added (e. g., "NewFeature_1" or "NewFeature_2").

The new feature is automatically added to the existing tree structure. Its position within the sort sequence of the desired level is determined by the value of the property "feature", which is displayed as the label of features within the tree structure. However, it is always possible to change the default order by simply applying drag and drop and moving features (with all their child nodes) to any other location within the tree structure.

All other properties of new features are set to the default values. This system behavior leads to visible features, which are always installed into the `INSTALLDIR`. To alter these defaults, users have to edit the new feature.

Rename a Feature

Renaming a feature is triggered by three different user actions:

- Selecting a feature from the tree view of features, and hitting **F2** on the keyboard activates the rename mode for the feature directly within the feature tree view.
- Selecting a feature from the tree view of features, right clicking it, and selecting **Rename** from the **context menu** activates the rename mode for the feature directly within the feature tree view.
- Selecting a feature from the tree view of features, followed by clicking on the **Feature** property displayed within the **details pane** on the right, opens the Direct Value editor interface for this property.

Once the value is changed as desired, hit the enter key. RayPack checks if the new value is valid. If so, it is saved for the current session. If not, the reaction depends on the method used for renaming.

If the renaming was attempted inside the tree view structure of Features, RayPack displays an error icon. Hover over the icon or the invalid value to get a hint regarding the reason for the invalid value. Change the value and hit enter again to re-check. Repeat this until a valid new Feature value is saved to the session, or hit escape to return to the value that was saved before the attempt to edit failed.

If the renaming was attempted inside the details pane, RayPack automatically corrects the invalid value. Trying to save the value "1Feature?" triggers the auto-correction to "_1Feature". Feature values may not begin with digits or contain special characters. RayPack knows about those rules, and supports packagers with a lower experience level to ensure the generation of valid target packages and at the same time reduce the amount of workload required to achieve that result.

Edit a Feature

Once a feature is selected from the tree view structure on the left hand side of the Feature view, the current settings are displayed within the details pane on the right. Since the properties are immediately displayed in a form view with decent control interfaces, there is no need to call an additional edit view to make changes to the settings. Users can change the displayed values and settings and simply change to another feature, view, or mode all while having the changes saved as part of the changes done during the current session.

The following feature properties are modifiable via the Feature view of the Visual Designer mode:

FEATURE

Complete

Display name: | Complete

Description: | Complete

Appearance:

| **VISIBLE** | **COLLAPSED** | **OPTIONAL** |
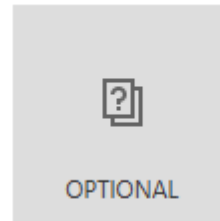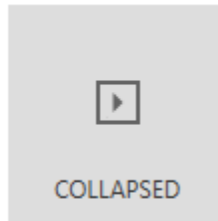
Feature

Affected `Feature` table column: Feature
The unique identifier for a feature.
See Rename a feature for details

Display Name

Affected `Feature` table column: Title
This is the feature label as it is displayed during package setup execution. If the setup is defined to allow users to actually see (or manipulate) the set of installed features, this is the value users see.

Description

Affected `Feature` table column: Description
This is the feature description as it is displayed during package setup execution. If the setup is defined to allow users to actually see (or manipulate) the set of installed features, this is the value users see when they pick the feature from the list of available features.
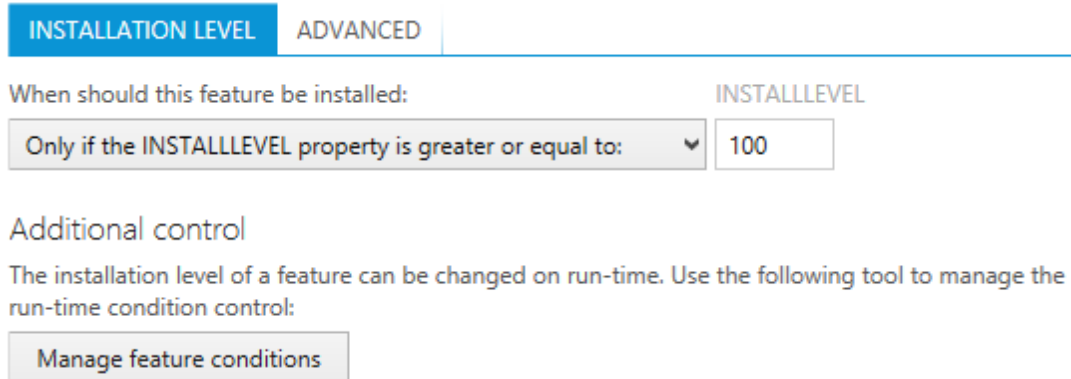
Appearance

Affected `Feature` table column: Display
The two basic decisions made here are, if the feature is required, which means the user cannot de-select it in any kind of custom setup configuration step, and if the feature is visible, which decides whether the user is aware of the existence of the feature as a functional unit of the installed application or not. Both settings can be made independent from each other. A third decision is only available for visible features: By activating the "This feature is expanded" checkbox, the custom installation step is loaded with the whole tree structure of child features visible. This is handy for those constellations where a specific feature has to be installed, but with a highly customizable set of optional sub selections, e. g., with a wide list of available, yet

optional language packs for an application.

Tab: INSTALLATION LEVEL

INSTALLLEVEL

Affected `Feature` table column: Level
By default, all features with an install level less than or equal to the value of the **INSTALLLEVEL** property of the package are installed during setup. This INSTALLLEVEL comparison is executed at runtime.

By selecting never or always, this default behavior can be replaced. It is also possible to enter a specific (integer) value as required INSTALLLEVEL.

Conditions

Affected Installer database table column: Condition
RayPack allows to install features conditionally. Clicking the **MANAGE FEATURE CONDITIONS** button opens the Condition table within the Advanced mode TABLES view for direct data row creation.

Tab: Advanced

Destination

Affected `Feature` table column: TargetPath
This value defines the directory into which the resources for the feature are installed during setup runtime. The default setting is the INSTALLDIR defined for the whole Installer package. To change that default, users should click on the button right of the destination path info display. Read the Select a folder section for details about the interface handling.

Advertised

Affected `Feature` table column: Attributes
Windows Installer can advertise the availability of an application without installing the application. If an application is advertised, only the interfaces that are required for loading and for starting the application are presented. If the advertised interface is started, Windows Installer installs the required components. This method is also referred to as installation-on-demand.

The default setting for RayPack packaging projects is to generally **allow advertisement** as an optional selectable functionality, but not to activate it from scratch. In order to mark a feature to be advertised by default, packagers set the Advertised value **Favor Advertise**.
Alternative settings are to **disallow advertisement** altogether or **prohibit advertisement on target os that do not support installation on demand**. The last two options take away the advertisement-decision from the installing end-user (no matter if it is a person or software deployment system)

The advertised setting is reflected as bit wise analyzed column value.

Remote Installation

Affected `Feature` table column: Attributes
Discussed here is how the Remote Installation property is a default value that defines how the feature comes up when the custom setup dialog is displayed for the first time, or how the feature will be installed if the custom setup dialog is not invoked.

If a feature is visible, the remote installation state of this feature can be changed by making a selection from the context menu option set within the custom installation feature tree.

However, if a component stored inside a feature has a different remote installation configuration than its parent feature, the component setting is dominant over the feature setting. Therefore, the remote installation options in the Features view are called *Favor* Local, *Favor* Source, etc.

Favor local initiates a copy of the feature sources to the features target directory on the target system, while **Favor source** installs a feature from the original `.msi` resource media (which might be stored on a remote hard drive, a network share, an USB stick or any other temporary or permanently available storage area).

The third option, **Same as parent,** is only available for child features and instructs the installer to apply the very same setting that is used for the direct parent feature.

Remove a Feature

> **Warning:**
> Deleting a feature is quite a rigorous measure with a high potential for extensive side effects. Make sure to double-check the relation and content status of a feature before actually deleting the logical feature object!

To trigger the feature deletion, packagers should select the feature in the tree view structure within the Features view. Once a feature is selected, its deletion is initiated either by hitting the **delete key** or by right-clicking the feature and picking **Remove** from the context menu.

Either way, packagers are by default requested to confirm the intention of the deletion. Even though the confirm dialog may be suppressed by activating the "In future do now show this confirmation for delete operations" checkbox, it is recommended to keep it for security reasons. Deleting a feature deletes the overall settings and logical relations between the contents managed by that feature. The contents themselves, such as files, registry keys, and the like, are preserved. In a worst case scenario the components of a deleted feature become orphaned. Those unmanaged components will never be installed. Windows Installer databases with orphaned components are not standard compliant, and will fail any attempt to install the target package.

Nonetheless, if a feature must be deleted, the confirm dialog offers a button **REMOVE** to do exactly that. Clicking that button deletes the feature and all child features, while preserving all components stored within the feature or its former children.

> **Note:**
> As soon as the feature object is deleted from the Feature view, its resembling data row marked as deleted within the affected Installer database table. Once the packaging project is saved or the history of changes is cleared, the feature object is no longer visible at all.
> Please refer to the Tables section to get more details about the object manipulation history and markup management.

Resources

The resources view allows users to manage the binary and icon resources files within a packaging project.

The main view of the Visual Designer area Resources consists of the actual list of resource files on the left. The secondary view to the right contains information about the currently selected item and provides buttons to trigger actions to be carried out on the resource item.
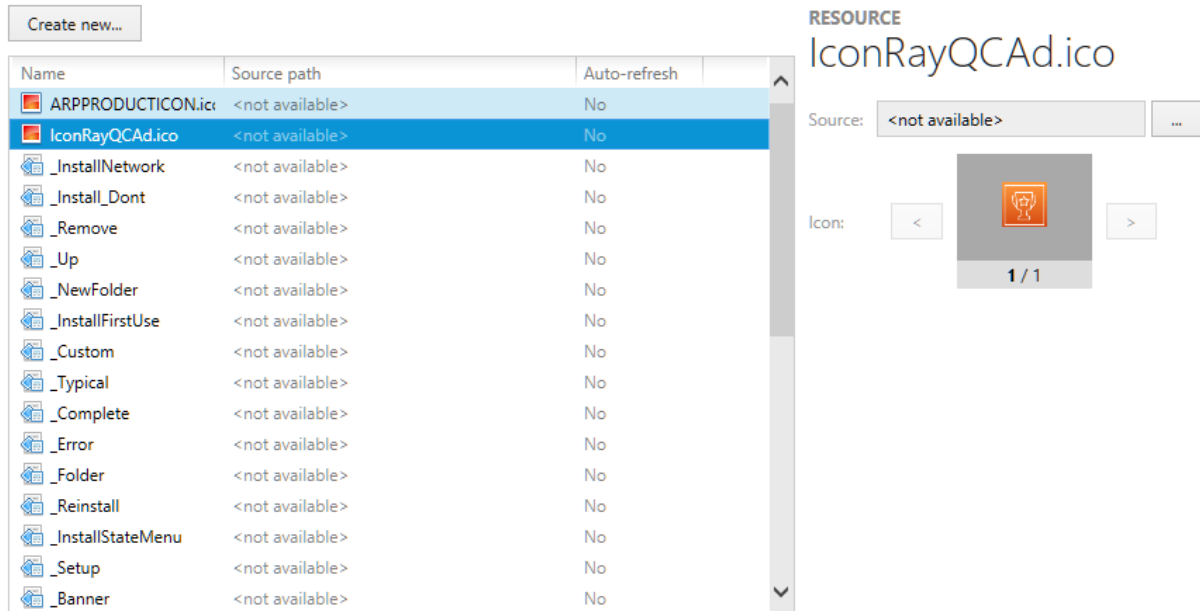
Resources managed here are mainly stored within the Installer database tables `Icon` or `Binary` and represent the physically existing binary files within a packaging project.

When a resource is added to an `.rpp` project, it is simultaneously added as IBD file to the sources folder which is saved along with the `.rpp` file. Depending on the resource type, the `.ibd` file resides in one of the sub-folders of `\[Project Name].rpp.Sources\Streams\`.

> **Note:**
> Depending on the origin of the RPP project contents, various sub-folders may have been added to the source streams. Especially when an MSI is transformed into an RPP collection, the non-standard resources of the original MSI tend to be organized in a custom set of organizational structures.

Core functionality within the Resources view:

- Add a resource
- Replace a resource
- Remove a resource
- Edit a resource

Add a Resource

Adding a resource to a packaging project is initiated from the Resources view. To add a new resource:

1. Click on the **Create new...** button above the list of already available resources.

   The **Wizard Selector Dialog** is displayed, requesting the user to declare what type of resource is intended to be added:

   - a new icon resource
     or
   - a new binary resource

> 📎 **Note:**
> Icon resources can be imported from `.ico`, `.exe`, or `.dll` files. All other file types are handled as standard binary resources.

To Add a New Icon Resource

1. Within the Wizard Selector Dialog: Select **New Icon Resource**
   The Icon Options Dialog is displayed.
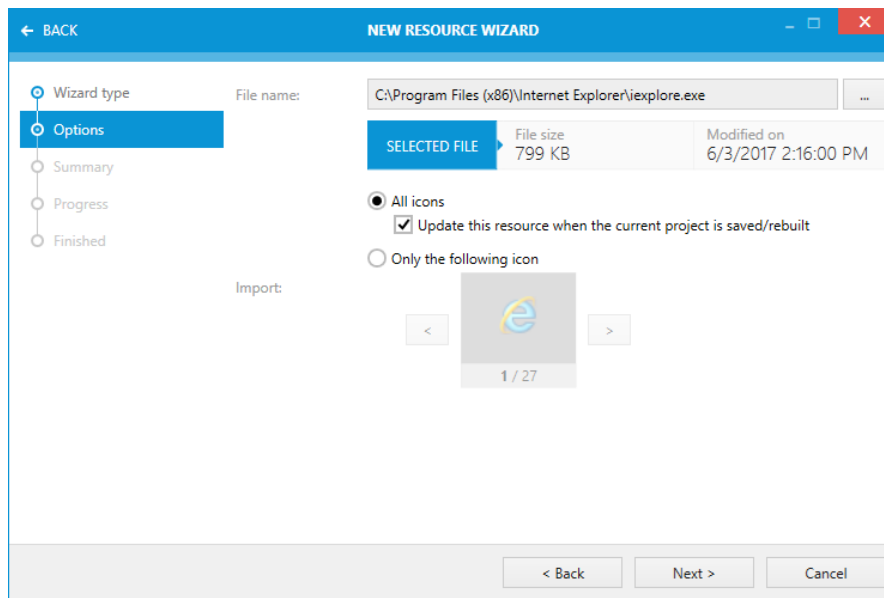
2. **Browse** the local system for the source file of the new icon resource
   Once the source file is selected, RayPack analyzes its content and automatically determines the **name**, **size**, **last update time** and **visual preview** for the new icon resource.

   These properties will be saved along with the resource and cannot be changed manually

inside the wizard. Nonetheless, it is possible to clear size and last update time by activating the checkbox **import the current icon only**. Please refer to Edit a resource for further details on how to change resource properties.

3. If the selected source is not an `.ico` file, RayPack is able to read all icons that are bundled within the source. The **icon preview interface** allows to navigate through all icons which were read. The arrow buttons below the icon preview allow to walk forward and backward through the icons. The index value of the currently displayed icon is shown in order to enable users to separate icons that are visually similar, but are not the same icon object inside the source file.

> 📎 **Note:**
> Please note that the icon index values start at 0 for the first icon found inside a source file, not at 1!

5. If the **All icons** radio button is **activated** and there is more than one icon given within the source file, RayPack imports them all with one step into a single resource.
If the **Only the following icon** radio button is **activated**, RayPack automatically reduces the source file to the icon index that is currently displayed within the icon preview interface.

6. Use the "**Update this resource when the current project is saved / rebuilt**..." checkbox to determine if RayPack should refresh the resource content from the physical file origin each time a project is saved or a target package format (e. g. MSI) is compiled from it. This option is only available when **All icons** radio button is checked.
Activate the checkbox to assign the automatic source update.

7. Click **Next >** to proceed to the **Summary Dialog**.

8. Check the listed information for correctness.
Click **Process** to create the new resource according to those settings.
Use the **< Back** button to return to the previous step and make changes before the resource creation is actually executed.

To Add a New Binary Resource

1. Within the Wizard Selector Dialog: Select **New Binary Resource**.
The Binary Options Dialog is displayed.
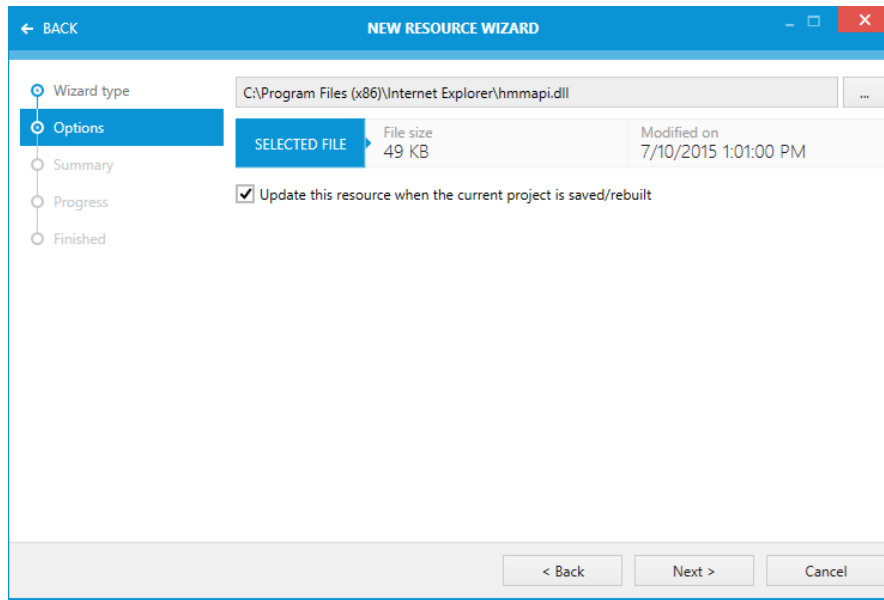
2. **Browse** the local system for the source file of the new resource.
Once the source file is selected, RayPack analyzes its content and automatically determines the **name**, **size**, and **last update time** for the new binary resource.
Whilst the name can be modified manually, size and last update time cannot be edited. Please refer to Edit a resource for further details on how to change resource properties.

The name of a resource has to be a unique, not empty, alphanumeric string with a maximum length of 62 characters. Resource names are internally saved into a column with an Identifier data type, which may contain the ASCII characters A-Z (a-z), digits, underscores (_), or periods (.). However, every identifier must begin with either a letter or an underscore.

3. Use the "**Update this resource when the current project is saved / rebuilt**..." checkbox to determine if RayPack should refresh the resource content from the physical file origin each time a project is saved or a target package format (e. g. MSI) is compiled from it.
Activate the checkbox to assign the automatic source update.

4. Click **Next >** to proceed to the **Summary Dialog**.

5. Check the listed information for correctness.
Click **Process** to create the new resource according to those settings.
Use the **< Back** button to return to the previous step and make changes before the resource creation is actually executed.

As soon as the resource creation is finished, the wizard closes automatically, and the object list within the Resources view is updated.

Replace a Resource

Replacing a resource means to change the source file used as packaging project resource.

Users can trigger the replacement procedure either

- by **right-clicking** a resource object within the list of project resource objects and selecting **Replace** from the **context menu**,
  or
- by selecting the object within the list of project resource objects and using the **...** button within the **details pane** on the right hand side of the Resources view.

The actual replacement procedure depends on the resource type:

Replacing an Icon Resource File

1. Once the replacement is initiated, RayPack displays the **Import / Replace Icons Dialog** for the selection of the file that should be imported into the resource.

2. Users should **BROWSE** for the desired icon source file from the local system.

> **Note:**
> Icon resources can be imported from `.ico`, `.exe` or `.dll` files. All other file types are handled as standard binary resources and can therefore not be imported into an icon resource. If such a change is required, users have to remove the original icon resource and create a new binary resource instead.

3. Once the source file is selected, RayPack analyzes its content and automatically determines the **name**, **size**, **last update time**, and **visual preview** for the new icon resource.

   These properties will be replaced along with the source file. They cannot be changed manually inside the wizard. Please refer to Edit a resource for further details on how to change resource properties.

4. If the selected source is not an `.ico` file, RayPack is able to read all icons that are bundled within the source. The **icon preview interface** allows navigation through all icons which were read. The arrow buttons below the icon preview allow the user to move forward and backward through the icons. The index value of the currently displayed icon is shown in order to enable users to separate icons which are visually similar, but are not the same icon object inside the source file.

> **Note:**
> Please note that the icon index values start at 0 for the first icon found inside a source file, not at 1!

5. If the **import the current icon only** checkbox is **not activated** and there is more than one icon given within the source file, RayPack imports them all with one step.
   If the **import the current icon only** checkbox is **activated**, RayPack automatically reduces the source file to the icon index that is currently displayed within the icon preview interface.

   The import of the current icon only setting used during the initial icon resource creation is not saved permanently. Therefore, the decision has to be made anew for each replacement execution.

6. Click **OK** to import the new source file or **Cancel** to close the dialog without changing the original icon resource object.

7. RayPack automatically updates the **file name**, **size,** and **last update time** property of the resource object to the according values from the new source file.
The name of the resource remains unchanged.

Replacing a Binary Resource File

1. Once the replacement is initiated, RayPack displays a **System Browser** for the selection of the file that should be imported into the resource.

2. Users should pick the desired file from the local system and click **Open** to start the import process.

3. RayPack automatically updates the **file name**, **size**, and **last update time** property of the resource object to the according values from the new source file.
The name of the resource remains unchanged.

Remove a Resource

Removing a resource means to delete the logical resource link between the packaging project and the physical source file. The physical source file remains on the local system and is not affected by the resource removal.

1. Users can trigger the remove procedure by **right-clicking** a resource object within the list of project resource objects and selecting **Remove** from the **context menu**.

   A confirmation dialog is displayed, allowing the user to make sure the resource removal was triggered intentionally and not accidentally.

2. Users should click **REMOVE** to delete the resource or **DO NOT REMOVE** to close the confirmation dialog and return to the Resources view without any changes.

> **Note:**
> As soon as the resource object is deleted from the Resources view, its resembling data row marked as deleted within the affected Installer database table (which may be `Icon`, `Binary`, or a custom binary source table).
> Once the packaging project is saved or the history of changes is cleared, the resource object is no longer visible at all.
> Please refer to the Tables section to get more details about the object manipulation history and markup management.

Editing Resources

Editing a resource entails changing the properties of the resource object stored within the packaging project directly by using the RayPack user interface.

To edit a resource, users should click on the resource object within the list of existing ones within the Resources view. Once selected, the resource details are available for manipulation within the details pane on the right hand side.

Once the properties of a resource are loaded into the details pane of the Resources view, the following manipulation options are available:

- Change the **name** of the resource.

  The name of a resource has to be a unique, not empty, alphanumeric string with a maximum length of 62 characters.
  Resource names are internally saved into a column with an Identifier data type which may contain the ASCII characters A-Z (a-z), digits, underscores (_), or periods (.). However, every identifier must begin with either a letter or an underscore.

- Change the **settings for auto-update** on build/save.

  Activate the checkbox "Update this resource when the current project is saved / rebuilt" to automatically refresh the resource from the physical file path whenever the packaging project is saved or a target package (e. g. MSI) is built from it. If the linked source file has been changed in the meanwhile, this setting makes sure that the latest file version is applied.

  If the checkbox is deactivated, changes made to the physical source file linked to the resource object will not influence your packaging project.

- **Showing source files in Windows Explorer**.

  Use the **Show in Explorer...** button from the details pane to open a file explorer window at the location given within the `<file name>` column. Use any of your standard methods to open the file in the application of your choice.

  Additionally, for icon resources it is possible to see a preview of the actual icon object(s) within the resource item. If there is more than one icon object stored within the icon resource, arrow

buttons make it possible to navigate through them.

> **Note:**
> The properties of a resource object that are available for manipulation, and the actions that can be executed on it, depend on the resource type (binary or icon) and on the origin of the resource (native file or binary package content).
>
> Resources that were already compiled into an MSI or the internal standard resources provided with each new RPP project by default (e. g., the RayPack default resources starting with "RPUI_"), cannot be opened for edition in an external application.
>
> Nonetheless, those integrated resources can be exported into physical files or replaced with physical files. Once replaced, the resource is available for refreshing and editing in external applications again. These advanced operations are available directly from within the Tables view.

File name, size, and last update time values of resources cannot be manipulated manually, but are automatically determined by RayPack.

Merge Modules

RayPack enables packagers to extend their packaging projects with common source file bundles, so called Merge Modules. The Merge Modules view within the Visual Designer mode of the PackDesigner allows the addition of predefined modules that were shipped with the RayPack product resources and also to add custom MSM files from external sources (e. g. local hard drives or network shares).

Merge modules that have already been added to a packaging project are listed in the Merge Module view. Each merge module is represented by a tile, containing the name of the module, its version and publisher. The icons at the bottom of the MSM tile provide direct access to the merge module specific items within the Installer database, as well as, the Merge Module Viewer, which provides details about the files and registry contents stored within the MSM container.

The following procedures are available from the Merge modules view of the Visual Designer mode:

Add Content from a Merge Module to a Packaging Project

Use the button **Add content from a Merge Module** to initiate the wizard for adding merge modules.
See *Add a merge module* for details.

Open the Table Entry for an Already Added Merge Module

Once a merge module has been added, RayPack shows a tile for that module within the Merge module view of the Visual Designer mode. Clicking on the arrow icon at the lower left corner of the tile opens the Advanced mode TABLES view with the related reference row loaded for display.

The exact jump target is determined by the merge status of the module; for those added with immediate merge, the target table is ModuleSignature. Tiles for modules that are assigned to be merged at build time, point to `RPModuleSignature`, which is a RayPack specific extension to the standard Installer database schema.

Open the Merge Module Viewer for an Already Added Merge Module

Once a merge module has been added, RayPack shows a tile for that module within the Merge module view of the Visual Designer mode. Clicking on the details icon at the lower left corner of the tile opens the Merge Module Viewer with details regarding files and registry contents stored with the MSM.

Refer to Knowledge Base Contents Regarding Merge Modules

RayPack users benefit from the pool of additional background information on hot packaging topics, gathered by the community of professionals.
Please make sure to have your user name and password for the Raynet support panel on hand, since these credentials are also required to gain access to the Knowledge base. If there is no login data on hand for the Raynet support panel, please contact our support team: support@raynet.de.

Add a Merge Module

Adding a merge module to a packaging project is initiated from the Merge Module view within the Visual Designer mode of PackDesigner.

To launch the wizard for MSM management, use the **Add content from Merge Module** button on the upper left corner of the Merge Modules view.

Please process the following wizard steps in turn to extend your project with merge module content.

Step 1: Merge Modules

Merge modules can be selected for import from the set of modules shipped along with the RayPack installation resources or from custom external resources.

To **add a module from the list of standard modules**, activate the **checkbox** in the outer left column of the standard merge module list.

The list is quite substantial and is therefore equipped with sorting by column (simply click on any column header to sort ascending or descending by that criterion), and filtering by keyword functionality (use Control + F to toggle the filter visibility). Use these features to quickly navigate to your required merge module.

Use the **Refresh** button above the list of pre-defined modules to update its contents from the internal Merge Modules library.

> **Warning:**
> Once a merge module has been added to a project, it cannot be removed by using this wizard. Therefore, when the wizard is run several times for the same project, the already imported merge modules from the pre-defined set list are marked, but that mark cannot be removed. To remove a merge module from a project, remove the Installer database content that represents it within the TABLES view.

To **add a module from an external resource**, use the **From disk...** button above the list of pre-defined modules.

A system explorer dialog is opened, allowing the user to search for an MSM file on local and network volumes.

Once the right MSM file is selected, the **Open** button is used to add the MSM file reference to the set of available modules. It is already marked for import, meaning that the checkbox in the outer left column is activated.

As soon as the collection of one or more merge modules that are about to be used for the packaging project are marked, the **Next** button has to be used to proceed with the next wizard step.

Confirm Inclusion of Depending Modules

The transition to the next step contains the analysis of the merge module collection, which is assigned for integration into the packaging project. If any of these modules has dependencies to other modules, RayPack displays a confirm dialog, requesting a user decision regarding the handling.

The confirmation dialog shows a list of depending modules, which can be toggled to display by using the **MORE...** button. The decision to include the required modules is either for all or for none, it is not possible to select only a subset of dependencies for inclusion.

Users have three options to choose from:

- Click **YES** to include the depending modules
- Click **NO** to proceed without the inclusion of the depending modules
- Click **CANCEL** to go back to the list of merge modules and change the current selection.

Step 2: Target Feature

Each merge module has to be wrapped into a package feature. Therefore, users have to pick a feature from the displayed tree view structure of already existing features or add a new feature.

The arrows left of a feature name indicate that it has nested child features. With a click on the arrow the next level of children is displayed (or toggled to be hidden again).

To add a root feature, the **Create** button is used and the **New root feature** option selected from the option menu.
If it is required to add child features, use the more advanced feature management interface within the Features view to accomplish that.

As soon as one of the features is selected, the **Next** button becomes available. Use it to proceed to the next wizard step.

Step 3: Options

When an MSM is added to an RPP project, RayPack allows two different methods for MSM import. Modules can either be imported and merged in one single step or imported and assigned for later merge at build time.
When an MSM is added to an MSI, the merge process is always executed immediately. Merge at build is not available in these cases, as well as, the whole wizard step Options.

Merge Now

This option immediately merges the content objects of the MSM file with those already existing within the packaging project. This means, for example, any registry key, file, folder, etc. that is contained within the merge module is taken over into the packaging project.

Adding an MSM for immediate merge affects the set of available Installer database tables, as well as, the content of existing ones. Effects are likely to appear within the tables `Directory`, `File`, `Registry`, `Feature`, `Component`, `FeatureComponent`, `ModuleSignature`, `ModuleDependency`, `ModuleComponents`, `_Validation`. The exact changes depend on the imported MSM file.

Use this option to see the impact of the MSM inclusion immediately, and deal with potential resource conflicts, for example, per single MSM.

Merge at Build

This option adds the Merge Module resources to the packaging project, but does not merge the actual contents (e. g. files, folders, registry keys, etc.) with the current packaging project contents.

When the package project is saved, the merge module sources are copied into the `<project name>.rpp.Sources` directory. The exact path reference is saved within the Installer database table `RPModuleSignature`, column ModulePath. This path is resolved at build to execute the actual merge process at that time.

> **Warning:**
> When a packaging project is transferred from one storage location to another, the `<project name>.rpp.Sources` directory must be transferred along with it, since path values, e. g. the path to merge module source files, are relative ones. Moving the `.rpp` file alone will lead to errors when that copy is opened for edition in RayPack from that new location.

Adding an MSM for merge at build affects the following Installer database tables:

- `Feature`
  Only if a new feature / component was added during the wizard execution

- `RPModuleSignature`
  Has been added with new entries, one for every imported merge module. The column ModulePath points to the MSM source file.

- `_Validation`
  If the tables named above were not available before, RayPack will automatically extend this table with addional internal validation rules.

Use this option to prepare all required resources for the desired target package first and handle possible conflicts later in one (potentially complex) build validation procedure.
Since MSM files are updated with new versions from time to time, Using the Merge at Build option may be a useful selection for long-lasting packaging projects.

Click on one of the two method tiles to apply it for this MSM import process execution.
As soon as one of the methods is selected, the **NEXT** button becomes available. Use it to proceed to the next wizard step.

Step 4: Summary

The Summary page lists all settings defined in the steps before for a final checkup prior to the module import being actually executed.
Use the **Next** button to start the import.
Use the **Back** button to make changes to the current settings.

Step 5: Progress

The Progress page is visible as long as RayPack actively imports the merge modules.

Naturally, merging modules immediately takes longer than importing for a later merge. The duration of the import process depends on the complexity and size of the imported MSMs, as

well as, on the performance of the underlying packaging machine.

Step 6: Finished

Once the import has been successfully accomplished, the Finished page is displayed. Use the **Finish** button to close the wizard and observe the updated list of merge modules used within the current project.

View Merge Module Details

Files, folders, and registry information of Merge Modules from the library (or the stock of MSM files present within a packaging project), may be reviewed within the Merge Module Content Viewer. To open the viewer, users have to left-click on the details icon, displayed at the bottom of any merge module tile from within the Merge Modules view.

As soon as the Merge Module Viewer is displayed, it provides a tab for Files and Registry items that are bundled within the merge module. Click on the tab title to display the content type item listing. The tab titles contain a number in brackets, revealing the type specific number of items present within the current MSM file.

Tab: Files

The flat list of files contains columns for the File name, the parent Folder, Version and Language. The table may be sorted by any of the given columns by simply clicking on the header label. Clicking on the same header again switches from ascending to descending order and vice versa.

Tab: Registry

The flat list of registry content provided within the MSM file displays the full path of the Key, the Name of the registry item, and the Value. If the Name column for an item is empty, the Value contains a default registry item. Just like within the Files tab, the table may be sorted according to the column values by clicking on the column headers.

Scanning for Merge Modules

Existing projects (RPP, MSI, and MST formats) can be scanned for the presence of any resources which may be replaced by one or more Merge Modules, present in PackPoint repository.

In order to perform scanning, press **Scan for replaceable** button inside the **Merge Modules** screen. The project will be immediately scanned, and the results will be presented using a tree, displaying possible matches and their matching content.

In order to assign one or more Merge Modules, select them using checkboxes and press **Use selected Merge Modules**.This on the other hand will trigger the **Merge Modules** wizard.
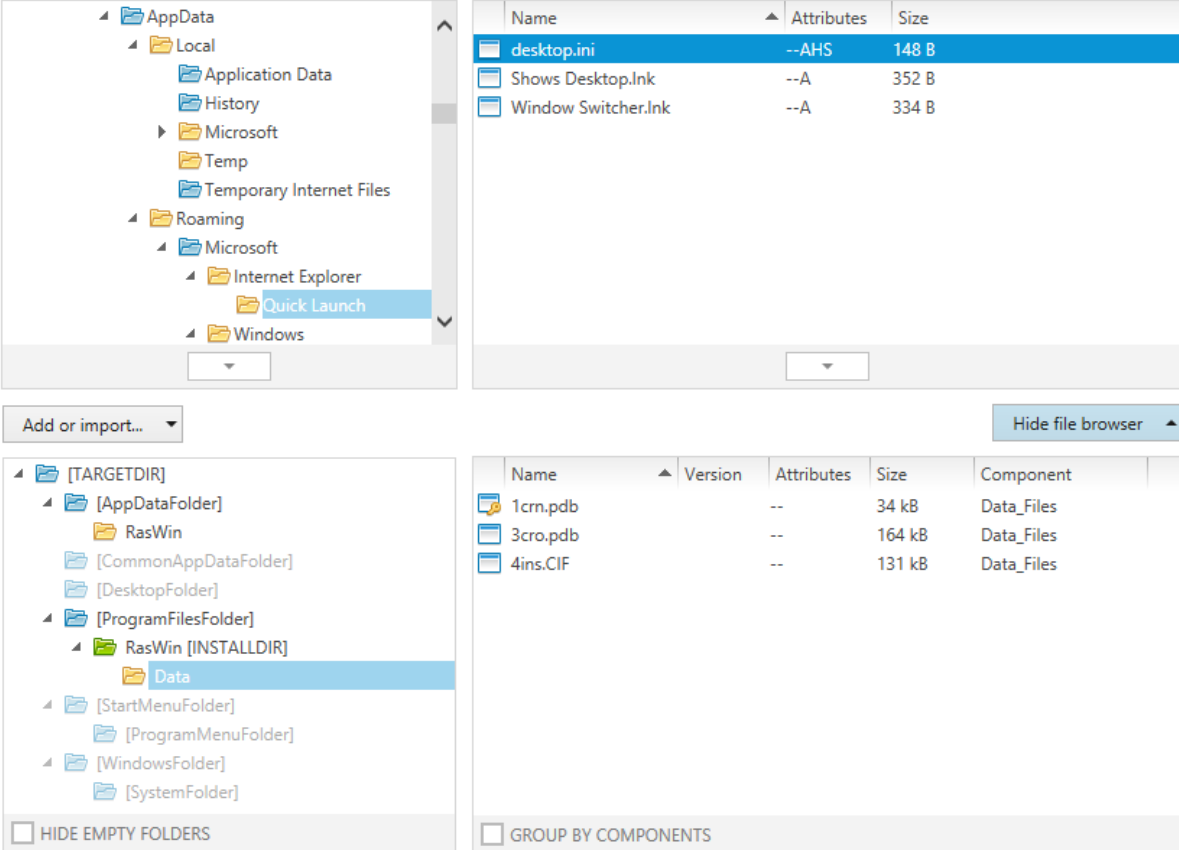
System Configuration

The System configuration view allows you to make changes to the Files & Folders, Registry, Shortcuts, Environment variables, and Properties settings.

Files and Folders

The Files and Folders view is designed to manage the transfer of directory structures and files from the package resources to the target machine during setup execution. In order to provide a user interface that does both, i.e. make it easy to put files and folders into the package and at the same time offer an elaborate set of manipulation options; make sure that those resources arrive and behave exactly as expected on the target machine.

The Package File Browser

Within the Visual Designer view Files and Folders, the main interface component is the **package file browser**, which consists of two windows. On the left it contains a tree structure of directories which are already available within the packaging project. Once one of those directories is selected, the details window on the right hand side presents the files which are stored within the directory.

The following sections provide an overview of the most important interface features designed to improve the user experience and efficiency of the package file browser:

The Directory Tree Color Scheme

Folders within the left window of the package file browser are displayed by appliance of a dedicated color scheme. Once a packager gets familiar with the scheme, it noticeably saves time spent on orientation within projects directory structures.

- **Blue folder icon**
  Blue folders represent Windows Installer predefined system folders. Their actual path on the target machine is not given as a static value, but is determined at Installer run time.

- **Green folder icon**
  Every RayPack packaging project has a maximum of one INSTALLDIR, which is the default product destination folder. The INSTALLDIR is the only folder that is displayed with a green icon.

- **Yellow folder icon**
  Folders created manually are displayed with a yellow icon. Within their parent directory, they are considered static.

- **Pale icon and folder name**
  Folders that have been created, but not filled with file content yet are displayed in a pale mode. As soon as content is put into them, the full color saturation is applied.
  Folders are considered empty, even if they have sub-folders, as long as there is no file placed in the folder itself or one of its sub-folders.

Hide Empty Folders

The checkbox to control the display of empty folders is placed at the bottom of the package file browsers left window. Activate the **HIDE EMPTY FOLDERS** checkbox to hide empty folders from the current view and work explicitly on folders that contain files or linked content. Deactivate the checkbox to return to the full view mode again.

Group by Components

The files within one folder of a projects directory structure may be wrapped in one common component, but usually they are not. In order to quickly determine which file belongs to which component, the **GROUP BY COMPONENTS** checkbox can be used. If it is activated, the file list is grouped by components. The arrow left of the component group header expands and collapses the list of files for that specific component to further improve the clarity of the grouped list.

Sort by Property

The list of files stored within a specific folder is displayed in a table with the columns Name, Attributes, Size and Component. Use the column headers to sort the list by one of these criteria (ascending or descending).

Go to Row

Each object that is displayed within the left directory or right file window of the package file browser is a representation of one specific row of the Directory or File table of the Installer database. With a right-click on the object and the selection of the Go to Row option from the context menu, RayPack switches to the TABLES view of the Advanced mode and focuses the related table row for the file or folder object.

The System File Browser

The first phase of working on files and folders is to import external resources into the packaging project. During this initial phase, users will often work with the **system file browser**. This is another set of two adjoining windows for a directory tree on the left and a file listing on the right. Whilst the package file browser contains files and folders present within the packaging project, the system file browser is an exact representation of the directories and files on the packaging machine that runs RayPack.
See *Add files and folders* for details.

File and Folder Object Manipulation

Once files and folders are available within the project, they need to be enriched with specific settings in order to control their behavior at Installer run time and during their presence on the target device. This includes settings regarding permissions, feature and component relations, attributes, and the like.
See *Edit file properties* and *Edit folder properties* for details.

During the import or creation of files or folders, RayPack applies standard settings for each new object within the project contents. Some of these defaults can be customized by modifications within the settings profiles. For example, whenever new files are added to a project via the Visual Designer view Files and Folders, the naming convention from the profile settings for pack designer is applied. This convention determines the name of the component that is automatically created by RayPack to wrap files that do not belong to another specific component.
See the help contents about *editing settings profiles* for details.

Files and folders are represented by items within the Installer database tables File and Directory. Actions regarding the file and folder objects are stored aside from that in additional tables, e. g., `CreateFolder`, or `RemoveFile`. Switch to the **TABLES** view of the Advanced mode to access these tables directly.

Add Files and Folders

Within the Visual Designer mode, adding folders to a packaging project may be done by manually by creating new objects inside the existing structure or by *importing existing objects from external resources*, such as local volumes, network shares, or flash drives. However, for files it is only possible to import them via the **Files and Folders** view, they cannot be created from scratch here.

The default template for new projects creates a basic structure of directories to start from. The root of this base is `[TARGETDIR]`, which is a so called predefined folder. The square brackets around the folder name indicate that the actual path and name of the folder on the target machine will be defined on run time. The `TARGETDIR` property specifies the root destination directory for the installation. Since there is a whole set of Windows Installer predefined system folders, RayPack users may choose to *use one of those predefined folders* or *create custom folders* at a specific location.

To Add a New Predefined Folder

As outlined before, the **Files and Folders** view contains a *package file browser*, which is always visible and an optionally displayable *system file browser*.

1. Right above the **package file browser**, an **Add or import...** button is available, revealing a set of folder creation options on click.

2. From the list of options, select the **Predefined folder...** option to display the dialog with a listing of all predefined folders within the **Special Installer Folders** dialog.

   Folders which are printed in a grey font style are already present within the directory structure of the current packaging project. They are presented in a view-only mode, since they cannot be added again or removed from the project by using the interface controls of this dialog.

3. The Special Installer Folders dialog contains the list of predefined folders in three columns:

   - **Use**
     This column indicates if a predefined folder has been marked for usage within the packaging project. Therefore, already existing folders show activated checkboxes in this column. Users have to activate the checkbox in this column to mark folders they want to add to the projects directory structure.

   - **Id**
     The predefined name of the folder.

- **Resolved path**
  This is the actual path to the predefined folder target as it would be resolved for the packaging machine on Installer run time. Since the packaging machine may vary from the packages later target machine in operating system and architecture, the resolved path is just intended to give an example of the actual path. At run time, the actual path is resolved to match the target machines environment.

4. Once the desired set of predefined folders is marked with **active checkboxes in the use column**, they can be added to the project by either using the **OK** button, which immediately creates the folder(s) and closes the Special Installer Folders dialog or by clicking the **Apply** button, which also adds the folder(s), but keeps the Special Installer Folders dialog open for further activity.

   When the **Apply** button is used, the newly added folders are automatically transferred into a view-only mode within the current dialog, since they cannot be removed from the packaging project via this dialog.

   The **Close** button closes the **Special Installer Folders** dialog without applying any new changes to the folder structure of the packaging project.

5. As soon as the **Special Installer Folders** dialog is closed, the directory structure within the package file browser is updated with the newly added folders.

In order to customize the default choice of predefined folders, refer to the Customizing predefined folders section.

> **Note:**
> 64-bit folders (for example `ProgramFiles64Folder`) are not shown if the target architecture of the current package is 32-bit (`Intel`). This setting can be changed in the Summary information view.

To Add a New Custom Folder

Creating a new folder always starts by selecting the parent directory of the new folder within the left window of the package file browser. Once the parent is chosen, there are several methods to trigger folder creation:

- **Right-click** the parent directory and select **New Folder**
- Click the **Add or import...** button above the package file browser and select **Folder** from the options menu
- Hit **insert** on the **keyboard**

Each of the named methods creates a new, empty folder with the default name `New folder`. If a folder with that name is already present within the same parent directory, an automatically incremented index value is added (e. g. `New folder (2)`)

The new folder is already focused for edition therefore immediately typing the actually desired

name of the new folder and hitting **Enter** on the keyboard, the default name will be replaced. Please make sure to use a valid folder name by avoiding special characters or duplicate folder names within the same parent directory. Of course, the folder name cannot be empty.

However, since the new folder itself is empty after creation, it is displayed in a pale font and icon style. (see *The directory tree color scheme*)

To Import Files or Folders

The **Files and Folders** view generally offers three methods for importing files or folders:

- *Drag and drop from a Windows explorer instance*
- *Transfer from system file browser interface*
- *Select from Windows system browser instance*

Import by Drag and Drop

When instances of RayPack and Windows explorer are open in parallel windows, it is possible to drag and drop files from the Windows explorer into any directory within the package file browser of the **Files and Folders** view.

To do so:

1. Select the target folder within the RayPack package file browser window (lower left window within Files & Folders)
2. Switch to the Windows explorer window
3. Mark one or more files
4. Drag the selection from the Windows explorer to RayPack
5. Drop the selection into the right window of the package file browser.
6. A copy of the selected files is created within the file stock of the packaging project.

> **Note:**
> Due to consistency reasons, it is not possible to drag & drop folder objects to a packaging project. The dropped selection must be a collection of single files, otherwise RayPack ignores the drop attempt. The import attempt is also rejected when the selection is dropped inside the left window of the package file browser.
> Please make sure to keep drag source and drop target application window at UAC level (if UAC is active).

Import by System File Browser

To import files or folders from the system file browser, users have to display the system file browser first. To do so, click on the **Show file browser** button on the upper right corner of the **Files and Folders** view.

Within the left window of the system file browser, navigate to the folder that has to be imported into the packaging project. Select it with a mouse click.
RayPack loads the list of files which are stored within the selected folder into the adjacent right window.

Within the left window of the package file browser, navigate to the target folder for the import.

Select it with a mouse click.

To import the **whole folder and all its contents** (sub-folders and files), use the **arrow** button below the **left window** of the system file browser. A copy of the selected folder is imported into the project and displayed within the left window of the package file browser.

If a **selection of files** has to be imported, use the right window of the system file browser. Select one or more files (if required, use the control key to multi-select files) and use the **arrow** button below the **right window** of the system file browser. A copy of the selected files is imported into the project and displayed within the right window of the package file browser.

	Note:
	Due to consistency reasons, it is not possible to import several folders or a sub-selection of files from different folders in one single step. To do so, apply the procedure given above for all required folders in turn.

The local browser provides full support of drag and drop operations. Files and folders can be imported by simply dragging a node from the local browser pane to the files view (to import files to currently selected folder) or to the folders view (to import files to a specific node).

	Note:
	Due to consistency reasons it is not possible to import a folder into the files view of the currently selected folder.

Import by Windows System Browser

1. Within the left window of the package file browser, **navigate to the target folder** for the import. **Select** it with a mouse click.

2. Reveal the set of object creation options for package folders with a click on the **Add or import...** button above the package file browser.

3. From the displayed menu, select **Import file(s)...** to open a Windows system browser dialog.

4. Browse to the required folder within the **Select files to be imported** dialog.

5. **Select one or more files** from that folder and use the **Open** button to import them into the packages file stock.

   A copy of the selected files is imported into the project and displayed within the right window of the package file browser. The windows system browser is closed automatically, and the Files and Folders view is updated to show the newly imported files within the right window of the package file browser.

	Note:
	Due to consistency reasons, it is not possible to import several folders or a sub-selection of files from different folders in one single step. To do so, apply the procedure

Using Variables to Reference Source Paths

For advanced scenarios, variable paths can be used to determine the source of physical files. Refer to the following sections for more information:

- Path variables
- RPVariable

With RayPack, files and folders can be imported to an MSI / MST / RPP based projects on build time. The scenario is particularly useful if one of the following applies:

- The packaging project is created for self-made application, with a lot of resources that change constantly (for example in terms of folder structures, names of files etc.)
- A folder containing a lot of files (40 000+) has to be imported

Linking a physical folder on a hard drive to a folder present in RayPack project can be configured from the folder properties dialog.

	Be aware:
	RayPack creates components and automatically assigns new random GUIDs to them. This means that projects using the "linked folders" functionality may be less suited for upgrade scenarios, because two packages created one by one will have different component identifiers..

The linked folders are resolved on the build-time. It means that if a linked folder is configured for an MSI / MST project, no files will be added to the MSI / MST unless a full rebuild of the package using the FILE > REBUILD menu is done. RayPack shows a warning inside the Folder Properties Dialog if the current project is not RPP.

To Define a Linked Folder

1. Select or create a folder to which files and folders are to be imported on build-time in the **Files and Folders** view
2. **Right click** the file and choose **Properties…** menu item from the **context menu**
3. Tick the checkbox **Generate the content of this folder during building**
4. The linking options will be expanded:

Source
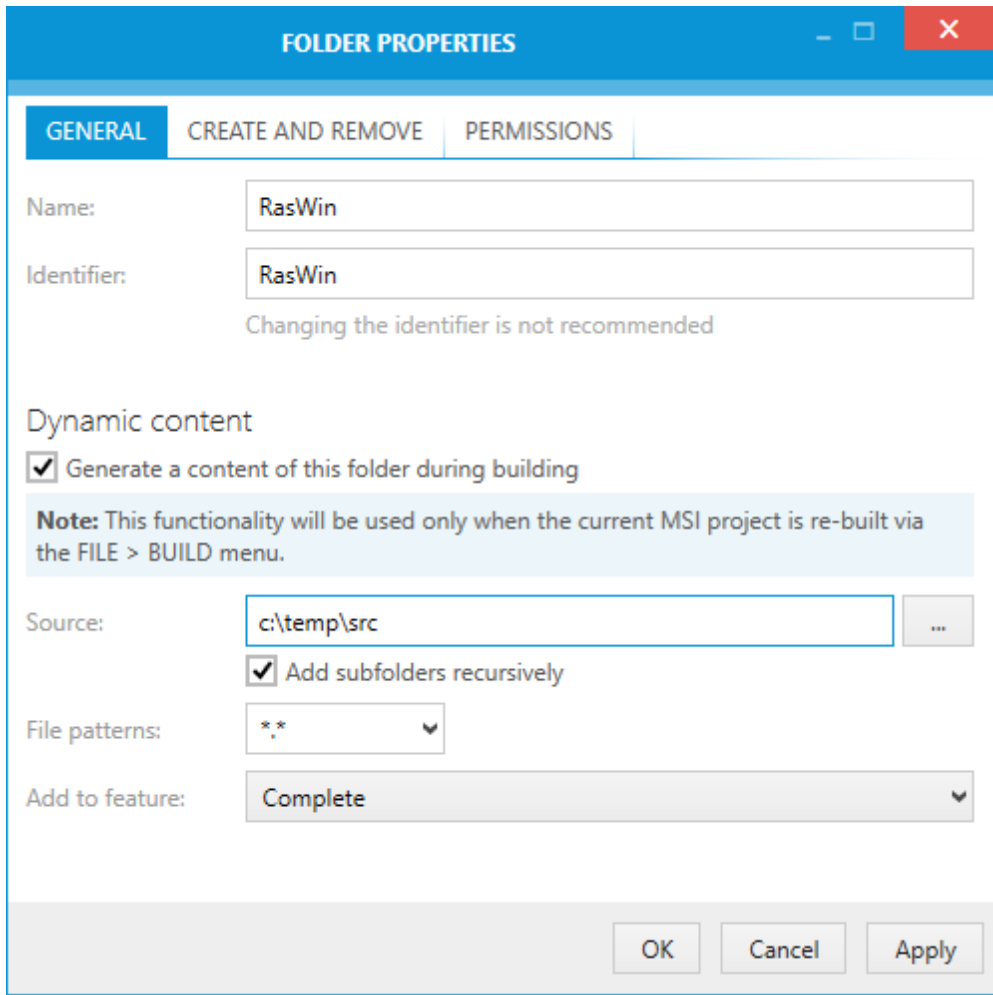
The source folder from which the content will be imported on build-time. In order to include subfolders recursively, tick the checkbox **Add subfolders recurisvely.**

File Patterns

The wildcard string to import only specific types of files. For example, typing `*.exe` will import only executable files. To import all files, leave this field empty or type `*.*`

Add to Feature

Select the feature to which imported components will be assigned. This field is required, so make sure there is at least one feature in your project before setting up the linked folder.

5. To build the project, press **FILE** > **BUILD** and select the target format.

> **Be aware:**
> The source folder selected above will not be imported itself. It serves as a root of content to be imported, meaning only its children (files and folders) will appear in the output file in the specified target folder.

Using Variables to Reference Source Paths

For advanced scenarios, variable paths can be used to determine the source of physical files. Refer to the following sections for more information:

- Path variables
- RPVariable

When RayPack detects that the file being imported is a font file, it automatically sets up the following extra settings:

- A new component that uses reference counting is created and used as a component of the imported file
  o If present, an existing component located in the same folder and having proper attributes may be used instead.
- A registration entry is added to the `Font` table. Font registration settings can be edited in the *File Properties dialog*.
- If not yet present, necessary actions are added to sequence tables to ensure the font is correctly installed

This process is fully automated and requires no action from the user.

To configure which files are considered as fonts, change appropriate settings in *Best practises section* in the Settings screen. By default, True Type fonts (`.ttf` and `.otf` files) are imported as fonts.

Renaming a File or Folder

The package file browser of the Visual Desginer view Files and Folders offers several methods to trigger the renaming of a file or folder:

Select the object that has to be renamed and
- **Left-click** it and hit **F2** on the keyboard.
- **Right-click** it, and select **Rename** from the **context menu**.

  Both methods trigger that the name of the object within the current view is set into the direct inline edit mode. Immediately typing the new object name and hitting the enter key saves the new name.

- **Right-click** it and select **Properties** from the **context menu**.
  Either the FILE PROPERTIES or FOLDER PROPERTIES dialog is displayed, allowing to edit the object name.
  See *Edit file properties* or *Edit folder properties* for details.

Renaming an object has to follow the same rules that are applied on object creation:
- File and folder names must be unique within the their direct parent directory
- Special characters or empty names are not allowed

Whenever a user tries to rename an object within the package file browser, RayPack checks the validity of the new value and shows an error icon if there are issues that prevent the new name from acceptance. Please hover over the icon to reveal the error message and adjust the new value according to that description.

Removing a File or Folder

The package file browser of the Visual Designer view Files and Folders offers two methods to trigger the removal of a file or folder:

Select the object that has to be deleted and
- **Left-click** it, and hit **Delete** on the **keyboard**.
- **Right-click** it, and select **Delete** from the **context menu**.

Once an object is removed, it is lost from the packaging project. If a folder is removed, all its contents are also removed.

Since an accidental object removing could lead to substantial issues, RayPack displays a **Confirm** dialog before the object is removed. Use the **REMOVE** button within that dialog to confirm the activity. If **DO NOT REMOVE** or **CANCEL** are clicked, the object is not deleted and the user returns to the Files and Folders view without any changes.

The display of the Confirm dialog may manually be abandoned for future removal procedures by activating the "In future do not show this confirmation for delete operations" checkbox and using the REMOVE or DO NOT REMOVE button next (using the CANCEL button does not save any changes made to the status of that checkbox). However, it is not recommended to skip the confirmation step, since it may prevent users from severe data loss.

Be aware:
RayPack does not protect predefined or specialized folders from deletion. Therefore, users have to make sure that potential side-effects of deleting directories, such as TARGETDIR or INSTALLDIR, are decently compensated. Unexpected side-effects may also be caused by deleting special files, such as files which are marked as keypath of a component. Double-check cross-reference consistency before such vital files or folders are deleted.

Note:
As soon as the file or folder object is deleted from the Files and Folders view, its resembling data row is marked as deleted within the affected Installer database table File.
Once the packaging project is saved or the history of changes is cleared, the file or folder object is no longer visible at all.
Please refer to the *Tables* section to get more details about the object manipulation history and markup management.

Editing File Properties

Once a file has been imported into one of the folders of a packaging project, its properties can be adjusted.

To do so, users
- **right-click** the file within the right window of the package file browser.
  - To edit the file properties within the Visual Designer interface, users select the **Properties** option from the **context menu**.
  - To edit the file properties directly within the resembling row of the File table of the **TABLES** view of the Advanced mode, users select **Go to row** from the **context menu**.
- **left-click** the file within the right window of the package file browser and use the key combination **Alt + Enter**

The **FILE PROPERTIES** dialog is displayed and reveals five tabs of property groups for manipulation:
- **GENERAL**
- **ATTRIBUTES**
- **SOURCE**
- **SELF-REG**
- **PERMISSIONS**

As soon as all required changes are executed, users can either use the **Apply** button to save the changes to the file object and keep the **FILE PROPERTIES** dialog open or use the **OK** button to save and close the dialog.
With a click on the **Cancel** button, the dialog is closed without applying unsaved changes.

The GENERAL Tab

Name

This is the name of the file as it appears on the target machine. It has to be unique among the files within one directory. A file name may not contain special characters or be empty. See MSDN for details regarding limitations for values of `FileName` columns.

Identifier

The identifier is the unique internal pointer to the file. The Identifier data type is a text string. Identifiers may contain the ASCII characters A-Z (a-z), digits, underscores (_), or periods (.). However, every identifier must begin with either a letter or an underscore.

Component

Select the parent component of the file from the list of available components of the current packaging project. A file always has to be related to a specific component. Therefore, when a file is added to a packaging project, RayPack automatically creates a wrapping component if required. The naming convention for these automatically generated components can be changed within the *pack designer settings* section of the settings profile editor.

If the list of existing components does not contain the required component yet, use the browse button [**...**] right next to the select control to open the **SELECT COMPONENT** dialog. See the section about common dialogs to get information about the handling of this dialog.

Key Path

If a file is marked to be the key path of a component, the existence of the key path determines the health state of that component. If the file does not exist on the target machine, the Installer's Repair functionality assumes that the component needs to be repaired.

File Size

The size of the file in bytes has to be a non-negative integer value. Even though the file size value may be edited manually, it is recommended to keep the automatically calculated values. Nonetheless, exceptional size value manipulations may be necessary to manage storage space requirements beyond the boundaries of best practice procedures.

Override Default Version

Activate this checkbox to gain access to the version value for the file. The version may be set manually, following the *standard format guidelines for file versions*.

Override Default Language

A list of decimal language IDs separated by commas as demanded by the *standard format guidelines for language values*.
Font files should not be authored with a language ID, as fonts do not have an embedded language ID resource. Thus, this column value should be left null for font files.

The ATTRIBUTES Tab

Read-only

This attribute indicates that a file should not be altered. Upon opening the file, file system API usually will not grant write permission to the requesting application, unless the application explicitly requests it.

Hidden

Tools like Windows Explorer do not show hidden files by default, unless asked to do so.

System

This attribute indicates that the file is a critical system file that is necessary for the computer to operate properly. Tools like Windows Explorer do not show system files by default even when hidden files are shown, unless asked to do so.

Vital File for Installation

Activate this checkbox if the installation process cannot be finished successfully without the correct installation of this file.
If a file is marked to be vital for the accurate operation of the component to which it belongs and its installation fails, the whole package installation stops and is rolled back. If files are not vital, users may ignore failed installations and resume the setup routine.

The SOURCE Tab

The source tab shows the details of the actual source file. Depending on the compression configuration and project type, the source file may be available as:

- External uncompressed file
- File compressed to external cabinet
- File compressed to embedded cabinet

Depending on the source location, the options presented in the source tab may differ.

File Name

This is the name of the source file. The name is unique within the same CAB file or folder. This value is read-only.

Source

This field shows the source location of the file. The badge on the left side shows the source type and can be one of the following values:

- **FILE** – the file is stored as external uncompressed resource. Its full path is presented on the right side.
- **CAB** – the file is stored in the external cabinet file. The full path to the cabinet file is presented on the right side.
- **EMBEDDED CAB** – the file is stored in the embedded cabinet file. The name of the CAB file is presented on the right side.

File Size

This shows the size of the source file (in bytes). This value is read-only.

Version

This shows the version of the file. If the file is non PE-executable, the `n/a` string will be shown instead. This value is read-only.

Language

This shows the numeric language identifier of the file. If the file is non PE-executable, the `n/a` string will be shown instead. This value is read-only.

In case any of the three values (size, version and language) are different than the values stored in the MSI tables, the data can be synchronized again by clicking on the **Update the MSI information to match these data** button.

Modified on

This shows the modification date of the file. This value is read-only.

Created on

This shows the creation date of the file. If the file is stored in cabinet file, the `n/a` string will be shown instead. This value is read-only.

Saved by

This shows the name of the person who saved the document. If the file is stored in the cabinet file, the `n/a` string will be shown instead. This value is read-only.

The SELF-REG + FONT Tab

The **SELF-REG + FONT** tab shows the details of the self-registration performed by the MSI engine when the package is installed or uninstalled, as well as font registrations. Self-registration uses the Windows Installer table `SelfReg`.

Using Font Table to Register Fonts

Fonts in various formats (for example `.ttf`, `.otf`, `.fon`) can be automatically registered by enabling the checkbox **Use Font table to register this font.** In most of cases, additional settings are not required.

In case the font contains a name that is not embedded in the font file (for example in case of `.fon` files), it is required to enter the name in the **Font** name field. Once the value is entered you must ensure it is identical to the name of the registered font. **True Type** fonts (`.ttf`, `.otf`) already contain an embedded name, and it is recommended to leave this value empty for them.

> **Note:**
> It is recommended that the component containing the font file have the `FontsFolder` specified in the `Directory_` column of the `Component` table.

Using SelfReg Table to Self-register Files

The installer calls the `DllRegisterServer` function during installation of the module; it calls `DllUnregisterServer` during uninstallation of the module.

> **Note:**
> Due to the technical limitations of the Windows Installer engine, only DLL files can be used with this option. RayPack does not prevent the self-reg option for other types of files, but they may be eventually ignored when the package is installed. If the user wants to register a COM file having any other extension (for example EXE, TLB, OCX), then the COM registration has to be extracted and deployed via the `Registry` table.

In order to enable the self-registration by the Windows Installer engine, the checkbox **Use the SelfReg table to self-register this file** has to be enabled. When the self-registration is enabled, an additional text field will be visible, providing a way to specify a size that should be reserved for the registration. The value is expressed in bytes and the default value is 0 (do not reserve any additional cost because of registration).

> **Note:**
> The usage of the `SelfReg` table may lead to a lack of control over several aspects available by the installer engine, including the ability to repair, advertise, rollback and cleanup the registration information. When any of this functionality is required, it is recommended to extract the COM registration from the registered file and put it into one of MSI tables (Registry, Class etc). More details about extraction are available here.

The PERMISSIONS Tab

Permission handling is a common task in RayPack, occurring for files, folders, registry objects, and the like. Therefore, please refer to the general description of the Object Permissions management in the Common Dialogs section.

Extracting COM Information

When adding any DLL library containing COM information, it is usually required to register it using either self-registration or manually creating necessary entries. RayPack provides an easy automated way to extract the necessary data and insert them to the Registry table, so that no runtime registration is required anymore.

> **Note:**
> If you prefer to self-register your files on installation, consider using the self-registration mechanism available in the file properties dialog.

To Extract the COM Registration Data

1. Locate your DLL / OCX file in the file browser view
2. Right click the file name and select the option Extract COM Information from the context menu
3. Depending on the amount of registry information, the process may take up to a few seconds
4. After the extraction is finished, a confirmation message will appear showing which registry entries were imported. All necessary registry information is assigned to the component containing the COM file.
5. The process can be repeated for files that were already extracted – in that case the registration info will be overwritten with the current one.

To Extract COM Data Automatically when Files are Added to the Project

By default, RayPack does not scan the files that are added to the project. In order to enable this functionality, the current profile has to be adjusted.

1. Click **FILE** and select **SETTINGS**. This will open the current profile configuration.
2. Navigate to the **Designing** tab and select **BEST PRACTICES**.
3. Make sure that the checkbox **Automatically create components and folder operations for folders in user profile** is enabled
4. Save the changes if necessary.
5. When adding new files to the project (for example from the local system browser or from system file open dialog), the COM information will be extracted and populated into the Registry table automatically.

File Operations

File operations are special actions that are performed by the installer engine when the package is installed or uninstalled. The following operations are supported by the Windows Installer:

- **Duplicate file** – copies a specified file from the package to a target location
- **Move file** – moves or renames any file (both from and outside of the package) to another location
- **Remove file** – removes file(s) or folder (on install, uninstall or both)
- **CreateFolder** – creates a folder

The following file operations are shown in the files and folders view:

- Remove file
- Duplicate file
- Move file

The following folder operations are shown in the folder properties dialog.

- Create folder (as checkbox)
- Remove folder (as checkbox)

In order to define a new file operation, users have to right click any file and select **New operation** from the context menu. The following operations will be available:

- Duplicate file
- Move file
- Remove file

When the context menu is invoked by right-clicking on any empty area within the file browser, the **Duplicate file** option is not available.

File operations are shown in the same view where all normal files are displayed. The view sorts the items so that operations are always visible on top and they have different icons to indicate the behavior and differentiate from normal files. When the icon is hovered upon, a description containing partially resolved paths will be displayed as a tooltip.

The properties dialog can be used to customize the behavior of any file or folder operation.

Type

Denotes the type of the file/folder operation.

Folder (only Remove and Create folder)

Specifies the target folder that has to be created/removed.

Component

Specifies the component controlling the file/folder operations.

Source (only Move, Remove and Duplicate file)

Specifies the source object that will be duplicated/moved/removed. For Duplicate file operation, this section contains the combobox to select the source file. For other type of operations, users have to select the directory and the file name of requested source file.

Destination (only Move and Duplicate file)

Specifies the destination object after moving/copying the source object. A directory and a file name has to be provided. If the value of the `File` name field is empty, it is assumed that the source name has to be used. The target and source full file paths cannot be the same.

Scheduling (only Remove file and Remove folder)

Specifies when the operation will be executed. The allowed options are:
- Remove on installation
- Remove on uninstallation
- Remove on both installation and uninstallation

The file operation wizard presents different steps, depending on the type of operation selected.

Source Screen (only Move and Remove file operation)

The following properties are required:

- **Directory**
  A source directory containing the moved/renamed/removed file. This has to be a valid entry present in the `Directory` table.

- **File name**
  The name of the file that will be moved/renamed/removed. The file does not have to belong to the package.

> **Note:**
> (Remove file operation only)
> - In order to delete all files in the folder, simply type the wildcard expression (`*.*`). Alternatively, the ALL FILES checkbox can be checked to mark removal of all files.
> - The Source screen contains a component property. Please refer to the section about the Destination screen below to read about the available options for the selection.

Destination Screen (only Move and Duplicate file operation)

The following properties are required:

- **Directory**
  A destination directory to which the selected file will be moved/copied.

- **File name**
  The name of the target file after moving/copying. The full path of the moved/copied file has to be different that the full path of the source file.

- **Component**
  The file operation has to be attached to a component to control and trigger its execution. Please select whether:

  o A new component has to be created to contain the operation.
    To do so, activate the upper option of the Component radio control. RayPack will automatically set a default name for the new component.

  o The operation should be added to an existing component.
    To do so, activate the lower option of the Component radio control. It is possible to either keep the pre-selected component, to select another one from the selector control, or to use the common Select Component dialog to define the parent component object. Within this dialog it is also possible to create a new component manually, which allows to define the name of the component, instead of letting the internal RayPack logic define it automatically as provided by the option described above.

Scheduling Screen (only Remove file operation)

This screen contains a radio button controlling when the removal has to be performed. Permitted options are:

- **On installation** – the file(s) will be removed when the controlling component is installed
- **On uninstallation** – the file(s) will be removed when the controlling component is uninstalled
- **Both on installation and uninstallation** – the combination of the two above options.

> **Note:**
> The new file operations wizard always creates a new component when invoked from the files and folders view. If the wizard is invoked from the components view, it automatically assigns the newly created operation to the source component.

Replacing Files

Any file in an MSI / RPP / MST project can be replaced. The replacement operation updates the sources of the file, but does not affect any connections, including KeyPaths, formatted field references, component assignments, etc. An example scenario involving file replacements without changing the remaining MSI structure are quick patches that upgrade a few files to newer versions.

To Replace a File

1. Select the file to be replaced in the **Files and Folders** view
2. **Right click** the file and choose **Replace…** menu item from the **context menu**
3. Select a new version of the file from the file system
4. The necessary entries, including file size, version, and source path will be updated automatically.

> **WARNING**
> Make sure the file has internally a higher version than the old file, otherwise Windows Installer will not be able to replace the file during the upgrade process. If the new version is lower but the file has to overwrite the old version anyway, it is necessary to override the file version in the file properties dialog.

The path to the updated version of the file will be created in the RPSourcePath table in the MSI/RPP/MST. When the project is rebuild (RPP/MSI/MST) or saved (MSI/MST only), the entry will be converted and the actual resources will be built. RayPack preserves all file sequences, so that the newly created MSI can be easily used as a target image for patching.

Editing Folder Properties

Once a folder has been created as part of a packaging projects directory structure, its properties can be adjusted.

To do so, users should:
- **right-click** the folder within the right window of the package file browser.
  - o To edit the folder properties within the Visual Designer interface, users select the **Properties** option from the **context menu**.
  - o To edit the folder properties directly within the resembling row of the `Directory` table of the **TABLES** view of the **Advanced** mode, users select **Go to row** from the **context menu**.
- **left-click** the folder within the right window of the package file browser and use the key combination **Alt** + **Enter**.

The **FOLDER PROPERTIES** dialog is displayed and reveals three tabs of property groups for manipulation.
- **GENERAL**
- **CREATE AND REMOVE**
- **PERMISSIONS**

The GENERAL Tab

Name

The name of the folder as it appears on the target machine. It has to be unique among the folders within one directory. A folder name may not contain special characters or be empty. Internally, RayPack uses the folder name to build the folders `DefaultDir` value within the `Directory` table.

Identifier

The identifier is the unique internal pointer to the folder. The Identifier data type is a text string. Identifiers may contain the ASCII characters A-Z (a-z), digits, underscores (_), or periods (.). However, every identifier must begin with either a letter or an underscore.

This tab contains an editor of linked folders as well. Refer to the chapter *Importing files and folders on build time* for more information.

The CREATE AND REMOVE Tab

Remove on Uninstall if Empty

If this checkbox is active, RayPack adds a row to the `RemoveFile` table within the Installer database to make sure that the folder is removed when uninstalling, otherwise it will be left empty on the system.

Create on Install if Empty

If this checkbox is active, RayPack adds a row to the `CreateFolder` table within the Installer database to make sure that the folder is created during installation, even if it is empty.

The PERMISSIONS Tab

Permission handling is a common task in RayPack, occurring for files, folders, registry objects, and the like. Therefore, please refer to the general description of the *Object Permissions management* in the *Common Dialogs* section.

Using the Feature Filter

The feature filter is a drop-down that contains a list of all features available for the current project.
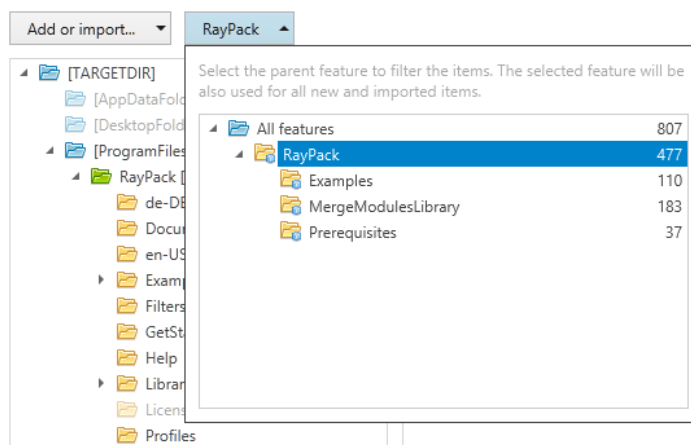
The drop-down shows the exact tree structure of features similar to the one presented in the Features tab. Each feature contains additional number, informing how many items are contained within that feature. The numbers are not cumulative - this means that the count number 477 for the *RayPack* feature means there are 477 elements directly in this feature, not including the children items.

By default, *All features* is selected. Changing the value of the drop-down affects affects the following:

- **Items filtering**
  The files and folders browser displays only elements belonging to the selected feature. This may be useful when working on certain features in the MSI package and to easily recognize which items belong to which feature.

- **Default target feature**
  As long as a feature is selected, all new items added to the view will be added to that feature. For example, to import a few files to a specific feature, select the required feature in the feature browser before importing them using a standard way. The following activities can recognize the default feature:

  - Importing file(s)

- Importing folder
- Creating new folders (all auxiliary components will be assigned to the selected features)

After a feature is selected, the drop-down will be collapsed.

Changing Default Installation Folder

The path to the installation directory determines where the application will reside once it is installed on the target device. The default value of this Installer property is `[ProgramFilesFolder]ProductName\`.

`[ProgramFilesFolder]` is a predefined folder wild-card and will be resolved at runtime to match the actual target device Program Files Folder, which depends on the operating system running on that device. Take a look at the Predefined folders section within the Files and folders chapter for more information.

In order to keep the later target package as generic as possible, it is highly recommended not to use an absolute path value (such as `C:\Program Files (x86)\RayPack\`) as Installation directory.

The value for the installation directory has to be changed by using the context menu option **Set as install directory**.

Registry

The Registry view is designed to manage the transfer of registry keys and values from the package resources to the target machine during run-time. In order to provide a user interface that does both, i.e. make it easy to put those objects into the package and at the same time offer an elaborate set of manipulation options; make sure that they arrive and behave exactly as expected on the target machine.
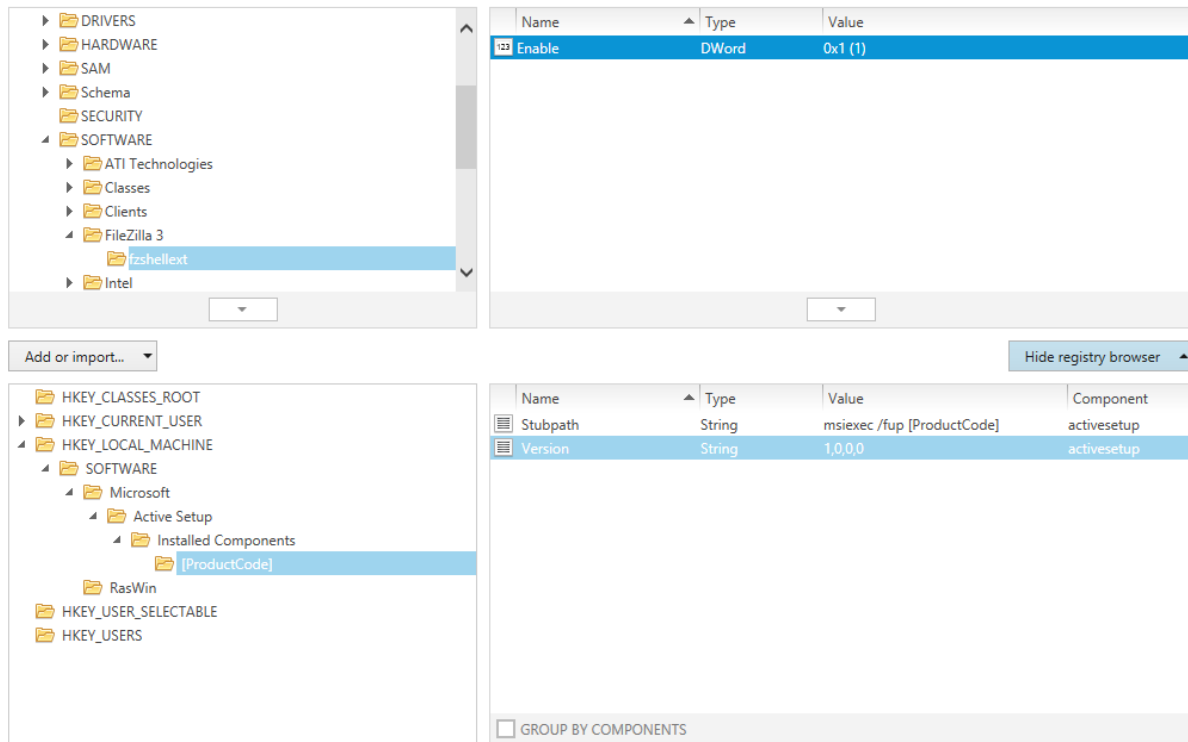
The Package Registry Browser

Within the Visual Designer view Registry, the main interface component is the **package registry browser**, which consists of two windows. On the left it contains a tree structure of hives and keys that are already available within the packaging project. Once one of those keys is selected, the details window on the right hand side presents the values which are stored within the key.

The following sections provide an overview of the most important interface features designed to improve the user experience and efficiency of the package registry browser:

Group by Components

The values within one key of a projects registry content structure may be wrapped in one common component, but usually they are not. In order to quickly determine what value belongs to which component, the **GROUP BY COMPONENTS** checkbox can be used. If it is activated, the value list is grouped by components. The arrow left of the component group header expands and collapses the list of values for that specific component to even improve the clarity of the grouped list.

Bulk Operations on Registry Contents

Since the amount of required registry objects within packaging projects may grow to an enormous extend, RayPack offers to import or export registry contents. These bulk operations support the handling of larger registry contents and significantly save time spent on standard packaging project tasks.

Sort by Property

The list of values stored within a specific key is displayed in a table with the columns Name, Type, Value and Component. Use the column headers to sort the list by one of these criteria (ascending or descending).

Go to Row

Each object that is displayed within the left registry tree structure or right value inspector window of the package registry browser is a representation of one specific row of the `Registry` table of the Installer database. With a right-click on the object and the selection of the Go to Row option from the context menu, RayPack switches to the **TABLES** view of the Advanced mode and focuses the related table row for the key or value object.

Browse the Registry Tree by Keyboard

Besides simply expanding and collapsing sub-tree structures with a mouse-click on the arrow icons left of a registry key name, users can also navigate by using their keyboard:

- To move vertically to the next adjacent registry tree node, use the Up and Down arrow keys
- To expand a node, use the Right arrow key
- To collapse a node, use the Left arrow key

Further keyboard shortcuts are available as well:
- Hit Insert to add a new key to the currently selected one
- Hit Delete to remove the currently selected key
- Hit F2 to rename the currently selected key
- Hit Alt+Enter to open the properties editor dialog for the selected key

The System Registry Browser

The first phase of working on registry contents is to import external resources into the packaging project. During this initial phase, users will often work with the **system registry browser**. This is another set of two adjoining windows for a registry key tree on the left and a value listing on the right. Whilst the package registry browser contains registry objecs present within the packaging project, the system registry browser is an exact representation of the registry contents on the packaging machine that runs RayPack.
See *Add registry keys and values* for details.

Key and Value Object Manipulation

Once registry objects are available within the project, they need to be enriched with specific settings in order to control their behavior at Installer run time and during their presence on the target device. This includes settings regarding permissions, feature and component relations, types, and the like.
See *Edit registry key properties* and *Edit registry value properties* for details.

Registry keys and values are represented by items within the Installer database table Registry. Switch to the **TABLES** view of the *Advanced mode* to access the `Registry` table directly.

Adding Registry Keys and Values

Within the Visual Designer mode, adding registry objects to a packaging project may be done by manually creating new ones inside the existing structure or by *importing existing objects from external resources*, such as local volumes, network shares, or flash drives.

The *default template for new packaging projects* creates a basic structure of hives to start from:
- `HKEY_CLASSES_ROOT`
- `HKEY_CURRENT_USER`
- `HKEY_LOCAL_MACHINE`
- `HKEY_USER_SELECTABLE`
- `HKEY_USERS`

Keys can only be created as child nodes to those predefined parent hives.

To Manually Add a Single Key

- **Right-click** the designated parent node and select **New key** from the **context menu**
  or
- **Left-click** the designated parent node and hit **Insert** on your **keyboard**

Either way, a new registry key is created within the left window of the package registry browser interface, and immediately set into the direct inline editing mode, which allows users to change the name of the key by simply starting to type it.

By hitting **Enter** on the keyboard, the new key name is evaluated regarding the required compliance with the **naming convention for registry keys**:
- Key names consist of one or more printable characters (ASCII characters A-Z (a-z), digits, underscores (_), or periods (.)).
- Key names may not begin or end with a backslash (\).
- Key names are not case sensitive.
- Key names have to be unique among the siblings directly below the same parent key.
- Key names are not localized into other languages, although values may be.

Manually Add a Sub-tree of Registry Keys

It is a common task to build a structure of nested registry keys to create an explicitly stated result. To achieve that with the least possible effort, RayPack allows packagers to enter qualified key structures in one single step:

To create a sub-tree of registry keys, type the key names separated by backslashes (`\`) when the direct inline editing mode for registry keys is active.

To create a key called RayPack within `HKEY_LOCAL_MACHINE` that is nested into a Raynet key below Software:

1. Right-click `HKEY_LOCAL_MACHINE`
2. Select **New key** from the context menu
3. Type "`Software\Raynet\RayPack`"
4. Hit Enter

> **Note:**
> Please note that the registry tree depth is restricted to 512 levels.

To Manually Copy a Key from the Local System

1. In order to be able to copy registry contents from the local system, users have to display the system registry browser.
   To do so, the **Show registry browser** button at the right, upper corner of the Registry view has to be clicked.
   To close the system registry browser, use the Hide registry browser button below the system registry browser windows.

2. Within the left tree structure explorer window of the system registry browser:
   Select the key that has to be copied into the packaging project registry contents.

3. Click on the arrow button at the bottom of the tree structure explorer window of the system registry browser.

4. A copy of the registry key (including all sub-keys and values it contains) is created immediately.
   Please note that the new key is inserted at the very same position its master holds on the local system.

To Manually Add a Registry Value

Users can go several ways to trigger the creation of a value inside a key:

- Within the left window of the package registry browser, **right-click the parent key**, select **New Value** from the context menu, and pick the desired **value type** from the sub-menu.
- Once a registry key is selected within the left window of the package registry browser, **right-**

click somewhere within the right window, select **New Value** from the context menu, and pick the desired **value type** from the sub-menu.

The new value is added to the set of already existing ones in the right package registry browser window with a default name depending on the value type (e. g. `New DWORD value`). As known from new keys, a new value is immediately set into direct inline edit mode, which allows users to start typing the desired value name right away.

By hitting **Enter** on the keyboard, the new value name is evaluated regarding the required compliance with the **naming convention for registry values**:

- Value names are stored in the Installer database table Registry as formatted column values:
  - o Text string that is processed to resolve embedded property names, table keys, environment variable references, and other special substrings.
  - o Square brackets (`[ ]`) or curly braces (`{ }`) with no matching pair are left in the text.
  - o A name like `[propertyname]` is replaced by the value of the property. If `propertyname` is not a valid property name, then the substring resolves as blank.
- The value name is NULL for default values, but may not be NULL for any other value type.
- Value names may contain any combination of alphanumeric and special characters.

For further details regarding formatted column types or the `Registry` table within Installer databases, please refer to the MSDN documentation.

Once a registry value is created within the Visual Designer view Registry, users may *open its properties for edition* by either double-clicking the row of the value within the right window of the package registry browser or by right-clicking it, and selecting Properties from the context menu.

To Manually Copy a Value from the Local System

1. In order to be able to copy registry contents from the local system, users have to display the system registry browser.
   To do so, the **Show registry browser** button at the right, upper corner of the Registry view has to be clicked.
   To close the system registry browser, use the Hide registry browser button below the system registry browser windows.

2. Within the left tree structure explorer window of the **package registry browser**:
   Select the **target key** into which the values from the local system should be copied to.

3. Within the left tree structure explorer window of the **system registry browser**:
   Select the **source key** that includes the value that has to be copied into the packaging project registry contents.
   (Use Control on the keyboard to multi-select values)

4. Click on the arrow button at the bottom of the right value explorer window of the system registry browser.

5. A copy of the registry value is created immediately.

Once a registry value is created within the Visual Designer view Registry, users may *open its properties for edition* by either double-clicking the row of the value within the right window of the package registry browser or by right-clicking it, and selecting Properties from the context menu.

Importing Registry Contents

Managing packaging project registry contents may include the handling of severe registry key and value amounts. In order to enable packagers to get that done the fast and easy the possible, RayPack includes bulk operation functionality for registry contents.

To import registry contents from a `.reg` file into a packaging projects Registry table:

1. Within the Visual Designer view **Registry**, click the **Add or import...** button at the left hand side of the details pane. Select **Import** from the options menu.

   As an alternative, it is also possible to right-click a key from the tree structure and select **Import entries** from the **context menu**.

2. The **Select Registry File** dialog allows browsing of the local system for a `.reg` file to import.

3. Once the desired file is selected, click **Open** to start the import procedure:

RayPack analyses the contents given in the `.reg` file and adds them where applicable. As soon as the import is finished, an info dialog is displayed, showing the number of imported registry entries as well as optional details.

The newly imported registry entries are stored within an automatically generated component, which is named `RayPackRegistryImportComponent`. If several import procedures are executed for the same packaging project, an incrementing index value is added to the component names, e. g., `RayPackRegistryImportComponent_1`.

> **Note:**
> Importing already existing values creates additional copies of them, there is no mechanism to replace existing registry contents by import.

Exporting Registry Contents

RayPack allows to export the currently available content of a packaging projects `Registry` table. To trigger the registry export into a .reg file:

1. Within the Visual Designer view **Registry**, right-click anywhere within the left window of the package registry browser.

2. Select **Export all entries** from the context menu.

3. The **Select Registry File** dialog allows browsing of the local system for the destination of the `.reg` file generated by the export routine.

4. Once the desired location and file name are selected, click **Save** to start the export procedure:

RayPack analyzes the registry contents and translates them into the structure required by the `.reg` file format. As soon as the export is done, users can pick the `.reg` file from the defined target location and transfer the registry contents to other machines with simple copy & paste operations on that file.

> **Note:**
> Exporting registry contents removes the additional registry control information RayPack used to organize the registry contents within the packaging project. Therefore, if duplicate entries had originally been stored within individual components, the information regarding which registry object was originally stored in what component is lost after the export.

Renaming a Key or Value

The PackDesigner mode Visual Designer contains a Registry view with all contents stored within a packaging projects `Registry` table. To rename a key or value by using the Registry view interface, users have the following options at hand:

Rename a Registry Key

- Within the package registry browser, **right-click** any key, and select **Rename** from the **context menu**.
- Within the package registry browser, **left-click** any key, and hit **F2** on the keyboard.

  Either way, the key name is set into direct inline edit mode, and starting to type immediately changes the name.
  Hit **Enter** on the keyboard to trigger the automatic input validation procedure, and save the new key name.
  Use **Escape** to restore the former key name.

- Within the package registry browser, **right-click** any key, select **Properties** from the **context menu**, and modify the **Name** property within the now displayed properties dialog.

  Click **OK** or **Apply** to trigger the automatic input validation procedure, and save the new key name.

> **Note:**
> Please make sure to follow the *naming conventions for registry key names*.
> Renaming root hives (e. g. `HKEY_CLASSES_ROOT`) is not possible.

Rename a Registry Value

- Within the package registry browser, **right-click** any value, and select **Rename** from the **context menu**.
- Within the package registry browser, **left-click** any value, and hit **F2** on the keyboard.

   Either way, the value name is set into direct inline edit mode, and starting to type immediately changes the name.
   Hit **Enter** on the keyboard to trigger the automatic input validation procedure, and save the new value name.
   Hit **Escape** to restore the former value name.

- Within the package registry browser, **right-click** any value, select **Properties** from the **context menu**, and modify the **Name** property within the now displayed properties dialog.

   Click **OK** or **Apply** to trigger the automatic input validation procedure, and save the new value name.

> **Note:**
> Please make sure to follow the *naming conventions for registry value names*.
> Renaming default values is not possible.

Removing a Key or Value

The PackDesigner mode Visual Designer contains a Registry view with all contents stored within a packaging projects `Registry` table. To remove a key or value by using the Registry view interface, users have the following options at hand:

Remove a Registry Key

- Within the package registry browser, **right-click** any key, and select **Remove** from the **context menu**.
- Within the package registry browser, **left-click** any key, and hit **Delete** on the keyboard.

Either way, the remove procedure is triggered. Once a key is removed, it is lost from the packaging project - along with all keys and values it contained.

Remove a Registry Value

- Within the package registry browser, **right-click** any value, and select **Remove** from the **context menu**.
- Within the package registry browser, **left-click** any value, and hit **Delete** on the keyboard.

Either way, the remove procedure is triggered. Once a value is removed, it is lost from the packaging project.

Confirm Registry Object Removal

Since an accidental object removal could lead to substantial issues, RayPack displays a **Confirm** dialog before a registry object is actually removed. Use the **REMOVE** button within that dialog to confirm the activity. If **DO NOT REMOVE** or **CANCEL** are clicked, the object is not deleted and the user returns to the Registry view without any changes.

The display of the Confirm dialog may manually be abandoned for future removal procedures by activating the "In future do not show this confirmation for delete operations" checkbox and using the REMOVE or DO NOT REMOVE button next (using the CANCEL button does not save any changes made to the status of that checkbox). However, it is not recommended to skip the confirmation step, since it may prevent users from severe data loss.

> **Note:**
> As soon as a key or value object is deleted from the Registry view, its resembling data row is marked as deleted within the affected Installer database table `Registry`.
> Once the packaging project is saved or the history of changes is cleared, the key or value object is no longer visible at all.
> Please refer to the Tables section to get more details about the object manipulation history and markup management.

Editing Registry Key Properties
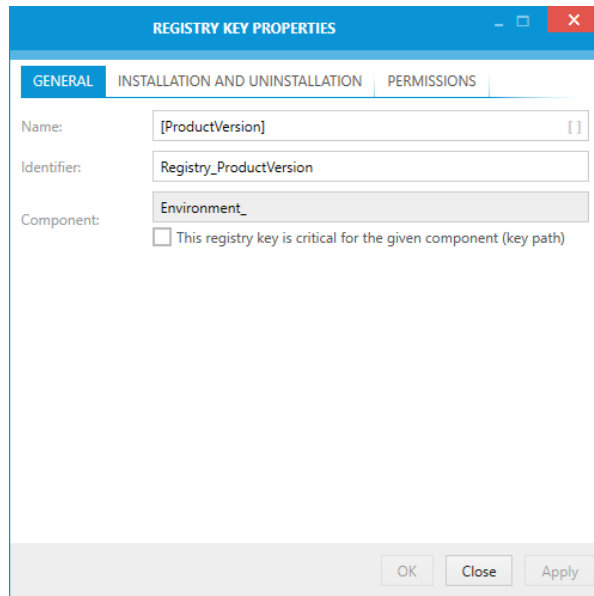
Once a key has been created below one of the standard registry hives of a packaging project, its properties can be adjusted.

To do so, users should

- **right-click** the key within the left window of the package registry browser.
  - To edit the key properties within the Visual Designer interface, users select the **Properties** option from the **context menu**.
  - To edit the key properties directly within the resembling row of the Registry table of the TABLES view of the Advanced mode, users select **Go to row** from the **context menu**.
- **left-click** the key within the left window of the package registry browser and use the key combination **Alt + Enter.**

The **REGISTRY KEY PROPERTIES** dialog is displayed and reveals three tabs of property groups for manipulation:

- **GENERAL**
- **INSTALLATION AND UNINSTALLATION**
- **PERMISSIONS**

As soon as all required changes are executed, users can either use the **Apply** button to save the changes to the registry key object and keep the REGISTRY KEY PROPERTIES dialog open, or use the **OK** button to save and close the dialog.
Clicking on the **Cancel** button closes the dialog is closed without applying unsaved changes.

The GENERAL Tab

Name

This is the name of the key as it appears on the target machine. It has to be unique among the keys within the same hive or parent key. A key name may not be empty or have a backslash (\) at its start or end. See MSDN for details regarding limitations for values of RegPath columns.

Identifier

The identifier is the unique internal pointer to the key. The Identifier data type is a text string. Identifiers may contain the ASCII characters A-Z (a-z), digits, underscores (_), or periods (.). However, every identifier must begin with either a letter or an underscore.

Component

A key always has to be related to a specific component. As soon as the first key is added to one of the standard registry hives, RayPack creates a component for that hive and automatically adds any key to the component of its parent hive:

- Keys within `HKEY_CLASSES_ROOT` are added to component `Registry_HKCR`
- Keys within `HKEY_CURRENT_USER` are added to component `Registry_HKCU`
- Keys within `HKEY_LOCAL_MACHINE` are added to component `Registry_HKLM`
- Keys within `HKEY_USER_SELECTABLE` are added to component `Registry_HKMU`
- Keys within `HKEY_USERS` are added to component `Registry_HKU`

> **Note:**
> The component property cannot be modified via the interface provided within the Visual Designer mode. However, manipulation is always possible by manually changing the content stored within the `Registry` table of the Installer database. Please refer to the topic about the Advanced Mode **TABLES** view for further details.

Key Path

If a registry key is marked to be the key path of a component, the existence of the key path determines the health state of that component. If the registry key does not exist on the target machine, the Installer's Repair functionality assumes that the component needs to be repaired.

The INSTALLATION AND UNINSTALLATION Tab

RayPack offers four standard behaviors for registry key handling during installation and uninstallation routines. The name column within the Registry table of the key is used to store the behavior information. Each of the non-standard options below is symbolized with a specific first letter for that column:

- Automatic
  This is the default option which is set for new registry keys. Use it if no special handling is required.

- Install only (+)
  This key is not used during uninstallation, and is only created during installation, if not already present on the target machine.

- Uninstall entire key (−)

Using this option causes the uninstallation sequence to delete the key even if it is not empty. During installation, the key is not used.

- Install if absent, uninstall if empty (*)
  This option handles the key during installation as if a + would have been set, but additionally removes it from the target machine during uninstallation if it is empty. If any sub-keys or values are still residing inside the key when the uninstallation is triggered, the key itself is not handled due to its own behavior settings.

> **Note:**
> The installer removes a registry key after removing the last value or sub-key under the key. To prevent an empty registry key from being removed when uninstalling, write a dummy value under the key needed to be kept and enter + in the Name column. If * is in the Name column, the key is deleted with all of its values and sub-keys, when the component is removed.

The PERMISSIONS Tab

Permission handling is a common task in RayPack, occurring for files, folders, registry objects, and the like. Therefore, please refer to the general description of the *Object Permissions management* in the *Common Dialogs* section.

Editing Registry Value Properties

Once a value has been created within one of the registry keys of a packaging project, its properties can be adjusted.

To do so, users should
- right-click the value within the right window of the package registry browser.
  - To edit the value properties within the Visual Designer interface, users select the **Properties** option from the context menu.
  - To edit the value properties directly within the resembling row of the Registry table of the **TABLES** view of the Advanced mode, users select **Go to row** from the context menu.
- left-click the value within the right window of the package registry browser and use the key combination **Alt + Enter.**

The **REGISTRY VALUE PROPERTIES** dialog is displayed with its single **GENERAL** tab.
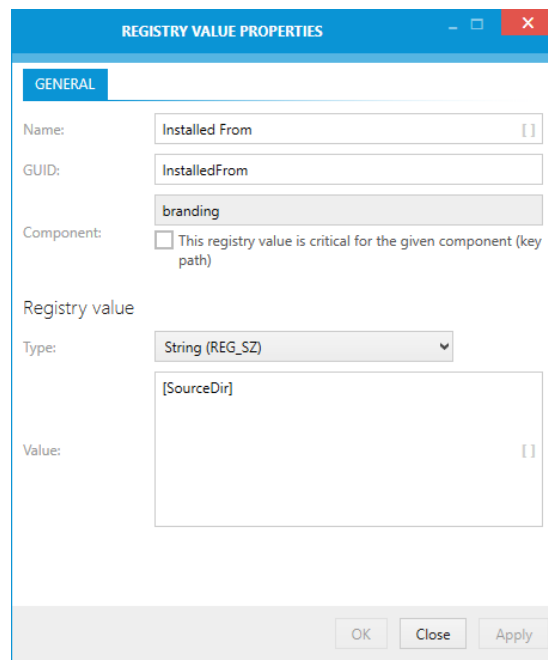
As soon as all required changes are executed, users can either use the **Apply** button to save the changes to the file object and keep the REGISTRY VALUE PROPERTIES dialog open, or use the **OK** button to save and close the dialog.
Clicking the **Cancel** button closes the dialog without applying unsaved changes.

The GENERAL Tab

Name

This is the name of the value as it appears on the target machine. It has to be unique among the values within the very same parent key. A value name may not be empty for any value type but default. Registry value names are physically stored within a Formatted database table column. Therefore the usage of placeholders for MSI properties in brackets (e. g. [MyProperty]) is provided. See MSDN for details regarding limitations for values of Formatted columns.

> **Note:**
> Registry keys and values are both stored within the Registry table. They only differ in the existence and interpretation of specific columns.
> For example:
> If a row within the Registry table represents a key, the string within the name column is scanned for specific symbolic letters, which control the behavior of the key during installation and uninstallation (compare Edit registry key properties).
> If a row within the Registry table represents a value, then the name may not be empty for all values types but default.

Identifier

The identifier is the unique internal pointer to the registry object. The Identifier data type is a text string. Identifiers may contain the ASCII characters A-Z (a-z), digits, underscores (_), or periods (.). However, every identifier must begin with either a letter or an underscore.

Component

A registry value always has to be related to a specific key and along with that to a specific component. As soon as the first key is added to one of the standard registry hives, RayPack creates a component for that hive and automatically adds any key or value stored inside to the component of its parent hive:

- Values within `HKEY_CLASSES_ROOT` are added to component `Registry_HKCR`
- Values within `HKEY_CURRENT_USER` are added to component `Registry_HKCU`
- Values within `HKEY_LOCAL_MACHINE` are added to component `Registry_HKLM`
- Values within `HKEY_USER_SELECTABLE` are added to component `Registry_HKMU`
- Values within `HKEY_USERS` are added to component `Registry_HKU`

Key Path

If a value is marked to be the key path of a component, the existence of the key path determines the health state of that component. If the value does not exist on the target machine, the Installer's Repair functionality assumes that the component needs to be repaired.

Registry Value

Type
Each registry value must have one out of 6 available types:

- default
  The default value type has a special standing, as there can only be one default value per parent key, and its properties may be used for key behavior manipulation.
  Once a default value has been added to a key, only its actual value and key path settings are modifiable. Internally, the default type is usually stored as String, since the actual data value may be used to point to external resources, such as the path to an executable for an advertised component.

- String (`REG_SZ`)
  This type is a standard string, used to represent human readable text values.

- 32-bit integer (`REG_DWORD`)
  This type represents data by a four byte number and is commonly used for boolean values, such as "0" - disabled, or "1" - enabled. Additionally, many parameters for device driver and services are of this type.
  When this type is active, the integer value may be interpreted as hexadecimal or decimal value. Please activate the required option.

- Binary (`REG_BINARY`)
  The value is interpreted and stored as a hexadecimal value. Therefore, only digits and the letters a-f are valid contents of the data value.

- Expandable string (`REG_EXPAND_SZ`)
  This type is a null-terminated expandable data string. It may be a string containing a variable to be replaced when called by an application. This type is only available on operating systems using an advanced registry editor such as REGEDT32

- Multistring (`REG_MULTI_SZ`)
  This type contains multiple double null-terminated lines of string values. Click the ADD button below the data display to create a new line. Users have to select whether the given data is appended, prepended or used as replacement for existing values of the registry value data content.
  By selecting one of the value rows and clicking REMOVE SELECTED, the row is deleted from the data row collection.

  Please note that this type will only result in a multi-string value if there is more than one value row or the action attribute is set to 'Append' or 'Prepend'. Otherwise, a standard string value (REG_SZ) will be created.

For values that are not default ones, the type can be switched freely. The current value is automatically transformed into the new types data schema.

Value

This is the actual content stored within the registry value. The domain and input controls of the value depend on the chosen value type (see paragraph above).

Registry Operations

Registry operations are special actions that are performed by the installer engine when the package is installed or uninstalled. RayPack contains wizard interfaces to allow packagers to easily define RemoveRegistry operations, which remove registry keys or values on install, as natively supported by the Windows Installer technology.

In order to define a new registry operation from the Visual Designer mode interface, users have to right click any registry key or value that is visible within the Registry view, and select **New operation** from the context menu. The sub-menu allows to pick the option "Remove registry on install...".

Registry operations are not shown as items within the Registry view. In order to review, modify or remove them, users have to switch to the Advanced mode: The Operations tab ob the COMPONENTS view shows all operations registered for a specific component.

Note:
The very same wizard is available from the Advanced mode as well as the Visual Designer mode:

- From the Visual Designer mode: Call the Registry view, select any registry key or value, and right-click to get access to the context menu that allows to trigger the creation of a new Remove registry operation.
- From the Advanced mode: Call the COMPONENTS view, select the Operations tab, and right-click to get access to the context menu that allows to trigger the creation of a new Remove registry operation.

As soon as the Registry operation wizard is invoked, it is ready for detailed operation definitions. The following steps are required to actually create the new operation:

Step 1: Key

Note:
If the wizard is called from the Registry view, it is already loaded with the properties of the registry object that was used to trigger the RemoveRegistry operation creation.
If the wizard is called from the COMPONENT view tab Operations, all properties have to be defined manually.

Registry root
Select the affected registry hive from the predefined list of options:
- HKEY_LOCAL_MACHINE
- HKEY_CURRENT_USER
- HKEY_CLASSES_ROOT
- HKEY_USERS

Key
Specify the path to the key that contains the registry items which have to be removed. Please be aware that backslashes have to be used to distinct the registry tree levels, and that blanks and special characters are not allowed within this definition string.

Remove all sub-keys and values
or
Remove specific value by name
Once the parent key is determined, users have to define whether the whole key, including all sub-keys and values has to be removed, or if only a specific value from the key container should be removed. If the Remove by name option is selected, the Name input field has to be filled with the name of the single value that is about to be deleted at package run-time.

Component
The RemoveRegistry operation has to be attached to a component to control and trigger its execution during install. Please select whether:

- A new component has to be created to contain the operation.
  To do so, activate the upper option of the Component radio control. RayPack will automatically set a default name for the new component.

- The operation should be added to an existing component.
  To do so, activate the lower option of the Component radio control. It is possible to either keep the pre-selected component, to select another one from the selector control, or to use the common Select Component dialog to define the parent component object. Within this dialog it is also possible to create a new component manually, which allows to define the name of the component, instead of letting the internal RayPack logic define it automatically as provided by the option described above.

Step 2: Summary

Use the summary page to check the correctness of the operation properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the operation
- If changes are due, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the summary page again.

Step 3: Finished

Once the new operation has been created, the wizard can be closed by using the **Finish** button at its lower right corner.

Using the Feature Filter

The feature filter is a drop-down that contains a list of all features available for the current project.
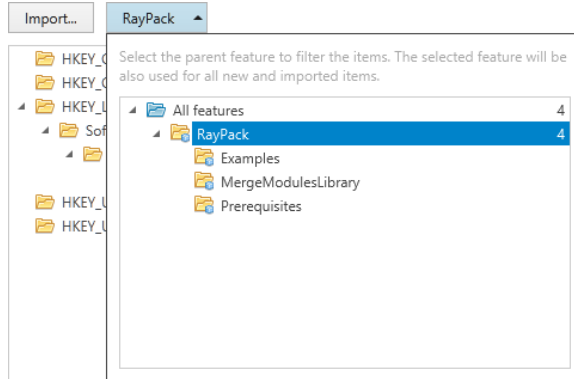
The drop-down shows the exact tree structure of features similar to the one presented in the Features tab. Each feature contains additional number, informing how many items are contained within that feature. These numbers are not cumulative.

By default, *All features* is selected. Changing the value of the drop-down affects affects the following:

- **Items filtering**
  The registry browser displays only elements belonging to the selected feature. This may be useful when working on certain features in the MSI package and to easily recognize which items belong to which feature.

- **Default target feature**
  As long as a feature is selected, all new items added to the view will be added to that feature. For example, to import a few files to a specific feature, select the required feature in the feature browser before importing them using a standard way. The following activities can recognize the default feature:

    - Importing registry key(s)
    - Creating new keys
    - Creating new values

Shortcuts

Software applications do not only have to be installed on target machines, but also thoroughly prepared for user access and execution. Therefore, shortcuts are provided per application, pointing to a specific file or software feature. Adding the right shortcuts may decide whether an application is easy to access or never found by users of the target machine at all.

Shortcuts may be added, for example, to the start menu or desktop. Most vendor delivered installation resources come with pre-defined shortcuts on both. How these shortcuts are treated

is usually part of the installation description and in accordance with best practices defined in an enterprise-wide standard for all applications and devices.
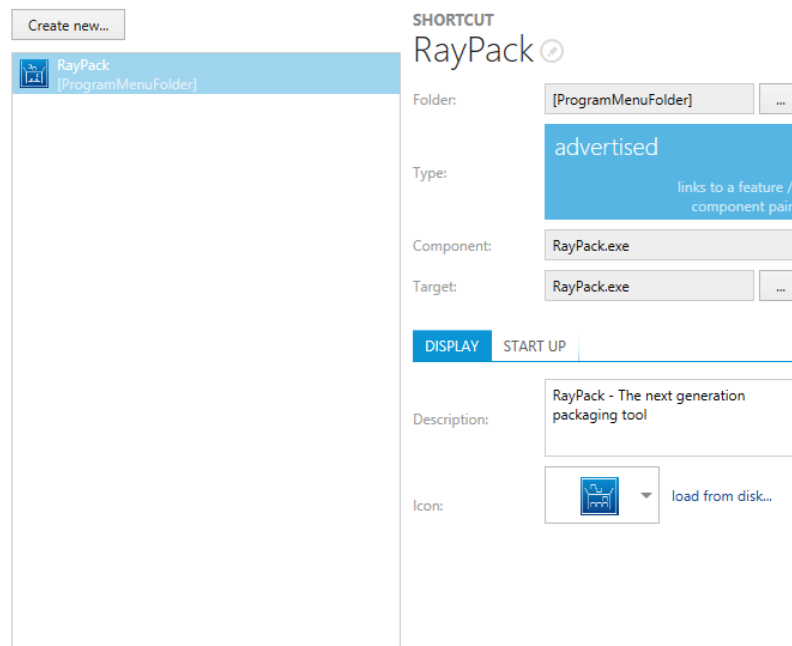
The following sections describe how the RayPack Visual Designer mode supports packagers regarding common shortcut management tasks:

- Create a shortcut
- Rename a shortcut
- Remove a shortcut
- Edit a shortcut

The Shortcut View Interface

The Shortcut view is split vertically. On the left-hand side there is a list box with all shortcuts currently available within the packaging project and a button to create new ones. As soon as one of those existing items is selected, the details pane on the right-hand side displays shortcut properties and at the same time makes them available for direct modification.

Go To Row

Internally, shortcuts are stored within the Shortcut database table. Users may jump to the data row of any given shortcut by right-clicking it in the list box within the Visual Designer view Shortcuts and selecting Go To Row from the context menu. RayPack switches to the Advanced mode and loads the content of the Shortcut table, the called data object already in focus. Please refer to the chapter regarding the TABLES editor for more information about the highly responsive Advanced mode interface and available options.

Use the back button at the upper left corner of the RayPack application window to return to the Visual Designer mode view Shortcuts.

Add a Shortcut

In order to add a shortcut to a packaging project, go to the Shortcut view of the Visual Designer mode.
Click the **Create new** button, which is available in the upper left corner of the views content area.
The **Add Shortcut** wizard is displayed.
Work your way through the steps of the wizard to define all required properties for the new shortcut.

At any time, using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, enables navigation within the already processed steps.
To exit the wizard without creating a new object, use the **Cancel** button, also located at the bottom of the wizard dialog.

Step 1: Target Type

Shortcuts within a package may not only point to an internal resource, which is brought to the target device during the installation of the package, but to any resource already available on the target device. In this first step, users have to make a primary decision regarding this shortcut property.

As the common target type actually is a **file within this package**, this option is preselected by default. The currently active option has a check mark in the upper right corner of its description box. Hover over the options and click anywhere inside the highlighted option value box to switch the active setting to that option. The check mark is immediately transferred to that description box.

Click **Next** to proceed or **CANCEL** to exit the shortcut creation.

Step 2: Target Selection

The interface of this step depends on the option selected in the previous one:

File within this package
RayPack displays the standard package file browser interface, which allows packagers to easily browse for the desired target file. Within the file management interface, users may not only pick one of the already added packaging project resource files, but also import one for direct usage.

As soon as a file is selected from within the display area on the right-hand side or the package file browser interface, the **Next** button becomes available.

> **Be aware:**
> When files (or folders) are imported into the packaging project during this wizard step, using the **CANCEL** button at a later time does not automatically remove those files again. Please make sure to remove them via the controls of the Files & Folders view if required.

File outside of this package
The shortcut target is specified by the **path** to its location and the **name of the file** that has to be invoked on the target machine.

The value given for the **folder property** may be a real path, such as `C:\Program Files (x86)\RayPack\` or a dynamic reference defined by the identifier of a known directory on the target machine, such as `INSTALLDIR`. RayPack cannot know the local file system of the target device, thus the entered folder value is not checked for validity.

The second option is usually the more useful one, since hardcoded path values may become invalid due to the actual architecture or operating system on the target machine. Therefore, it is recommended to click on the **browse** button **[...]** placed at the right hand side of the Folder input field, and use the interface of the **SELECT A FOLDER** dialog to pick the matching folder identifier. This placeholder is replaced at Installer run time to point to the correct folder according to the actual properties of the target device.
Please refer to the Select a folder topic within the Common Dialogs section for details regarding the interface handling of the **SELECT A FOLDER** dialog.

The **file name** value must contain the name of a file that is available on the target machine. If the given name does not point to a reachable file, the end user will not be able to successfully execute the shortcut.

Packagers should keep in mind that executable files need to be run in the context of the currently logged on user. Therefore pointing to files that are usually restricted from general access needs to be double-checked.

Folder and file name are automatically linked to build a command line that is actually executed when the end user clicks on the shortcut once it has been placed on the target machine. As soon as both values are given, the **NEXT** button becomes available.

Step 3: Advertising

Shortcuts may either be advertised or not.

Advertised shortcuts, which are the recommended ones to use within packaging projects, check if their target is actually usable before it is invoked. This means that the target of the shortcut is checked for the availability of all required critical files. If any of them is missing or corrupt, the MSI Repair functionality is executed on the shortcuts target component.

Advertisement will not work for resources that have simply been copied to the target machine,

have been installed by non-MSI methods, or for those files that are no executables at all. In these cases a **standard Windows shortcut** is all that can be provided: If it cannot be accessed, the target is invoked and a standard Windows error is raised.

Step 4: Display and Location

The values for **Display name** and **Shortcut description** are mere information for the end-user, explaining where on the system the shortcut points to.

Selecting a **location** controls where the shortcut object will be put on the target system, meaning where the end user will find it on their Windows interface. If the packager does not change the default setting, RayPack automatically adds the shortcut to the Program Menu Folder on the target machine. Other typical locations would be the Start Menu or Desktop folder. Click on the browse button (displayed as a button with the label "...") right of the location input field to change the default setting.

Please refer to the *Select a folder* topic within the *Common Dialogs* section for details regarding the interface handling of the **SELECT A FOLDER** dialog, which is used for the definition of the shortcut location.

	Note:
📎	According to Best Practices the Desktop folder is only populated with shortcuts if the installation description provided by the customer demands so. Even though many vendor packages create a desktop shortcut by default, it is not regarded to be very user friendly. Just imagine a device with 20 applications installed, and for every application there is a shortcut on the desktop - clean and clear orientation might be considered to look slightly different, agreed?

Each shortcut may be visually augmented with an **icon**. Packagers may either pick one of the already existing icon resources from the current packaging project or add a new icon by loading it from the local disk.

To pick an existing icon resource:
1. Users expand the icon resource selector interface with a click on the downwards arrow icon next to the standard icon.
   The list of icon resources stored within the packaging project is displayed.

2. Select the desired icon from the list on the left-hand side of the selector interface and if required, navigate to the icon index that should be used. Use the arrows pointing to the left and to the right to change the currently active icon index value.

3. Click somewhere outside the icon resource selector interface to save the current setting for the shortcut icon.

To load a new icon resource from disk:
1. Users click on the "load from disk" link at the right-hand side of the preview panel for the current shortcut icon state.

2. Within the system browser for icon selection, the desired icon resource has to be selected.

3. By clicking on the Open button the icon is loaded into the project and selected to be the icon for the new shortcut.

> **Be aware:**
> When an icon resource is loaded into the packaging project during this wizard step, using the CANCEL button at a later time does not automatically remove that icon resource again. Please make sure to remove it via the controls of the Resources view if required.

Step 5: Start-up

Command line arguments

The command-line arguments for the shortcut. Note that the resolution of properties in the Arguments column of the Shortcut database table is limited.
When properties are used within this column value, the actual value of the property must have been resolved already once the shortcut is created on the target machine.

Working directory

The folder selected for this value is displayed as the "Start in" setting within the properties dialog of a shortcut on the target system. The working directory is the default for file opening and saving operations, and the current directory used by the invoked application.

Please refer to the Select a folder topic within the Common Dialogs section for details regarding the interface handling of the SELECT A FOLDER dialog, which is used for the definition of the working directory.

Hot key

To define a hot key for the new shortcut, packagers click into the usually greyed out hot key input field. As soon as the background color of the input field has switched to white, the next combination of keys pressed on the keyboard is recorded as hot key for the shortcut object. It is possible to use a single keystroke or a combination such as Control + Shift + b.

> **Tip:**
> Hot keys should only be added to shortcuts if the customers installation description demands to do so. Conflicts between already existing hot key settings on the target device may lead to the experience of inconvenient system behavior for the end user.

Start-up mode

Decide the behavior of the target application window when the shortcut is executed. Select one of the available options for the Run property of the later Windows shortcut object:

- Normal window - the application is invoked with the standard settings saved for the application.
- Maximized window - if no (user-) specific settings override this command, the invoked application covers the whole screen.
- Minimized window - if no (user-) specific settings override this command, the invoked application resides minimized in the task bar.

> **Note:**
> Some programs override the Run property injected by shortcuts. For those applications packagers have to check whether there is a way to alter this behavior.
> The Run property works for programs and shortcuts to programs, but not for other target file types.

Step 6: Summary

Use the summary page to check the correctness of the shortcut properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the shortcut
- If changes are necessary, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst Next-ing to the summary page again.

Step 7: Finished

Once the new shortcut object has been created, the wizard can be closed by using the **Finish** button at its lower right corner. The Shortcut view is updated, and the list of existing shortcuts contains the newly created object.

Rename a Shortcut

The PackDesigner mode Visual Designer contains a Shortcut view with a list of all objects stored within a packaging projects `Shortcut` table. To rename a shortcut by using this view interface, users have the following options at hand:

- Within the list of shortcuts, **right-click** any item, and select **Rename** from the **context menu**.
- Within the list of shortcuts, **left-click** any item, and hit **F2** on the keyboard.

    Either way, the shortcut name is set into direct inline edit mode, and starting to type immediately changes the name.
    Hit **Enter** on the keyboard to trigger the automatic input validation procedure, and save the new name.
    Use **Escape** to restore the former item name.

- Within the list of shortcuts, **left-click** any item to load its properties into the **edit form** on the right hand side. The uppermost row of the details area displays the shortcut name. Double-click the name or click on the edit icon right next to it to use the Direct Value Editor for changes on the shortcut name.

The name of a shortcut is automatically defined by RayPack when the wizard for new shortcut objects calculates the properties for the new shortcut object according to the given user input.

If users manually change the shortcut name to an invalid value (e.g., any name containing special characters such as slash (`/`), colon (`:`), or question mark (`?`)), RayPack applies auto-correction methods to prevent invalid names to be saved.

> **Note:**
> The value actually stored within the Name column of the `Shortcut` database table has to follow the format restrictions for the Filename data type. If the entered name does not comply the rules for short file names, RayPack automatically transforms the value to create a mixed value of short and long file name, separated by a vertical bar (`|`)
>
> Please refer to the MSDN documentation for the Filename column type to get further details on possible values.

Remove a Shortcut

The PackDesigner mode Visual Designer contains a Shortcut view with a list of all objects stored within a packaging projects `Shortcut` table. To remove a shortcut by using this view interface, users have the following options at hand:

- Within the list of shortcuts, **right-click** any item, and select **Delete** from the **context menu**.
- Within the list of shortcuts, **left-click** any item, and hit **Delete** on the keyboard.

Either way, the remove procedure is triggered. Once a shortcut is removed, it is lost from the packaging project.

> **Be aware:**
> If files, folders, or icon resources were created during the Add Shortcut wizard execution, these additional packaging project contents are not automatically removed when the shortcut itself is deleted. Please make sure to remove them via the controls of the Files & Folders or Resources view if required.

Confirm Shortcut Object Removal

Since an accidental object removal could lead to substantial issues, RayPack displays a **Confirm** dialog before a shortcut is actually removed. Use the **REMOVE** button within that dialog to confirm the activity. If **DO NOT REMOVE** or **CANCEL** are clicked, the object is not deleted and the user returns to the Shortcut view without any changes.

> **Note:**
> As soon as an item is deleted from the Shortcut view, its resembling data row is marked as deleted within the affected Installer database table `Shortcut`.
> Once the packaging project is saved or the history of changes is cleared, the shortcut object is no longer visible at all.
> Please refer to the Tables section to get more details about the object manipulation history and markup management.

Edit a Shortcut

Once a shortcut has been created, its properties can be adjusted. To do so by using the Visual Designer mode interface of PackDesigner, users select the shortcut from the list of all shortcuts on the left-hand side of the Shortcuts view.
As soon as an item is selected, RayPack loads its properties into the edit form at the right-hand side of the Shortcuts view.

The following settings are visible and (partially) modifiable within the edit shortcut form:

Name

For details regarding the renaming procedure for shortcuts, please refer to the Rename a shortcut topic.

Folder

This is the folder in which the shortcut will be created. To change the value, use the browse button that is available right next to the read-only input field, and use the Select a folder dialog.

Type

Shortcuts can be either advertised or non-advertised. To change from advertised to non-advertised or vice versa, users have to click on the tile. All depending shortcut properties are changed along with the type setting.

Component

Whenever a shortcut is created, RayPack automatically generates a component specifically for this shortcut.

For advertised shortcuts:
The component property is not modifiable, since RayPack automatically stores advertised shortcuts in one of the packaging project default components. By default it is named `[Product Name]_AllOtherFiles` (the actual default depends on the settings of the currently used RayPack settings profile. Please refer to the Settings section for details regarding common naming conventions used within PackDesigner).

For non-advertised shortcuts:
The component property is modifiable. Click on the name of the currently selected component to display the list of component alternatives to choose from. If advanced component manipulation options are needed to define the correct component for the shortcut, then click on the browse button [**...**] right next to the selection box to display the SELECT COMPONENT dialog.

Target / Command

For advertised shortcuts: Target
Click on the browse button [**...**] right next to the read-only Target input field to select a file by using the standard package file browser interface.

For non-advertised shortcuts: Command
If a shortcut has been created directly in RayPack's PackDesigner, it will mostly look similar to something like this: `[FOLDER]file.exe`. However, there are other options for the format of this setting, e. g. something like `[#file.exe]`. In order to be able to cover the wide spectrum of possible notations, RayPack offers both - syntax suggestions for property usage (simply type an "[" into the input field to get suggested contents to pick from), and a dialog to select one of the resource files within the packaging project (click on the browse button [**...**] right next of the Command input field to select a file by using the standard package file browser interface).

Tab: DISPLAY
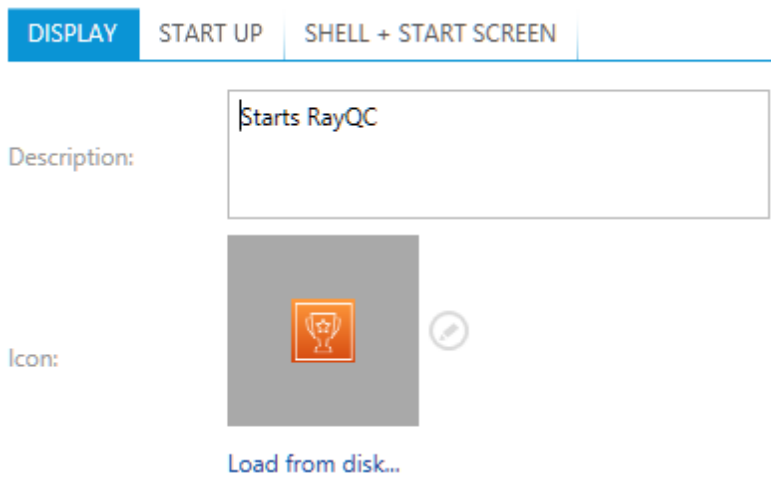
Description

The value for description is mere information for the end-user, explaining details about the shortcut. For example, where the shortcut points to on the system.

Icon

Each shortcut may be visually augmented with an **icon**. Packagers may either pick one of the already existing icon resources from the current packaging project or add a new icon by loading it from the local disk. If an icon has been added to the shortcut, a preview is shown as visible icon in the collapsed icon resource selector interface.

To pick an existing icon resource:
1. Users expand the icon resource selector interface with a click on the downwards arrow icon next to the standard icon.
   The list of icon resources stored within the packaging project is displayed.

2. Select the desired icon from the list on the left-hand side of the selector interface and if required, navigate to the icon index that should be used. Use the arrows pointing to the left and to the right to change the currently active icon index value.

3. Click somewhere outside of the icon resource selector interface to save the current setting for the shortcut icon.

To load a new icon resource from disk:
1. Click on the "Load from disk..." link under the preview panel for the current shortcut icon state.

2. Use the New Resource wizard to specify import options

3. By clicking on the **Finish** button the icon is loaded into the project

4. Click on the icon selector to choose the newly imported icon.

> **Be aware:**
> When the shortcut icon is changed, the former used icon resource is not removed from the packaging project. Nonetheless, when a new icon is loaded into the project, a new, permanent icon resource is added to the packaging project. Please make sure to manage icon resources via the controls of the Resources view if required beyond the needs of the current shortcut object manipulation.

Tab: STARTUP

Arguments

Here are the command-line arguments for the shortcut. Note that the resolution of properties in the Arguments column of the Shortcut database table is limited.
When properties are used within this column value, the actual value of the property must have been resolved already once the shortcut is created on the target machine.

Working Directory

The folder selected for this value is displayed as the "Start in" setting within the properties dialog of a shortcut on the target system. The working directory is the default for file opening and saving operations and the current directory used by the invoked application.

Please refer to the Select a folder topic within the Common Dialogs section for details regarding the interface handling of the SELECT A FOLDER dialog, which is used for the definition of the working directory.

Hot Key

To define a hot key for the new shortcut, packagers click into the usually greyed out hot key input field. As soon as the background color of the input field has switched to white, the next combination of keys pressed on the keyboard is recorded as hot key for the shortcut object. It is possible to use a single keystroke or a combination such as Control + Shift + b.

> **Note:**
> Hot keys should only be added to shortcuts if the customers installation description demands to do so. Conflicts between already existing hot key settings on the target device may lead to the experience of inconvenient system behavior for the end user.

Startup Mode

Decide the behavior of the target application window when the shortcut is executed. Select one of the available options for the Run property of the later Windows shortcut object:

- Normal window - the application is invoked with the standard settings saved for the application.
- Maximized window - if no (user-) specific settings override this command, the invoked application covers the whole screen.
- Minimized window - if no (user-) specific settings override this command, the invoked application resides minimized in the task bar.

> **Note:**
> Some programs override the Run property injected by shortcuts. For those applications, packagers have to check whether there is a way to alter this behavior. The Run property works for programs and shortcuts to programs, but not for other target file types.

Tab: SHELL + START SCREEN

This tab consist of two section: one for checkboxes controlling the behavior of a shortcut and a preview area which shows a visual representation of currently selected combination of options.
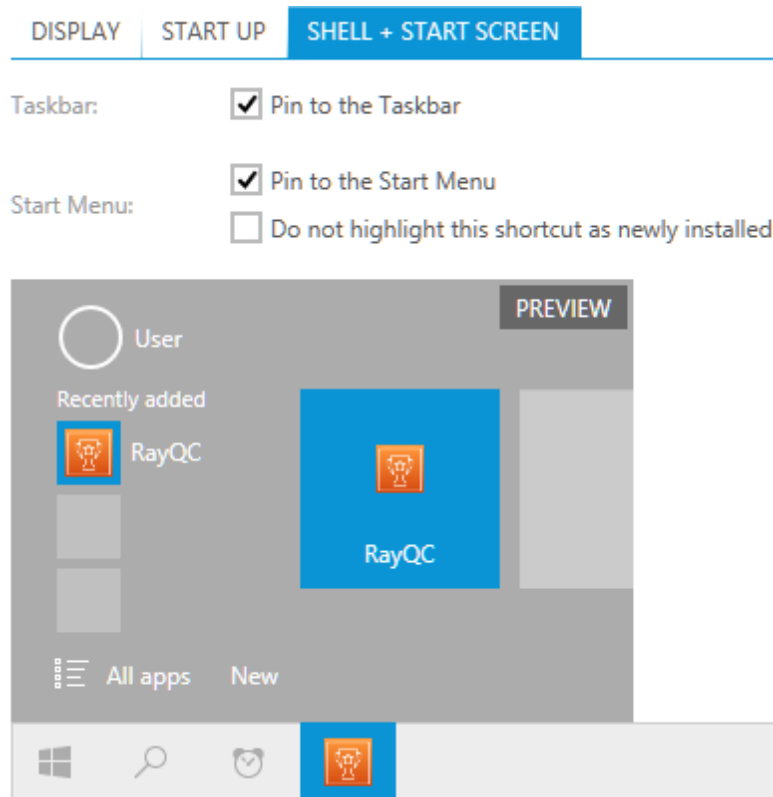
Pin to the Taskbar

Control whether the shortcut is pinned to the Windows Taskbar. This functionality is only valid for Windows 7 and newer.

> **Note:**
> Using this option requires that the shortcut is non-advertised. If you try to enable this option for an advertised shortcut, it will be automatically converted into a non-advertised shortcut.

Pin to the Start Menu

Control whether the shortcut is pinned to the Windows Start Menu. This functionality is only valid for Windows 7 and newer. Pinning to start menu is not available on Windows 8.1.

> **Note:**
> Using this option requires that the shortcut is non-advertised. If you try to enable this option for an advertised shortcut, it will be automatically converted into a non-

advertised shortcut.

Do not Highlight This Shortcut as Newly Installed

Control whether the shortcut is displayed in the **Recently added** section (naming may vary depending on version and language of the target Operating System).

> **Note:**
> This option requires Windows Installer Schema 500. If the current schema is lower than 500, RayPack will prompt for confirmation before converting the schema to the required version.

Convert Shortcut Types

An advertised shortcut is a special type of shortcut that uses an entry point of the installer engine to validate the correctness of the installation and repair it if necessary. Advertised shortcuts can be used only for files that are physically present in the package and that are keypaths of their components.

Non-advertised shortcuts can link to both, resources present in the package and resources that are only available on the target machine (for example starting `iexplore.exe`). Non-advertised shortcuts do not check the installation when they are executed.

Usually, advertised shortcuts should be the preferred wherever possible. There are certain circumstances in which the advertising mechanism is not required. RayPack provides a way to convert shortcuts between advertised and non-advertised type.

To Convert an Advertised Shortcut to a Non-advertised Shortcut

1. In the shortcut view, select the shortcut that has to be converted
2. In the sidebar view, click the ADVERTISED tile
3. The shortcut will be automatically converted to a non-advertised shortcut, the tile label will change to **NON-ADVERTISED**

To Convert a Non-advertised Shortcut to an Advertised Shortcut

1. In the shortcut view, select the shortcut that has to be converted
2. In the sidebar view, click the **NON-ADVERTISED** tile
3. The shortcut will be automatically converted to an advertised shortcut, the tile label will change to **ADVERTISED**
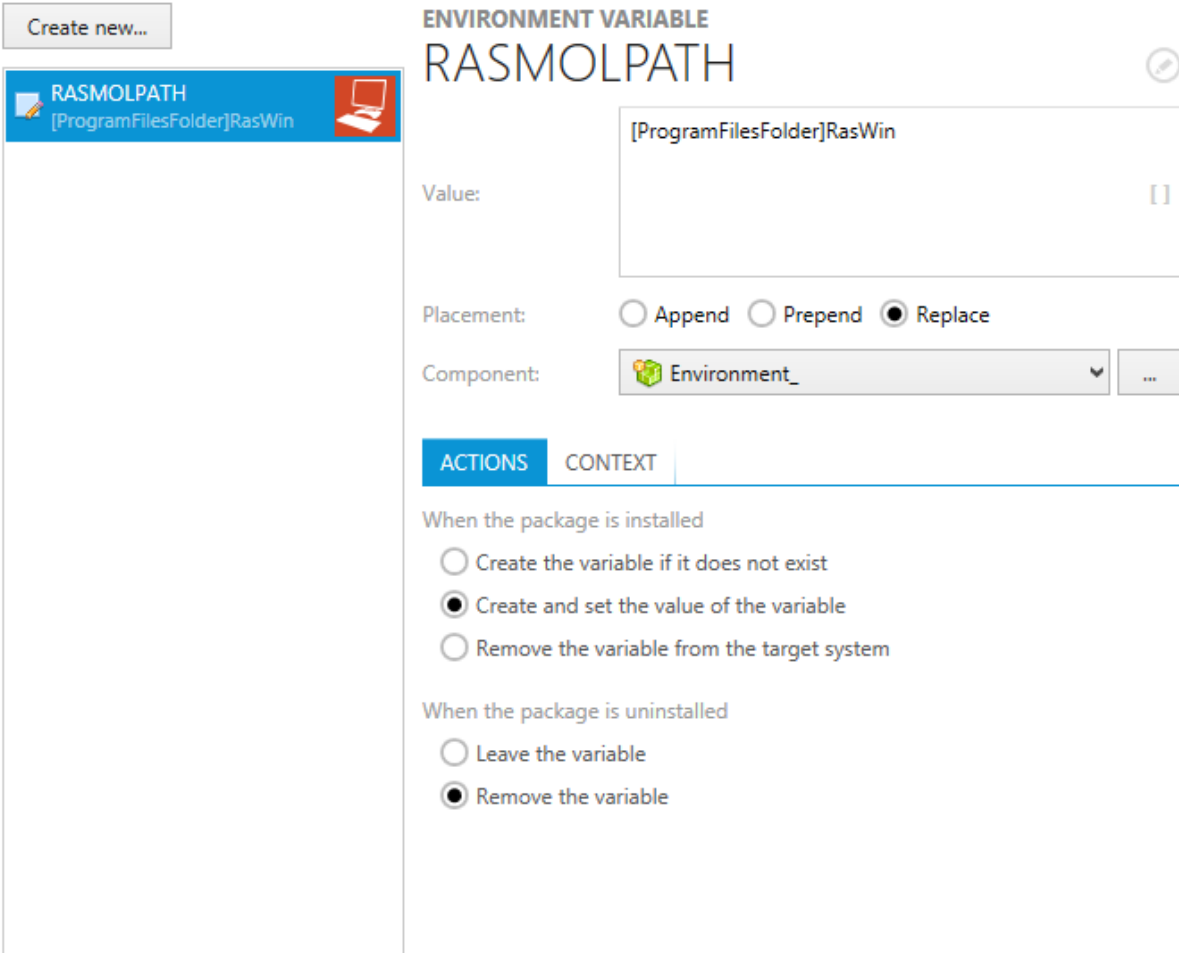
> **Be aware:**
> The conversion of non-advertised to advertised may not always be possible due to several reasons. The following conditions have to be fulfilled for successful conversion:

- The command has to resolve to any file present in the package
- The target file must be a keypath of its component

If any of the aforementioned points is not fulfilled, RayPack will display a warning message, stating that the conversion is not possible.

Environment Variables

Environment variables control the behavior of programs as they store default information for drives, paths, and the like. In MSI related packaging projects, environment variables are stored in the `Environment` table.

> **Hint:**
> The list has different icons for variables set for the current user (green icon) and system context (red icon).

RayPack's PackDesigner mode Visual Designer contains an Environment Variables view, which allows packagers to execute a set of standard functions for environment variables:

- *Add an environment variable*
- *Remove an environment variable*
- *Edit an environment variable*

The Environment Variables view is separated into a list of already added environment variable data objects on the left-hand side and a details pane with information about the currently selected environment variable data object loaded into a form for direct inline editing.

As soon as an environment variable object is right-clicked, the context menu offers the option **Go to row**, which enables to switch to the **TABLES** editor of the Advanced mode with the data row of the currently displayed environment variable object focused.

Add an Environment Variable

In order to add an environment variable data object to a packaging project, go to the Environment variables view of the Visual Designer mode.
Click the **Create new...** button, which is available in the upper left corner of the views content area.
The new environment variable wizard is displayed.

At any time, using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, enables navigation within the already processed steps.
To exit the wizard without creating a new object, use the Cancel button, also located at the bottom of the wizard dialog.

Step 1: Actions

The behavior of an environment variable data object can be manipulated from three separated angles:

When the Package is Installed

- **Create the variable if it does not exist**
  If the environment variable already exists on the target machine, no action is applied during installation.
  If it does not exist, a new environment variable is created on the target machine.

- **Create or set the value of the variable**
  If the environment variable already exists on the target machine, the action defined above (append, prepend, replace) is executed.
  If it does not exist, a new environment variable is created on the target machine.

- **Remove the variable from the target system**
  If the environment variable exists on the target machine, it is removed during installation.
  If it does not exist, no action is applied during installation.

When the Package is Uninstalled

- **Leave the variable**
  No action is applied during uninstallation.

- **Remove the variable**
  If the environment variable exists on the target machine, it is removed during uninstallation.

Step 2: Details

Environment Variable

The localizable name of the environment variable as it is visible for the end user on the target machine. Any non-empty alphanumerical string is a valid name value.

Internally, RayPack uses the Name column to store information about the behavior and target scope of an environment variable. A visual interface for the manipulation of these settings is provided once an environment variable data object is created. Therefore, it is recommended to simply set the visible display name of the environment variable as desired during this initial create procedure, and go for advanced options later when the data object is ready for edition.

Value

The value of the environment variable as it will be set on the end users target machine.

Internally, the value is saved in a formatted table column. The Formatted data type is a text string that is processed to resolve embedded property names, table keys, environment variable references, and other special sub-strings. Please refer to the official MSI documentation for details regarding the Formatted data type.

RayPack uses the Value column to store information about the behavior and target scope of an environment variable. A visual interface for the manipulation of these settings is provided once an environment variable data object is created. Therefore, it is recommended to simply set the visible value of the environment variable as desired during this initial create procedure, and go for advanced options later when the data object is ready for edition.

Placement

- **Append**
  The value entered above is added as a suffix to the value of the environment variable as it exists on the target machine during run time.
  RayPack automatically prepends the following symbols to the Value column of the Environment table to trigger this behavior: `[~];`

- **Prepend**
  The value entered above is added as a prefix to the value of the environment variable as it exists on the target machine during run time.

RayPack automatically appends the following symbols to the Value column of the Environment table to trigger this behavior: `;[~]`

- **Replace**
  The value entered above fully replaces the value of the environment variable as it exists on the target machine during run time.
  Please note that the Placement property is only available for manipulation if the behavior of the environment variable during installation is set to "Create or set" (see section on Actions above)

Step 3: Context

Per-system

System variables are related to resources and settings of any logged user on the target system.
Location in the target device registry: `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet \Control\Session Manager\Environment`

Per-user

User variables are related to resources and settings owned by various user profiles within the system. These variables usually do not refer to critical system resources or locations, which are necessary for the operating system to run.
Location in the target device registry: `HKEY_CURRENT_USER\Environment`

Step 4: Summary

Use the summary page to check the correctness of the environment variable properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the new item
- If changes are necessary, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst Next-ing to the summary page again.

Step 5: Finished

Once the new environment variable object has been created, the wizard can be closed by using the **Finish** button at its lower right corner. The Environment variables view is updated, and the list of existing variables contains the newly created object.

Remove an Environmemnt Variable

The PackDesigner mode Visual Designer contains an Environment Variables view with a list of all objects stored within a packaging projects `Environment` table. To remove an environment

variable data object, users **right-click** any item from the list, and select **Remove** from the **context menu**.

The remove procedure is triggered and executed immediately. Once an environment variable is removed, it is lost from the packaging project.

> **Be aware:**
> As soon as an item is deleted from the Environment Variables view, its resembling data row as well is deleted from the affected Installer database table `Environment`.

Edit an Environment Variable

Once an environment variable data object has been created, its properties can be adjusted. To do so by using the Visual Designer mode interface of PackDesigner, users select the environment variable item from the list at the left-hand side of the Environment Variables view.

As soon as an item is selected, RayPack loads its properties into the edit form at the right-hand side of the view.

At any time, users can go to the corresponding Installer database table row to compare the actual column content to the settings made in the Visual Designer interface.

The following settings are visible and modifiable within the edit environment variable data object form:

Name

The localizable name of the environment variable as it is visible for the end user on the target machine. Any non-empty alphanumerical string is a valid name value.

When RayPack saves the properties of an environment variable data object to the Environment table of the Installer database, prefixes will be added to the user defined value of the name property. It is recommended not to use those prefix symbols (e. g. equals (=), exclamation mark (!), plus (+), minus (−), or asterisk (*)) within the Name input field at all, but let RayPack apply them due to the settings made for the Action and Behavior properties (see below).

At any time, users can go to the corresponding Installer database table row to compare the actual column content to the settings made in the Visual Designer interface.

Value

This column contains the localizable value that is to be set as a formatted string.

When RayPack saves the properties of an environment variable data object to the Environment table of the Installer database, standardized symbols will be added to the user defined value. It is recommended not to use those prefix symbols (e. g. NULL ([~]), separator (;), plus (+), or minus (−)) within the Value input field at all, but let RayPack apply them due to the settings made for the Action and Behavior properties (see below).

Placement

- **Append**
  The value entered above is added as a suffix to the value of the environment variable as it exists on the target machine during run time.
  RayPack automatically prepends the following symbols to the Value column of the Environment table to trigger this behavior: `[~];`

- **Prepend**
  The value entered above is added as a prefix to the value of the environment variable as it exists on the target machine during run time.
  RayPack automatically appends the following symbols to the Value column of the Environment table to trigger this behavior: `;[~]`

- **Replace**
  The value entered above fully replaces the value of the environment variable as it exists on the target machine during run time.
  Please note that the Placement property is only available for manipulation if the behavior of the environment variable during installation is set to "Create or set" (see section on Actions above)

Component

Every environment variable data object needs to be assigned to a specific component. At run time, environment variable data objects are only handled when the component they are associated to is installed or uninstalled. If the parent feature of a component is not installed or uninstalled during run time, the environment variable data objects are not considered at all.

Select from the list of existing components. If the current packaging project does not contain the right component yet, please use the **NEW** button at the right-hand side of the component selection control to add a component to the project. Make sure to assign the new component to a feature, since unassigned components may lead to invalid target packages later. To assign a component to a feature, switch to the COMPONENTS view of the Advanced mode.

Tab: Actions

The behavior of an environment variable data object can be manipulated from three separated angles:

When the Package is Installed

- **Create the variable if it does not exist**
  If the environment variable already exists on the target machine, no action is applied during installation.
  If it does not exist, a new environment variable is created on the target machine.

- **Create or set the value of the variable**
  If the environment variable already exists on the target machine, the action defined above (append, prepend, replace) is executed.
  If it does not exist, a new environment variable is created on the target machine.

- **Remove the variable from the target system**
  If the environment variable exists on the target machine, it is removed during installation.
  If it does not exist, no action is applied during installation.

When the Package is Uninstalled

- **Leave the variable**
  No action is applied during uninstallation.

- **Remove the variable**
  If the environment variable exists on the target machine, it is removed during uninstallation.

Tab: Context

Per-system

System variables are related to resources and settings of any logged user on the target system.
Location in the target device registry: `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet \Control\Session Manager\Environment`

Per-user

User variables are related to resources and settings owned by various user profiles within the system. These variables usually do not refer to critical system resources or locations, which are necessary for the operating system to run.
Location in the target device registry: `HKEY_CURRENT_USER\Environment`

Properties

Properties in the Windows Installer world can be regarded as a special set of placeholders. Defined by either packagers or target machine operating systems, properties can be used within any packaging project setting that allows formatted values.

Whenever a property is used inside a formatted string value (by setting the name in square brackets, e. g., `[MyProperty]`), it is replaced with the actual value at run time.

The **naming convention for properties** is defined by the following rules:

- Property names are internally stored in an Identifier data column, which restricts the property name to contain the ASCII characters A-Z (a-z), digits, underscores (_), or periods (.). However, every identifier must begin with either a letter or an underscore.

- The visibility of a property depends on the usage of lower-case letters in its name:

  o If there are any lower-case letters, a property is always private.
  o If the property name exclusively consists of upper-case letters, it is always public.

- The MSI standard contains a set of predefined properties (such as the list of predefined system folders, mentioned in the *Files and Folders* section). It is not allowed to re-use any of those predefined property names, e. g., to add a custom public property `ORIGINALDATABASE` additionally to the private predefined property `OriginalDatabase`.
  Please refer to the MSI Property Reference documentation for a full list of predefined properties.

> **Note:**
> Private properties cannot be modified from anywhere outside the actual package context. Their values cannot be injected via command line or manipulated in any dialog or message of the UI sequence. On the contrary, public properties may be used to transfer information from a packages external environment into the internal logic.

RayPack's PackDesigner mode Visual Designer contains a Properties view, which allows packagers to execute a set of standard functions for properties:

- *Add a property*
- *Remove a property*
- *Edit a property*

The Properties view consists of a list of already added property data objects. Each item within the list is represented by a row within the Windows Installer database table `Property`. A context menu is revealed when any of the listed properties is right-clicked, offering a **Go to row** option, which enables to switch to the TABLES editor of the Advanced mode with the data row of the currently active property object focused.

> **Be aware:**
> It is not possible to use properties within the value column of other properties. In order to prevent the implementation of circular references, this can only be done by using a specific custom action type. Please refer to the section regarding CUSTOM ACTIONS in the Advanced mode to find out how to create a Set Property custom action (type 51).

Add a Property

In order to add a property to a packaging project, users go to the Properties view of the Visual Designer mode.

With a click on the **Create new...** button, the two basic property creation options become visible (and selectable):

- *Add a predefined property*
  The MSI standard defines a set of predefined properties. In order to make it faster and easier for packagers to use those predefined properties within packaging projects, RayPack contains a quick list to access them. Use this add option to pick one of the predefined properties. Please refer to the MSI Property Reference documentation for a full list of predefined properties.

- *Add a custom property*
  It is usually necessary to add properties that go beyond the default definitions of the MSI

standard. In those cases, select the simple new property procedure.

To Add a Predefined Property

1. From the options menu revealed by a click on the **Create new...** button, select **Predefined property**.
   The **PREDEFINED PROPERTIES** dialog is displayed, containing a list of all modifiable properties defined by the MSI standard.

2. Use the search field to filter the list by keyword, sort the list by clicking on any column header, or expand and collapse the groups to navigate through the total of displayed properties.

3. The left column of the property list contains a checkbox. If a property has already been added to the packaging project, this checkbox is active and set into read-only mode, since the PREDEFINED PROPERTIES dialog does not allow to remove properties from a project. Please refer to the *Remove a property* section to get details about how to accomplish that.
   Use the checkbox column to mark all predefined properties that have to be added to the current project.

4. Once the desired set of properties is defined, a click on the **OK** button at the lower right corner of the dialog adds them to the Properties view and closes the PREDEFINED PROPERTIES dialog. Clicking the Cancel button closes the dialog without making changes to the set of properties defined for the current packaging project.

To Add a Custom Property

1. From the options menu revealed, click on the **Create new...** button, select **New property**.

2. RayPack automatically adds a new item to the list of available properties.
   The name of the property is set to the default value **NewProperty**. Since property names have to be unique, an automatically incremented index is added if required (e. g. NewProperty1, NewProperty2, etc.).
   Since the default name contains lower-case letters, new properties are private by default.

As soon as a predefined or custom property has been added to the list of available properties for a packaging project, the property name and value may be modified.
Please refer to the *Edit a property* topic for details.

Remove a Property

The PackDesigner mode Visual Designer contains a Properties view with a list of all objects stored within a packaging projects `Property` table.

This view offers two methods to trigger the removal of a property:

Select the object that has to be deleted and
- Left-click it, and hit **Delete** on the **keyboard**.

- Right-click it, and select **Remove** from the **context menu**.

Removing a group of properties can be achieved as well by multi-selection. Use the Control or Shift key to select more than one property at a time. Once the desired set of selected items is gathered, apply one of the triggers named above to initiate the remove procedure.

> **Be aware:**
> If a property has already been used somewhere else within the packaging project, e. g., to set the data value of a registry value, these references are not deleted along with the property data object itself.
>
> It is recommended to remove the references that point to a property before the property itself is removed from the stock. Following this Best Practice avoids orphaned property references, which might lead to the generation of invalid target packages.
>
> To do so, use the advanced search functionality of the **TABLES** editor of the Advanced mode. Enter the property name in square brackets as search term (e. g. `[MyProperty]`) and search the whole database for that phrase. If required, a replace procedure might be executed as well. For details regarding the advanced search functionality, refer to the help topic about Search & Replace functionality.

Confirm Object Removal

Since an accidental object removing could lead to substantial issues, RayPack displays a **Confirm** dialog before a property is actually removed. Use the **REMOVE** button within that dialog to confirm the activity. If **DO NOT REMOVE** or **CANCEL** is clicked, the object is not deleted and the user returns to the Properties view without any changes.

> **Note:**
> As soon as an item is deleted from the Properties view, its resembling data row is marked as deleted within the affected Installer database table `Property`.
> Once the packaging project is saved or the history of changes is cleared, the object is no longer visible at all.
> Please refer to the *Tables* section to get more details about the object *manipulation history and markup* management.

Edit a Property

The PackDesigner mode Visual Designer contains a Properties view with a list of all objects stored within a packaging projects `Property` table.

To edit the name of a property:
- **Double-click** the property item somewhere inside the **Name** column
- **Left-click** the property item somewhere inside the **Name** column and press **F2** on the keyboard

To edit the value of a property:

- **Double-click** the property item somewhere inside the **Value** column
- **Left-click** the property item somewhere inside the **Value** column and press **F2** on the keyboard

Either way, the field that is about to be modified is immediately set into a direct inline edit mode. Users can simply start to type and the column content will be replaced by the newly typed text.

> **Note:**
> Once any cell of the table displayed within the Properties view is set into direct inline edit mode, using the arrow keys on the keyboard navigates through the list, moving the direct inline edit mode itself from cell to cell. When several properties have to be edited during one session, this method is quite quick, since the usual sequence of clicking with the mouse, entering new settings, hitting enter, re-positioning the mouse pointer, set cell into direct inline edit mode, etc. is simply cut short.

Configuring Isolation for Virtualization

Folders and registry keys have special configuration dialog which allows users to configure desired level of merging when building App-V 4.6/5.x/Thin-App projects. To trigger the settings, right click a registry key or a folder and go to the **ISOLATION** tab:
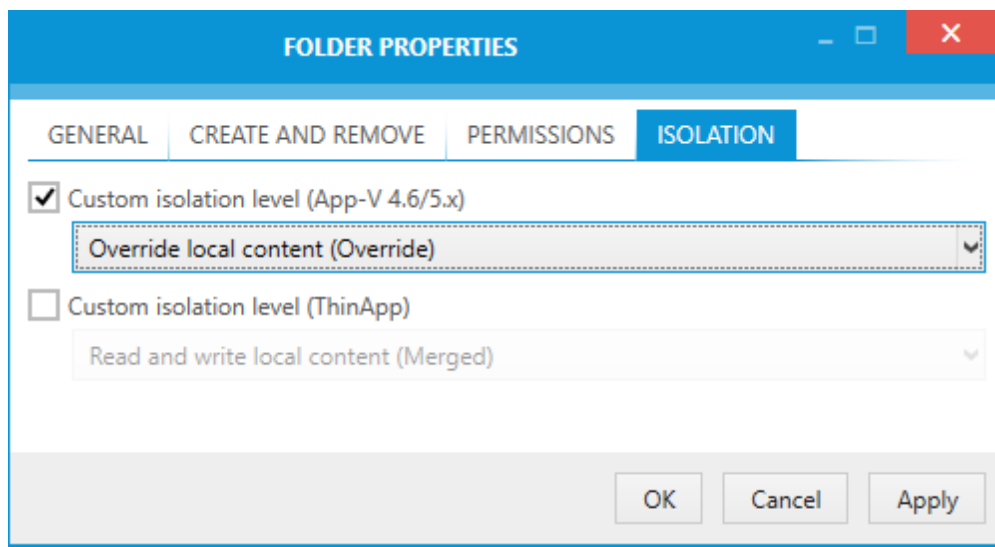
When no custom isolation is used, RayPack will try to use reasonable defaults based on type and location of a resource, to provide the best possible results that should work for most of the packages.

System Interaction

The System interaction view allows you to make changes to the INI files, TXT changes, File extensions, Context menu, Services, ODBC, and drivers settings.

Note:
If any of the items listed above is not available within your RayPack installation, check for the actual set of features covered by your license. To do so, visit the About section and expand the license details with the full listing of licensing options.

INI Files

The INI file format is an informal standard for configuration files as simple as text files with a basic structure composed of 'sections' and 'properties'. RayPack supports INI file management by *auto-validation and auto-correction*.

Tip:
Where possible, you should avoid using hardcoded values and paths, and use properties instead. This allows you to customize the contents of INI files during installation. To gain direct access to the keys and values extracted from the INI file sections into the Installer database, right click any added INI file and pick **Go to Row** from the context menu. The Advanced mode is displayed, with the contents of the `IniFile` table already loaded into the **TABLE** editor view.
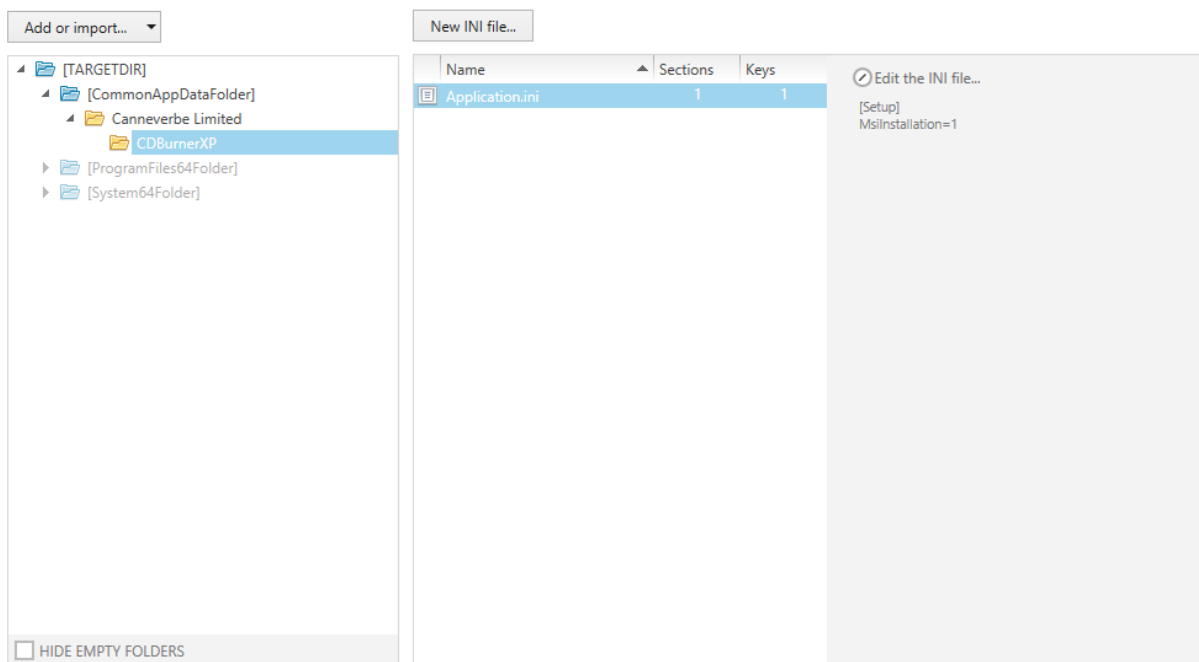
The INI Files View Structure

The INI files view is separated into three main areas:

- At the left hand side of the INI files view, there is a representation of the **directory tree** that has been established within the packaging project.

- The set of INI files stored inside a specific folder of the packaging project directory structure is loaded into the window at the center of the INI files view as soon as the folder itself is selected with a left click. The **list of INI files** shows the file name, as well as the number of sections and keys extracted from the file contents. Click on the tables column headers to sort the file list ascending or descending according to that specific property.
- Once an INI file is selected from the list of files within a specific directory, its contents are loaded into a **preview area** at the right-hand side of the INI files view.

The usual path of working with INI files is to select a specific directory first, create or select an INI file from that directory, and then manipulate the actual content of that file. Therefore, the orientation is designed to start at the left and go on to the right for further details.

Reorganize INI Files View Areas

Depending on the current activity it may be required to adjust the width of the three columns of the INI file view. To do so, point to the lines that mark the border of each display area, and press the left mouse key as soon as the mouse pointer switches to the dynamic width indicator symbol. Keep the mouse key pressed, and slide the window limit to the left or to the right. As soon as the mouse key is no longer pressed, the new area dimensions are kept for as long as the current instance of RayPack is active.

Standard INI File Management Procedures

Please refer to the following sections to read about object management functionality available within the INI files view of the Visual Designer mode:

- *Add an INI file*
- *Rename an INI file*
- *Remove an INI file*
- *Edit an INI file*

> **Note:**
> This view allows you to manipulate INI files that will be installed on the target system. In order to simplify the integration of INI files into the general folder structure of the target package, the view includes the folder management window as already known from the *package file browser* of the *files and folders* view. Please refer to the help topics about that view for details regarding standard folder management functionality, such as adding predefined folders, folder properties, and the like.

Add an INI File

Within the Visual Designer mode, adding INI files to a packaging project may be done by *manually creating new files* inside the existing structure, or by *importing existing files* from external resources, such as local volumes, network shares, or flash drives.

Create a New INI File

1. Each INI file resides inside a specific directory of a packaging projects folder structure. Therefore, when a new INI file has to be added, the first step is to **browse the directory tree in the left folder area of the INI file view for the desired target folder.**

   If the target directory does not exist yet, add new folders to the packaging project wide folder tree structure first, e. g. by right-clicking any existing folder and picking New Folder from the context menu.

2. Once the parent folder for the new INI file is selected by a left click, the file area for that specific folder is loaded into the center of the INI file view.

   Use the **New INI file...** button above that area to add a new INI file with a sample collection of data. As soon as the file is created, its properties dialog is displayed for direct content manipulation. Please refer to the *Edit an INI file* section to get details regarding INI file content manipulation.

3. New INI files are automatically named with a RayPack default INI file name, e. g. `New INI file.ini`. If an INI file with that name already exists inside the currently selected folder, the automatically generated file name is extended with an incrementing index value, such as `New INI file (2).ini`.

   It is highly recommended to rename those new files to resemble the actual set of values bundled inside. Please refer to the topic *Rename an INI file* to get details on INI file renaming rules.

Import an INI File

1. Within the packaging project directory tree on the left-hand side of the INI file view, **navigate to the target folder** for the import. **Select** it with a mouse click.

2. Reveal the set of object creation options for package folders with a click on the **Add or import** button. As an alternative, users may also right-click somewhere inside the left folder or right file display area, and reveal the object creation options available in the context menu.

3. From the displayed menu, select **Import file(s)...** to open a Windows system browser dialog.

4. Browse to the required folder within the **Select files to be imported** dialog.

5. **Select one or more INI files** from that folder and use the **Open** button to import them into the packages INI file stock.

   A copy of the selected files is imported into the project and displayed within the center of the INI files view. The windows system browser is closed automatically.

Rename an INI File

To change the name of an INI file within the PackDesigner Visual Designer mode:

1. Users have to browse the packaging projects directory tree, **select the parent folder**, and load the list of INI files inside that folder to the directory content display area. From the list of INI files:

   - **Left-click** a file name and hit **F2** on the keyboard
   - **Double-click** on a file name
   - **Right-click** a file name and select **Rename** from the context menu

2. Either way, the file name is set into direct inline edit mode. Simply type the new file name to replace the existing one.

3. To exit the direct inline edit mode without changing the prior INI file name, hit **Escape** on the keyboard.
   To save the new file name, hit **Enter**.

Since packaging projects are highly standardized data records, many objects have to follow specific naming conventions to be valid. Please make sure to follow the naming conventions for INI files when a file is renamed:

- INI file names must be unique within the same parent directory
- INI file names must not be empty
- INI file names may consist of alphanumerical strings
- INI file names are stored in a database column of the FileName type. Please refer to the *MSI standard documentation* to get additional information regarding the restrictions of short and long file name notations which are applicable to this data type.
- RayPack automatically adds new INI files with the file extension `.ini`. However, it is possible to apply any extension required, e. g. `.conf`, `.cfg`, or `.txt`.

Remove an INI File

To remove an INI file from a packaging project by using the PackDesigner Visual Designer mode interface functionality:

1. Users have to browse the packaging projects directory tree, **select the parent folder**, and load the list of INI files inside that folder to the directory content display area. From the list of INI files:

   - **Left-click** a file name and hit **Delete** on the keyboard
   - **Right-click** a file name and select **Delete** from the context menu

2. Either way, the file is immediately deleted from the packaging project.

> **Be aware:**
> As soon as an item is deleted from the INI files view, its resembling data row as well is deleted from the affected Installer database table `IniFile`.

Edit an INI File

To change the contents of an INI file within the PackDesigner Visual Designer mode:

1. Users have to browse the packaging projects directory tree, **select the parent folder**, and load the list of INI files inside that folder to the directory content display area. From the list of INI files:

    - **Left-click** a file to load its contents into the file content area on the right-hand side of the INI files view, and use the **Edit the INI file...** button above the actual file content listing
    - **Left-click** a file and hit **Alt + Enter** on the keyboard
    - **Right-click** a file and select **Properties** from the context menu

2. Either way, the INI file properties dialog is displayed, showing the listing of sections with their subordinated key / value pairs.

To change the INI file contents, users can either manually edit the current content, or replace it with new *content loaded from an external INI file* resource.

Note:

Inside RayPack packaging projects, INI file contents are not stored in a single file, but as several rows within an Installer database table. RayPack applies MSI standard rules on INI file contents, which may very likely not be meet by the original content structure of an imported INI file. Therefore, it is a symptom of *auto-correction and auto-validation*, that the displayed content of an INI file after its import into RayPack is different from the content of the original file. The same re-organisation is applied when changes to an INI file are saved, which may lead to differences in the displayed content structure between two editing sessions. However, these variations exist due to the appliance of MSI standard rule sets.

Tip:

If the structure of the INI file content is of vital importance for the target application, which may be the case for older applications, it is recommended to upload the original INI file resource as a standard file resource via files and folders, and simply manage the required changes and dynamics via the INI file control mechanisms.

Loading Content from an External INI File

1. Loading new content from an external source is initiated by using the **Load...** button from the lower left corner of the INI FILE PROPERTIES dialog.

2. A windows system browser is displayed, allowing users to browse for INI files.

3. As soon as the desired source file is opened, RayPack imports the content.

   Invalid lines, such as comments, are automatically removed whilst the import is executed. If the newly imported content is deemed to be invalid, a warning is shown below the content listing box as soon as the import is done.

Auto-validation and Auto-correction for INI File Contents

No matter how the content got into the INI FILE PROPERTIES content input field, users may try to save the newly defined content by either hitting the **OK** or the **Apply** button at the lower right corner of the INI FILE PROPERTIES dialog. Please note that using Apply attempts to save the changes and keep the current dialog open, whilst OK closes the dialog after successful saving.

On save, RayPack analyzes the content and marks lines that cannot be parsed as valid INI file content for packaging projects. Error messages regarding invalid content are displayed below the content editor area.

The auto-correction feature removes comments and not required empty lines from INI file content. Additionally, if a key is entered with an empty value, RayPack automatically translates the empty value into a placeholder ("[~]"), which is valid towards MSI standard requirements for

INI file contents. if any auto-correction has been applied to the given content, RayPack shows a notification message when the OK or Apply button is used. Therefore, the dialog for content manipulation is not closed automatically if the save functionality could not be completed without errors or auto-corrections.

As long as there are invalid items present in the content area, the changes to the INI file content are not saved. Users have to either solve the named issues, or cancel the content edition in general.

Please note that hints regarding applied auto-correction measures are displayed as info boxes with yellow background. Any changes to the INI file content have been saved and the input containing the INI file content is updated to match the current state of the INI file according to the resembling content of the Installer database. Error messages are displayed with a red background color. If an error message is displayed, the changes could not be saved, and the database content is not updated yet.

TXT Changes

This view allows you to manipulate text based files that will be installed on the target system. Using a simple method you can achieve the same flexibility with text based files as with ini files.

There are two basic change operation types for text based files: Find & Replace and Find & Append. Changes can be applied to any type of text based file, including TXT, SQL, HTML, and XML files. Patterns for search operations may be defined as plain strings or regular expressions, and applied during installation and / or de-installation sequences.

	Be aware:
☝	Since text manipulations are combinable with regular expressions, it is not possible to achieve a clean rollback once a task is executed. RayPack adds text manipulation tasks to the end of the execution sequences, which reduces the risk of erroneous changes. However, if an installation or uninstallation fails after text manipulations have been executed, there is no automatic rollback mechanism to restore the original state of the affected files.

Within this TXT changes view, a list of actual manipulation tasks is managed. RayPack stores them within the custom database table RPTextReplacements.
These tasks are executed at run time, controlled by predefined custom actions, which are automatically added to your packaging project with the first TXT change task. Look-up the **CUSTOM ACTION**S view of the Advanced mode and browse for the actions called `RPTextReplacements` and `RPTextReplacementsData` for details on settings and execution parameters. (It is also possible to navigate to the data row resemblance of a replacement task directly. To do so, right-click a task from the list displayed within the Visual Designer view TXT changes, and select the Go to row option from the context menu.)

	Be aware:
☝	All objects required for TXT changes are plainly visible within the packaging project resources. However, manipulating the custom actions required for the actual task execution is not recommended. Please keep the required custom actions as they are, and if not absolutely required, do not change their position within the run time

sequences either.

TXT Changes View Areas

The TXT changes view is divided into a list of already existing change tasks on the left and a details pane on the right, which shows the properties of the currently selected object from the task list.
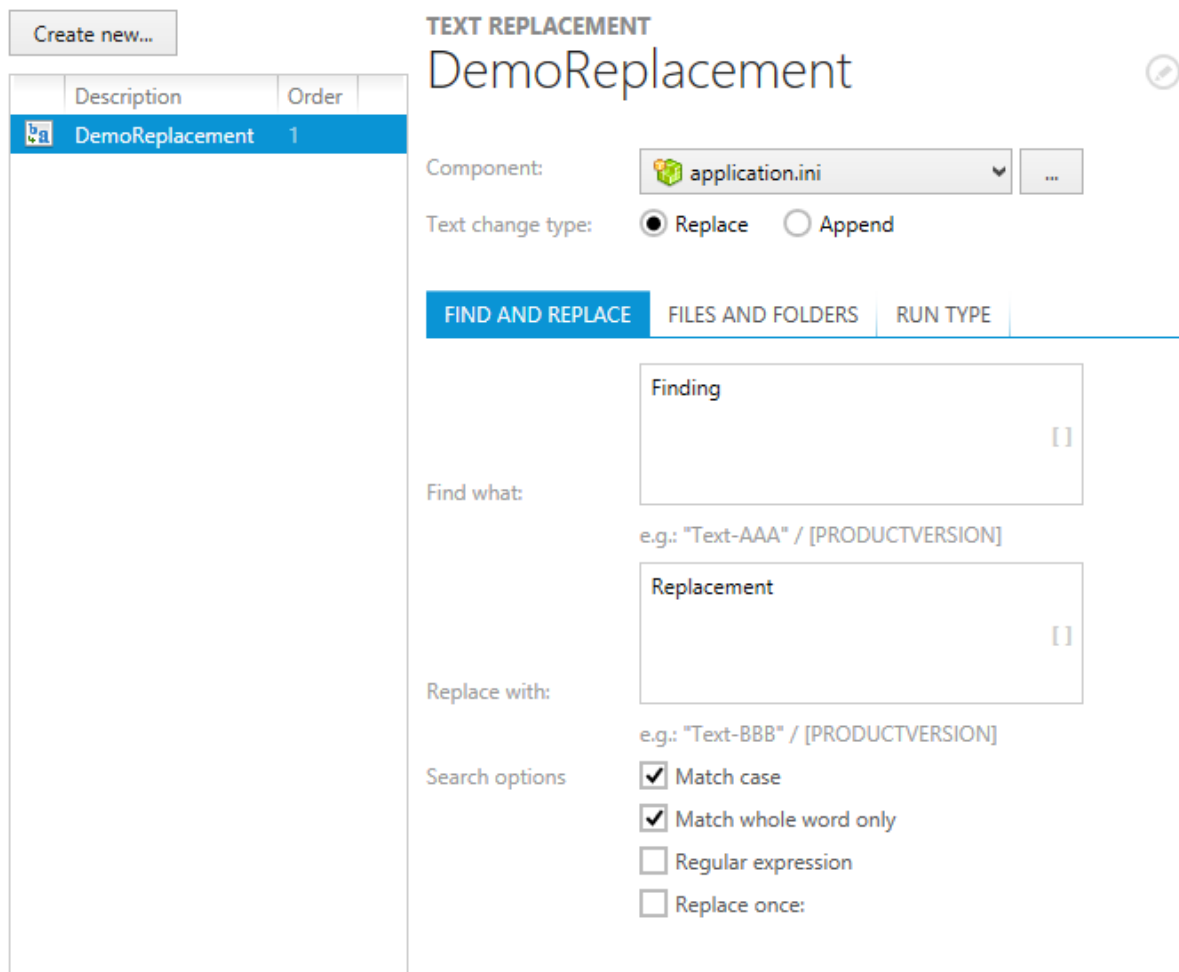
The list of existing change tasks is used to control the actual sequence of execution during run time. New tasks are automatically added to the last position of the ordered list. To change the sequence order, right-click the object that should be moved, and select one of the sequence order options from the context menu:

- Install first - moves the task to the first position of the sequence
- Install earlier - decrements the order value of the task by one
- Install later - increments the order value of the task by one

- Install last - moves the task to the last position of the sequence

The details pane of the TXT changes view is used to display and edit the task properties. Please refer to *Edit a TXT changes task* for a detailed description.

Standard TXT Changes Management Procedures

Please refer to the following sections to read about object management functionality available within the TXT changes view of the Visual Designer mode:

- *Add a TXT changes task*
- *Rename a TXT changes task*
- *Remove a TXT changes task*
- *Edit an TXT changes task*

Add a TXT Changes Task

In order to add a TXT changes task to a packaging project, users go to the TXT changes view of the Visual Designer mode.
With a click on the **Create new...** button, wizard for task creation is invoked.

Work your way through the steps of the wizard to define all required properties for the new task.

At any time, using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, allows to navigate within the already processed steps.
To exit the wizard without creating a new object, use the Cancel button, also located at the bottom of the wizard dialog.

Step 1: Description

The name of the task has to be given as an unique alphanumerical string. Special characters (e. g. blanks, equals (=), exclamation mark (!), plus (+), minus (-), or asterisk (*)) are not allowed.

Since the description is the property that is displayed in the task list of the TXT changes view, it is recommended to use a self-explanatory value. By doing so it becomes easier to quickly access a specific task from a large list.

As long as the input field is marked by red background, the value is not valid yet. Please use the mouse cursor to hover over the input field and reveal the message that indicates what needs to be done to achieve a valid task description.

Step 2: Component

Each change task has to be related to a specific component. To select the component from the existing set of the current packaging project, users expand the selector control and click on the desired component.

If a new component has to be created, users click on the **[...]** button on the right hand side of the component selector control. The *select component dialog* is displayed. Please refer to the resembling *common dialogs* topic for further details.

Step 3: Text Changes Type

Two basic operational TXT change types are available:

Find & Replace
Searches for a specific string or a string matching the specified pattern and replaces the original content.

Find & Append
Searches for a specific string or a string matching the specified pattern and appends additional text to it.
This option is mostly used when a regular expression is evaluated during the search algorithm, and the varying original text phrases that actually match the search pattern should not be changed to one common new phrase by the txt change operation.

To select one of the two options, click on the resembling tile. The currently active option is marked with a check mark icon in the upper right corner of the type tile.

Step 4: Find and Replace

According to the selected change type, the entered values in this view are interpreted to execute the text change:

Find what
The definition of the actual string or pattern to search for.

> **Note:**
> If the entered value has to be evaluated as a regular expression pattern, the search option **Regular expression** has to be activated.

Replace / Append with
The new content that will be used as replacement or accretion. The value given here must be the actual text, regular expressions are not evaluated.

> **Note:**
> As indicated by the square brackets ("[ ]") at the right hand side of the input fields for search and replace / append, it is possible to make use of syntax suggestions to define the search string. Simply enter an opening square bracket ("[") to display the list of available suggestions for property and path values. Select the desired value from the list to add it to the already entered search string. The actual value at run time will be used for the task execution.

Match case
If this checkbox is activated, the case of the letters used within the entered search string has to

match the case of the matches in the searched file contents. This option is only valid for exact searches, not for regular expression matchings.

Match whole word only
Activating this checkbox disables the RayPack standard behavior to search for any occurrence of the search string. If the checkbox is activated, the string will only be matched if it is found as a whole word, meaning that there has to be a blank before the string.

Regular expression
This option must be activated if the search term is assigned for regular expression matching. Even if the search term is a regular expression, it will not be evaluated by the resolver engine as long as the search option "Regular expression" is inactive.

> **WARNING**
> Regular expressions are very powerful tools, which can be used to execute highly complex manipulation requirements. However, any potent tool can cause serious harm if used incorrectly. Therefore it is highly recommended to apply regular expressions on clearly defined target file groups. Responsible packaging engineers should make sure that vital resources of target machines operating systems are not damaged by their packages.

Replace once
Text replacements are usually executed on all matching strings within the given path and file type specification (see next step: *Files and Folders*). If this checkbox is activated, only the first occurrence is replaced.

Step 5: Files and Folders

In this step the files that have to be considered by the change task have to be defined:

Target folder
The folder on the target machine that will be searched for matching files.

As indicated by the square brackets ("[]") at the right hand side of the input field, it is possible (and recommended) to make use of syntax suggestions to define the target folder. Simply enter an opening square bracket ("[") to display the list of available suggestions for property and path values. Select the desired value from the list to add it to the already entered text in the input field. The actual property or path value at run time will be used for the task execution.

> **Be aware:**
> The definition of the target folder can cover whole system drives, e. g. the `c:` drive of the target machine. If the Include subfolders option is activated, the change task will search every directory on the defined drive for files with matching content. Depending on the system performance, and the amount of files on that drive such global target definitions may cause severe issues during the software package installation and / or uninstallation sequences. Therefore, it is highly recommended to reduce the target folder definition to the smallest required area.

The target folder property may not be left empty.

Include subfolders
The standard behavior for change tasks is to check all files within the defined target folder for the search term - files that are stored within subfolders are not considered. Activating this checkbox sets up a search task that considers all files below the target folder, no matter how deep they are nested.
Again, please be aware that including subfolders may considerably reduce the target package's sequence execution performance.

Include files
The files within the defined target folder that have to be considered by the change task. The file list is a comma separated list of either actual file names (e. g. "`file1.txt,file2.htm`"), or of file extensions (e. g. "`*.txt,*.htm`"). Combinations of both value types are valid as well.

The wild card operator asterisk ("`*`") can be used to match any custom file group beyond file type extensions, e. g. use definitions like `AnyFile*.ini` to match a set of files with an index counter within the file name.

The include files property may not be left empty.

Exclude files
Every rule is only as good as its best exception. Therefore it is possible to reduce the list of included files, e. g. to protect system or license files from undesired manipulation. All files given in this comma separated list will not be affected by the change task, even if target folder, include files restriction and search term definition would match.

Whilst the include files list is usually a combination of file types defined by their file name extension string (e. g. "`*.xml`"), the exclude files list is usually a list of specific files. Again, the wild card operator asterisk ("`*`") may be applied to widen the definition.

Step 6: Run type

Text manipulations may be executed during package installation and / or uninstallation run time. Activate the desired run type by clicking on the resembling option tile. The currently active option is marked by a check mark in the center of the tile. The default run type is INSTALL, which will execute the change task only during package installation sequences.

Step 7: Summary

Use the summary page to check the correctness of the task properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the object
- If changes are due, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the

summary page again.

Step 8: Finished

Once the new TXT changes task has been created, the wizard can be closed by using the **Finish** button at its lower right corner. The TXT changes view is updated, and the list of existing tasks contains the newly created object at the lowest position of the run order.

Rename a TXT Changes Task

1. To rename a TXT changes task, users load the list of existing tasks by calling the TXT changes view within the Visual Designer mode of PackDesigner.

2. Clicking on the list item of the object that has to be renamed displays its details in the editor panel on the right hand side of the RayPack application screen.

3. The task description is displayed above the form elements of the edit area.

4. The description becomes editable with a click on either the description text, or the edit icon on the right hand side. A *direct value editor dialog* is displayed, allowing to enter the new description content.

   The name of the task has to be given as an unique alphanumerical string. Special characters (e. g. blanks, equals (=), exclamation mark (!), plus (+), minus (-), or asterisk (*)) are not allowed.

5. By hitting **Enter** or clicking on the **save** icon within the direct value editor interface the new description is saved.

Remove a TXT Changes Task

To remove a TXT changes task from a packaging project by using the PackDesigner Visual Designer mode interface functionality:

1. Users have to call the TXT changes view. From the list of tasks on the left hand side:

   - **Left-click** a task item and hit **Delete** on the keyboard
   - **Right-click** a task item and select **Remove** from the **context menu**

2. Either way, the task is immediately deleted from the packaging project.

> **Be aware:**
> As soon as an item is deleted from the TXT changes view, its resembling data row as well is deleted from the affected Installer database table `RPTextReplacements`.

Edit a TXT Changes Task

1. To edit a TXT changes task, users load the list of existing tasks by calling the TXT changes view within the Visual Designer mode of PackDesigner.

2. Clicking on the list item of the object that has to be modified displays its details in the editor panel on the right hand side of the RayPack application screen.

The basic task properties are always available for edition, whilst additional options are grouped in separate tabs and may therefore not be visible when the edit area is loaded initially.

Basic Properties

Description
Changing the description of a TXT changes task is described in the topic *Rename a TXT changes task*.

Component
Each change task has to be related to a specific component. To select a different component from the existing set of the current packaging project, users expand the selector control and click on the desired component from the displayed list.

If a new component has to be created, users click on the **[...]** button on the right hand side of the component selector control. The *select component dialog* is displayed. Please refer to the resembling *common dialogs* topic for further details.

Text change type
Switching the radio button options Replace and Append automatically switches the action RayPack applies on the change task.

Tab: FIND & REPLACE / APPEND

The manipulation options available within this tab are described in the help topic *Add a TXT changes task*.

Tab: FILES AND FOLDERS

The manipulation options available within this tab are described in the help topic *Add a TXT changes task*.

Tab: RUN TYPE

The manipulation options available within this tab are described in the help topic *Add a TXT changes task*.

As usual within inline editing, all valid changes to the task options are saved immediately.

File Extensions

This view allows you to manipulate file extension settings that will be installed on the target system. Each file extension object created here is stored within the `Extensions` table of the Installer database.

At run time, the installer generates a matching set of registry keys and values. Defining a file extension object generates instructions for the target system, including information about the specific package feature that will install the actual extension server, and the specific component that will install the extension itself.

File Extension View Areas

The file extensions view is divided into a list of already existing extension objects on the left and a details pane on the right, which shows the properties of the currently selected item from the object list.

The details pane of the view is used to display and edit the file extension properties. Please refer to *Edit a file extension* for a detailed description.
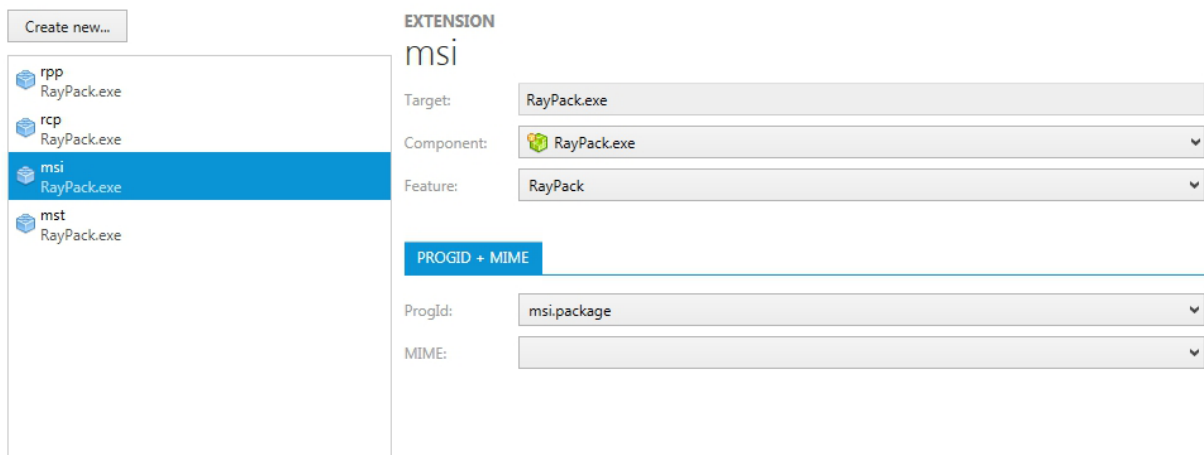
Standard file extension management procedures

Common management procedures for file extensions are described within the following topics:

- *Add a file extension*
- *Remove a file extension*
- *Edit a file extension*

File extension definitions are usually combined with context menu items, since each context menu data object refers to a specified file extension from the same packaging project.

Add a File Extension

In order to add a file extension to a packaging project, users go to the file extension view of the Visual Designer mode.

With a click on the **Create new...** button, the wizard for extension creation is invoked.

Work your way through the steps of the wizard to define all required properties for the new object.

At any time, using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, allows to navigate within the already processed steps.
To exit the wizard without creating a new object, use the Cancel button, also located at the bottom of the wizard dialog.

Step 1: Extension

Creating a new file extension object can be assigned by manually writing the extension string, or by selecting an already existing extension.

The default option is to create a new extension. To do so, users have to activate the upper radio button option in this wizard step, and enter the file extension string into the now editable input field below.

A file extension is basically a string of 1 to 255 signs length. Internally, it is stored in a column of type text, which allows any kind of string. Logically, file extensions have to be entered as alphanumerical string, without leading period and without special characters.

Since some file systems limit the length of valid file extensions to three (FAT) or six (VM/CMS) characters, it is recommended to make sure that the extension matches the target file system restrictions.

To create a new file extension based on an extension object that already exists within the current packaging project, users activate the second option of the radio selection "Use an extension that is already present in this Windows Installer database", and select the base extension from the now expandable list of existing file extension objects.

Step 2: Target Selection

In this wizard step the target executable for the file extension has to be selected. The directory tree of the currently opened project is displayed. Navigate through the tree structure to display the desired executable in the file list area at the right-hand side of this dialog. From the list of files, select the required executable with a left-click.
The Next button becomes available as soon as a valid file resource for file extension targets is selected.

Step 3: ProgId

Relating a file extension to a ProgId is optional. Therefore, the radio button is set to the default value "Don't setup ProgId". Set the radio selector to "Create new ProgId" or to "Use existing ProgId" to relate a ProgId to the file extension.

A programmatic identifier (ProgId) is a registry entry that can be associated with a CLSID. The format of a ProgID is `<Vendor>.<Component>.<Version>`, separated by periods and with no spaces. Although the internal storage format for ProgId names is flat text, there are some logical restrictions to the creation of valid ProgIds, since they must:

- Have no more than 39 characters.

- Contain no punctuation (including underscores) except one or more periods.
- Not start with a digit.
- Be different from the class name of any OLE 1 application, including the OLE 1 version of the same application, if there is one.

If an existing ProgId should be related to the file extension, users have to select the specific ProgId from the list of available ProgId's, which becomes available as soon as the option **Use existing ProgId** is enabled.

If a new ProgId has to be created, users may decide to either use the default icon (which is the standard setting defined by the active checkbox below the Create a new ProgId radio selector option), or define a specific icon. To define an individual icon, the checkbox "Use default icon" has to be de-selected. The icon selector interface is displayed as soon as the checkbox is un-selected.
An individual icon relation may be established with an icon resource that has already been added to the project, or with a new icon resource.

To pick an existing icon, users have to click on the button with the downwards pointing arrow icon, and select the desired one from the list of already available icon objects within the packaging project. The active index of the icon may be adjusted after it has been selected from the list, by simply using the left- and right-wards pointing arrows below the currently selected icon preview.

To add a new icon resource, users have to click on the "load from disk..." button, and use the windows file system browser to navigate to the desired icon resource file. Select the desired icon file, and click Open to add it to the project in general, and the new ProgId in particular.

Be aware:
Once a ProgId is related to the file extension, this relation cannot be removed via the Visual Designer interface. To remove a ProgId relation from a file extension object, users have to directly edit the resembling Installer database table row via the **TABLES** view of the Advanced mode.

Step 4: MIME

Building a relation between a file extension and a MIME object is optional. Therefore, the radio button is set to the default value **Don't setup MIME**. Set the radio selector to **Create new MIME** or to **Use existing MIME** to relate a MIME object to the file extension.

If there are no MIME objects stored within the current packaging project, the option list of the selector control for **Use existing MIME** is empty. If the MIME object has not been prepared already, a new MIME object may be defined when the **Create new MIME** option is enabled. Please adjust the MIME object properties by directly accessing the MIME database table.

Be aware:
Once a MIME is related to the file extension, this relation cannot be removed via the Visual Designer interface. To remove a MIME relation from a file extension object, users

---

have to directly edit the resembling Installer database table row via the **TABLES** view of the Advanced mode.

Step 5: Summary

Use the summary page to check the correctness of the object properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the item.
- If changes are due, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the summary page again.

Step 6: Finished

Once the new file extension has been created, the wizard can be closed by using the **Finish** button at its lower right corner. The file extension view is updated, and the list of existing objects contains the newly created extension at the last list position.

Remove a File Extension

To remove a file extension from a packaging project by using the PackDesigner Visual Designer mode interface functionality:

1. Users have to call the file extension view. From the list of extensions on the left hand side:

    - **Left-click** an extension and hit **Delete** on the keyboard
    - **Right-click** an extension and select **Remove** from the **context menu**

2. Either way, the extension object is immediately deleted from the packaging project.

> **Be aware:**
> As soon as an item is deleted from the file extension view, its resembling data row as well is deleted from the affected Installer database table `Extension`.
>
> Once a ProgId or MIME object is related to the file extension, this relation is not automatically removed along when the file extension itself is removed. To remove a ProgId or MIME relation from a project, users have to directly edit the resembling Installer database table rows via the **TABLES** view of the Advanced mode.

Edit a File Extension

1. To edit a file extension object, users load the list of existing extensions by calling the file extension view within the Visual Designer mode of PackDesigner.

2. Clicking on the list item of the object that has to be modified displays its details in the editor

panel on the right hand side of the RayPack application screen.

The following properties of the extension object may be manipulated:

Basic Properties

Extension

A file extension is basically a string of 1 to 255 signs length. Internally, it is stored in a column of type text, which allows any kind of string. Logically, file extensions have to be entered as alphanumerical string, without leading period or special characters.

Since some file systems limit the length of valid file extensions to three (FAT) or six (VM/CMS) characters, it is recommended to make sure that the extension matches the target file system restrictions.

The combination of values for extension and component must be unique within the current packaging project. Therefore, if a new extension value is entered, which is already used for another extension related to the same component, the extension input field is marked with red background color, and the new value cannot be saved. Please hover over the input field with the mouse pointer to reveal a tool tip with additional information on how to solve the issue.

Component

Select one of the components from the offered list.

Again, the combination of extension string and component must be unique within a packaging project, therefore trying to change the component property is only successful when the new value pair is unique.

Feature

When a file extension object is created, RayPack automatically sets the value for the Feature property to the first found Feature object that is assigned to the component defined for the file extension object.

To change the current setting, users pick one of the offered Feature objects from the selector control menu, or use the create new button (*) at the right-hand side of the selector control to create a new feature. Please be aware that the combination of feature and component selection has to be valid in order to prevent later issues during package run-time.

Tab PROGID + MIME

ProgId

To change the current setting, users pick one of the offered ProgIds from the selector control

menu.

If the desired ProgId does not exist yet, a new ProgId may be created by clicking on the create new button (*) at the right-hand side of the selector control. Clicking the button opens the dialog for entering new ProgId data rows into the Installer database.

> **Be aware:**
> Once a ProgId is related to the file extension, this relation cannot be removed via the Visual Designer interface. To remove a ProgId relation from a file extension object, users have to directly edit the resembling Installer database table row via the **TABLES** view of the Advanced mode.

MIME

To change the current setting, users pick one of the offered MIME objects from the selector control menu.

If the desired MIME does not exist yet, a new MIME may be created by clicking on the create new button (*) at the right-hand side of the selector control. Clicking the button opens the dialog for entering new MIME data rows into the Installer database.

> **Be aware:**
> Once a MIME is related to the file extension, this relation cannot be removed via the Visual Designer interface. To remove a MIME relation from a file extension object, users have to directly edit the resembling Installer database table row via the **TABLES** view of the Advanced mode.
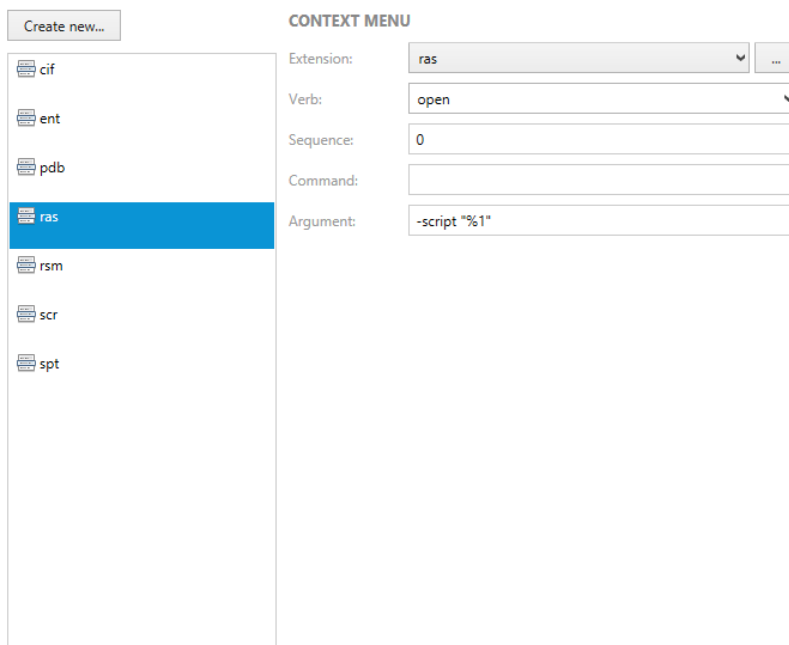
Context Menu

This view allows you to manipulate shell extensions (context menu items when right-clicking objects) that will be installed on the target system. In combination with a specific file extension, a context menu item allows to add predefined options to the context menu for action-options regarding specific file types on the target machine.

Internally, context menu items are stored within the `Verb` table of the Installer database. The combination of file extension and actual verb value has to be unique within a packaging project, to make sure that each activity option offered to the user via the context menu is clear and unambiguous. This leads to the condition, that a context menu item cannot exist without a relation to a file extension item created within the very same packaging project. Therefore, before context menu options are created, it is recommended to manage file extension handling objects first. To do so, users call the file extension view of the Visual Designer mode of PackDesigner.

Context menu view areas

The context menu view is divided into a list of already existing data objects on the left and a details pane on the right, which shows the properties of the currently selected object from the context menu item list.

The details pane of the view is used to display and edit context menu item properties. Please refer to *Edit a context menu* for a detailed description.

Below the input controls for object edition, the Action section is displayed, allowing users to

- call the File extension view of Visual Designer, loaded with the detail information of the file extension that is related to the current context menu object
- call the Advanced mode and jump directly to the Installer database row within the `Verb` table that resembles the current context menu item

Standard context menu management procedures

Common management procedures for context menu objects are described within the following topics:

- *Add a context menu*
- *Remove a context menu*
- *Edit a context menu*

Add a Context Menu

In order to add a context menu to a packaging project, users go to the context menu view of the Visual Designer mode.
With a click on the **Create new...** button, and the selection of the **Create context menu entry** option, the wizard for object creation is invoked.

Work your way through the steps of the wizard to define all required properties for the new object.

At any time, using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, allows to navigate within the already processed steps.
To exit the wizard without creating a new object, use the Cancel button, also located at the bottom of the wizard dialog.

Step 1: Select Extension

One of the mandatory properties of a context menu item is the file extension it has to be applied to. Therefore, this first wizard step cannot be skipped, or finished successfully without a valid file extension object in place. The combination of file extension and verb has to be unique within a packaging project, therefore it is possible to create several context menu items for the same file extension object, but impossible to create a new context menu item with the same verb and file extension values.

If there are no file extension objects available yet, this step shows a link to the wizard for file extension item creation. Click on **Add new extension** to call the wizard and prepare a file extension first. As soon as the file extension wizard is finished, the current view of the context menu creation wizard is updated. The newly created file extension object is already selected as extension for the context menu.

If more than one file extension is stored within the current packaging project, each file extension is represented by a separate tile. Users have to click on the tile of the desired file extension to mark it, and relate it to the new context menu object. Please note that the currently selected tile has a different background color and has a check mark icon in its upper right corner.

Step 2: Enter Verb

In this step users can choose whether to use a custom verb by manual definition, or to select a predefined one from the standard repository. The manual option is active by default. To activate the predefined option, click on the radio button selector in front of the predefined control label. As soon as one option is active, the other option becomes inactive.

Each context menu item needs to have a verb, therefore it is not possible to proceed without any selection or manual verb definition. Please note that the combination of file extension and verb must be unique. Therefore it is not possible to proceed until a unique combination is established. Hover over the input controls to reveal the tool tip with hints on how to solve the issue that prevents the wizard step from being valid.

Predefined verbs are localized automatically, whilst manual (also called custom) verbs cannot be localized on the fly.

Internally, verbs are stored as text values, and therefore may include alphanumerical signs and special characters. However, from a logical system perspective, verbs are commands which are interpreted by the shell extension, and thus have to follow some basic format guidelines:

- Custom verbs have to be interpretable according to the target application
- Verbs may not include special characters or blanks

Step 3: Context Menu Configuration

This step is designed to define the command that is displayed as context menu item on the target machine, as well as the sequence index of the current object.

Command
The command term is a localizable string, which can include properties and other variable contents of the packaging project. As indicated by the square brackets ("`[]`") at the right hand side of the input control, syntax suggestion is activated for the command string. Simply enter an opening square bracket ("`[`") to invoke the suggestion feature and see possible options.

Sequence
The sequence value is an integer that is used to order the context menu items per file extension. The context menu items are ordered ascending towards their sequence value. The one with the lowest value is automatically defined as default item.

Both, command and sequence can be left empty. The interpretation results will then depend on the shell extension on the target machine.

Step 4: Enter Argument

The command sent to the target machine may include arguments. The most commonly used is "`%1`", which represents the file name. An additional, though used less often, is "`%2`", which represents the name of the default printer specified for the target system.

Additional custom parameters may be added, but have to be handled with care: It is possible to use properties (applying the typical notation with square brackets (e. g. "`[MyProperty]`"), but the resolution of actual run time values can only be executed successfully when the installation sequence has already processed the property translation at the very moment when the component which contains the context menu item is installed.

As indicated by the square brackets ("`[]`") displayed on the right hand side of the input control, the syntax suggestion feature is enabled for the command input.

Step 5: Summary

Use the summary page to check the correctness of the object properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the item.
- If changes are due, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the summary page again.

Step 6: Finished

Once the new context menu has been created, the wizard can be closed by using the **Finish** button at its lower right corner. The context menu view is updated, and the list of existing objects contains the newly created item visible and editable.

Remove a Context Menu

To remove a context menu item from a packaging project by using the PackDesigner Visual Designer mode interface functionality:

1. Users have to call the context menu view. From the list of items on the left hand side:

   - **Left-click** an item and hit **Delete** on the keyboard
   - **Right-click** an item and select **Remove** from the **context menu**

2. Either way, the context menu object is immediately deleted from the packaging project.

Be aware:
As soon as an item is deleted from the context menu view, its resembling data row as well is deleted from the affected Installer database table `Verb`.

Edit a Context Menu

1. To edit a context menu object, users load the list of existing items by calling the context menu view within the Visual Designer mode of PackDesigner.

2. Clicking on the list item of the object that has to be modified displays its details in the editor panel on the right hand side of the RayPack application screen.

The following properties of the context menu object may be manipulated:

Extension

The selector control offers the list of available file extension objects within the current

packaging project. If a different extension is selected, and the new combination of extension and verb is not unique, the change is invalid and will therefore not be saved.

Hover over the selector control, which is marked with red background color in case of an invalid selection, and read the error message to solve incorrect extension selection states.

If the required file extension has not been added to the packaging project yet, users may invoke the dialog for the creation of new file extension objects by clicking on the **[...]** button at the right hand side of the file extension selector control. As soon as the new extension is created and the dialog window closed, the new item can be selected from the file extension selector control.

Verb

Users may either use the selector to choose one of the existing verbs, or simply type in a custom verb value. Please make sure to follow the restrictions for verb definitions.

As soon as a different verb value is selected or entered, the new combination of extension and verb checked for uniqueness within the packaging project. The change is invalid if the combination is not unique, and will therefore not be saved.

Hover over the selector control, which is marked with red background color in case of an invalid value, and read the error message to solve incorrect verb values.

Sequence

Enter an integer value for the context menu item order, or leave the input control empty.

Command

The command value is the localizable label of the context menu item which is displayed on the target machine when a user right-clicks a file with the defined file extension. If it is left empty, the target system defines a default label according to the verb value defined earlier.

Argument

The argument value may be left empty, contain parameters "`%1`" or "`%2`" for file or printer name, or make use of properties defined for the same component the file extension object related to the context menu item is installed by. Please make sure to use properties by applying the standard notation with square brackets (e. g. "`[MyProperty]`").
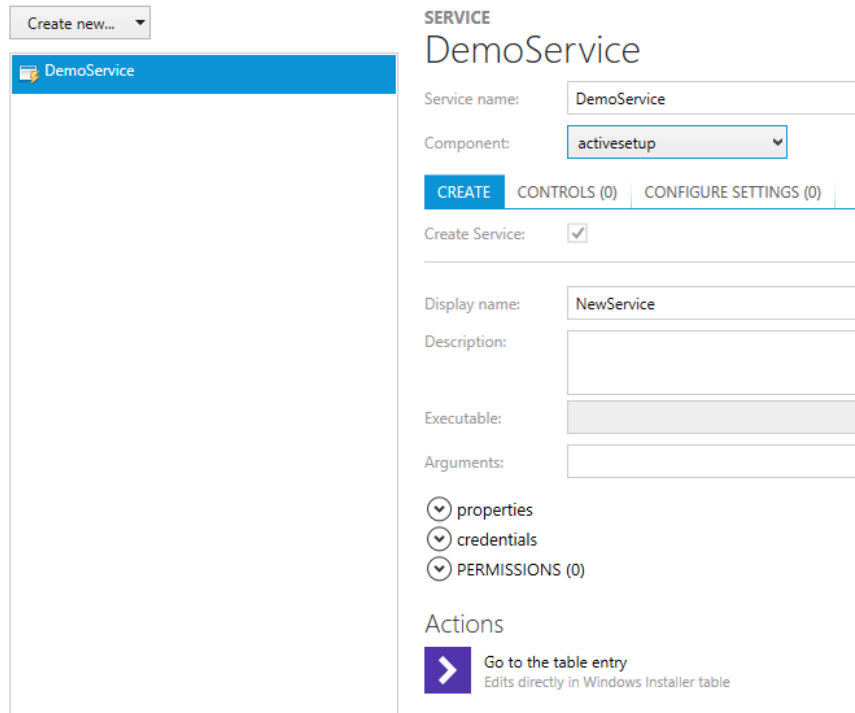
Services

This view allows you to manipulate services that will be installed on the target system. It also allows you to control existing services on the target system during the installation and de-installation.

Services View Areas

The services view is divided into a list of already existing data objects on the left and a details pane on the right, which shows the properties of the currently selected object from the services

item list.

The details pane of the view is used to display and edit service item properties. Please refer to *Edit a service* for a detailed description.

Standard Services Management Procedures

Common management procedures for service objects are described within the following topics:

- *Add a service creator*
- *Add a service controller*
- *Remove a service*
- *Edit a service*

Add a Service Creator

> **Note:**
> Service manipulation objects have to be related to components. Please make sure to provide a component for your packaging project before the wizard for the creation of new server mnipulation objects is started.

Service creator items are a set of service manipulation rules, that lead to the creation of a new service on the target machine at package installation. If an already existing service on the target machine has to be manipulated, users should create a new service controller item instead.

To add a new service creator, users have to:

1. Call the **services view** within the **Visual Designer** mode of PackDesigner.

2. Click on the **Create new...** button in the upper left corner of the view.

3. Select **Create service** from the options menu.

A new service creator object is automatically created and added to the list of service manipulation items in the left area of the services view. New service manipulation objects are named `NewService` by default. If an item with that name already exists, RayPack adds an incrementing index value to the default name (e. g. `NewService_1`).

The newly created service creator item is added to the `ServiceInstall` Installer database table. It is already equipped with some basic settings, such as:

- Service name - the default name mentioned above.
- Component relation - RayPack automatically adds the service creator object to one of the already existing components. The service executable is the components key path file.
- Create service - this option is predefined to be active.
- Display name - the same as the service name.
- Start-up type - the new service will start up automatically.
- Error control - the new service will log errors and continue on error.
- Service type - newly created services are set to run their own process in 32bit mode.

Especially the service name and component relation have to be adjusted in order to create the service according to the specific package needs.
Additional information, such as service control commands, permissions and required credentials, have to be added to the default object properties manually.
To adjust these settings, users have to *edit the service manipulation item*. Please refer to the specific help topic for details on how to do so.

Add a Service Controller

> **Note:**
> Service manipulation objects have to be related to components. Please make sure to provide a component for your packaging project before the wizard for the creation of new server manipulation objects is started.

Service controller items are a set of service manipulation rules, that lead to changes in the behavior or activity of an already existing service on the target machine at package installation. If a new service has to be created on the target machine, users should *add a new service creator* item instead.

To add a new service controller, users have to:

1. Call the **services view** within the **Visual Designer** mode of PackDesigner.

2. Click on the **Create new...** button in the upper left corner of the view.

3. Select **Control service** from the options menu.

A new service controller object is automatically created and added to the list of service manipulation items in the left area of the services view. New service manipulation objects are named `NewService` by default. If an item with that name already exists, RayPack adds an incrementing index value to the default name (e. g. `NewService_1`).

The newly created service controller item is added to the ServiceControl Installer database table. It is already equipped with some basic settings, such as:

- Service name - the default name mentioned above.
- Component relation - RayPack automatically adds the service control object to one of the already existing components.
- Create service - this option is predefined to be inactive.
- Display name - the same as the service name.
- Control - RayPack adds a start service control by default, which is executed during the installation sequence execution.

Especially the service name, component relation and control options have to be adjusted in order to control the service according to the specific package needs.
Additional information, such as further service control commands, have to be added to the default object settings manually.
To adjust these settings, users have to *edit the service manipulation item*. Please refer to the specific help topic for details on how to do so.

Remove a Service

To remove a service manipulation item from a packaging project by using the PackDesigner Visual Designer mode interface functionality:

1. Users have to call the services view. From the list of items on the left hand side**, right-click** an item and select **Remove** from the **context menu**

2. A dialog is displayed, requesting the user to confirm the object deletion.

3. The service manipulation object is immediately deleted from the packaging project when the user clicks on REMOVE to confirm.

Be aware:
As soon as an item is deleted from the services view, its resembling data row as well is deleted from the affected Installer database table `ServiceInstall` or `ServiceControl`.
Related objects, such as additional control commands for a service control item, are deleted along with the basic service manipulation object.

Edit a Service

Service manipulation objects come along with a quite substantial set of setting options. In order to provide a clean and easy to master interface for these objects, the edit service area is separated into four activity blocks:

- Basic properties
- Create
- Controls
- Control Settings

Whilst basic properties are always available for editing, the other settings are grouped in tabs users have to call manually.

Basic Properties

Service Name

This property is the name of the service on the target machine.
- If a service is created, the name will be set for internal service recognition and handling.
- If a service is controlled, the name used in this value has to match the name of the controlled service on the target device. The matching is executed case insensitive.

Service names may consist of max. 256 alphanumerical characters. It is not allowed to use slash ("/") and back-slash ("\") within service names.

Component

Each service manipulation item needs a component relation.

- For service creations, the key path of the related component must be the executable for the new service. Additionally, the service is only created if the component is actually installed during Installer sequences.
- For service controls, the component decides whether the service control task is executed during the Installer sequences.

Please keep in mind that the default component RayPack selects when the service manipulation object is added to the packaging project will very likely not be the one desired for the target package. Make sure to adjust this setting to the correct component!

Tab: Create

Note:

The properties users can manipulate within this tab are not of relevance for service control items.

However, to turn a **controller item into a service manipulation object** that actually creates a service, users may **deactivate** the **Create Service** checkbox available within this tab.

Users may likewise **turn a service control item into a service creator object** by **activating** the **Create Service** checkbox available within this tab.

If the manipulation type is switched, users have to check the service manipulation object regarding the existence and validity of all required settings and references. Especially for control items that have been turned into creator items, the key path file status of the related component must be verified towards the correct service executable.

Create Service

This option is active when the service manipulation item is designed to create a new service on the target machine.

It is not possible to deactivate this checkbox until a service manipulation item has been extended with a set of control commands (see below)

Display name

This is the localizable service name as it will be displayed for users on the target system.

Description

The description will be visible in the service management interface of the operating system on the target machine.

Executable

This read only field displays the executable file derived from the key path value of the related component. If this field is empty, the component does not contain a suiting key path file yet.

Arguments

Add parameters to the command line used to run the service. If more than one parameter is added, they have to be separated by null ("`[~]`").

Toggle Group: Properties

Start-up type

The option selected here is translated into a set of bit flags for service start-up control.

- Automatic - the service starts automatically with the operating system

- Manual - the service starts if and when the service control manager calls the StartService function.
- Disabled - the service cannot be started any more.

Error control

The option selected here is translated into a set of bit flags which control the reaction triggered on service start-up errors on the target machine.

- Log and continue - logs the error and continues with the start-up operation.
- Log and show message - logs the error, displays a message box and continues the start-up operation.
- Log and restart service - logs the error if it is possible and the system is restarted with the last configuration known to be good. If the last-known-good configuration is being started, the start-up operation fails.

If the checkbox "Abort install on error" is active, the installation of the target package itself will be aborted if the service could not be created successfully.

Service type

Services may run as a separate process, or as a service program with shared code components in the same process. Please refer to the MSDN documentation about service programs for details on how to pick the right option.

Interaction

This checkbox is used to mark services that interact with the desktop. If interaction is allowed is controlled by the security mechanisms of the target devices operating system.
Please note that the Login property within the credentials section must be set to "LocalSystem" or null ("[~]") when using this option.

Load order group

A Service Group is a collection of similar services that are loaded together at start-up. Entering a value into the Load order group input field indicates that the new service is added to the specified group.

Dependencies

Enter the names of service groups or individual services that must be started on the target device before the current service item may be started. If several dependencies have to be entered, separation by Null ("[~]") is due. Please refer to the MSDN documentation for details on how to handle complex dependency notations.

Toggle Group: Credentials

Login

The user profile used to run the service. Depending on the service type and interactivity settings, the login value must either be noted including the domain name (DomainName\Username or .\UserName for built-in domain users), or has to be represented by "LocalSystem" or null ("[~]").

Password

The password that matches the login user name defined above.

Toggle Group: Permissions

> **Note:**
> Permission handling is not available for service manipulation items within packaging projects that use a Windows Installer schema below 500. Please set the Installer schema via the Summary information view available within the Setup organisation section of the Visual Designer mode.

Within the permissions section, users are able to add permission definitions which are added to the `MsiLockPermissionsEx` Installer database table. Click on the Add button to create a new combination of Security Descriptor Definition and condition string. RayPack allows to add several permission handling items to a service.
Removing a permission restriction is executed by clicking on the delete button, displayed at the upper right corner of the permission definition set area.

SDDL
The string defined in the Security Descriptor Definition Language defines the actual permission set for the given service. When a permission set is added, RayPack prepares a default string that has to be adjusted to the actual needs of the packaging project. Even though it is possible to directly change the SDDL string within the displayed input field, it is recommended to use the extended editor dialog. Use the "edit sddl text" button below the SDDL input field to open that editor dialog.

Condition
If the permission is bound to specific conditions, these may be defined with support or the RayPack condition builder. Use the "edit in condition builder" button below the condition input field to open that editor.

Tab: Controls

Both service manipulation item types allow the definition of control commands to be applied to the service during installation and uninstallation run-time. When a new service control item is added to a packaging project, RayPack automatically adds an item to the control tab, assigning the service to start during installation and wait for the SCM report before the installation carries on. The identifier of the control command is defined automatically as a unique combination of several values (such as the name of the parent service manipulation object, an incrementing index value and the keyword "Control").

A service control item needs to have at least one control command, whilst service installer items may be executed without any control command at all.

To add a control command, users click on the Add button which is displayed above the list of already existing control commands. A new command is added with the very same default settings applied as described above.

To delete a control command, users click on the Delete button at the right hand side of the control command that is about to be deleted.

Please note that Deleting a control command is impossible for service control items when there is no other control command left for the service control object.

Each control command can be adjusted towards the following options:

Install action

The command to apply during the installation sequence of the target package may be any combination of:

- Start - starts the service during the StartServices action.
- Stop - stops the service during the StopServices action.
  Stopping a service also stops all depending services
- Delete - deletes the service during the DeleteServices action.
  Deleting a service causes the installer to stop it before it is actually deleted

Uninstall action

The command to apply during the uninstallation sequence of the target package may be any combination of:

- Start - starts the service during the StartServices action.
- Stop - stops the service during the StopServices action.
  Stopping a service also stops all depending services
- Delete - deletes the service during the DeleteServices action.
  Deleting a service causes the installer to stop it before it is actually deleted

Wait Type

Users may decide to let the installation wait for the SCM (Service Controll Manager) to report the service to be in a pending state before the installation proceeds, or to wait for max. 30 seconds for the service to complete before the installation proceeds.

Argument

Add parameters to the command line used to run the service. If more than one parameter is added, they have to be separated by null ("`[~]`").

Tab: Configure Settings

Within this tab, users may manage Events and Recovery actions for the service manipulation event. Both sub-tabs are displayed below the main tab navigation bar.

Sub-tab: Events

Service manipulation event objects are internally stored within the Installer database table `MsiServiceConfig`. A list of already created events is displayed when this sub-tab is loaded. To display the details for an event, the downwards arrow icon left of each events title bar has to be clicked. (Another click on the now visible upwards arrow icon collapses the details pane.)

To add an event, users click on the Add button above the list of already available event items,

and adjust the default settings of the newly created object.

To delete an event, users click on the Delete button that is displayed at the right-hand side of each event items data block.

The following event properties may be edited:

Event trigger
Events may be triggered by Install, Uninstall, and/or Reinstall - the actual combination of active triggers is an arbitrary user definition.

Wait type and **Argument**
The set of available options for the Argument property actually depends on the selected Wait type:

- Wait type: Configure the time delay of an auto-start service
  Argument options:
  o Turn off the auto-start service delay
  o Start the service after other auto-start services plus time delay

- Wait type: Change the list of privileges required by the service
  Argument options:
  o The argument string required to do so may be changed within the SERVICE PERMISSIONS dialog. Click on the edit argument link at the right-hand side of the argument input field to open the editor dialog. Activate the checkbox in the outer left column of the privilege list to define the privilege to be required for the service. Deactivate checkboxes to remove existing requirements.

- Wait type: Add a service SID type to the process token containing this service
  Argument options:
  o None
  o Restricted
  o Unrestricted

- Wait type: Configure the length of the time the SCM waits before proceeding with other shutdown operations
  Argument options:
  o The required argument value is an integer, resembling the seconds the SCM should wait.

- Wait type: Configure when to run the failure actions for this service
  Argument options:
  o Run the actions only if the service terminates without reporting SERVICE_STOPPED
  o Run the actions if the service terminates reporting SERVICE_STOPPED

Sub-tab: Recovery Actions

Service manipulation recovery actions are internally stored within the Installer database table

`MsiServiceConfigFailureOptions`. They define actions that have to be executed at the next system start after a failed service run.

A list of already created recovery actions is displayed when this sub-tab is loaded. To display the details for an action, the downwards arrow icon left of each items title bar has to be clicked. (Another click on the now visible upwards arrow icon collapses the details pane.)

To add a recovery action, users click on the Add button above the list of already available event items, and adjust the default settings of the newly created object.

To delete a recovery action, users click on the Delete button that is displayed at the right-hand side of each action items data block.

The following recovery action properties may be edited:

Trigger
Recovery actions may be triggered by Install, Uninstall, and/or Reinstall - the actual combination of active triggers is an arbitrary user definition and translated into a bit flag value.

Reset Period
The reset period defines the time (in seconds) of successful service up time that is required before the SCM failure count for the specific service is reset.

Reboot Message
If a reboot is due, the entered value will be displayed before the system restarts.

Command
If a command has to be executed as part of the recovery routine, it has to be defined here. If no value is given, the existing command will be used unchanged. Entering null ("`[~]`") suppresses any command execution.

SCM Actions
Each recovery action may include one or more Service Control Manager (SCM) actions. Internally, these actions are stored as a matching pair of sequence arrays within the MsiServiceConfigFailureAction table columns Actions and DelayActions.

A list of already created recovery actions is displayed when the details pane of a recovery action is loaded. To display the details for an SCM action, the downwards arrow icon left of each items title bar has to be clicked. (Another click on the now visible upwards arrow icon collapses the SCM actions details pane.)

To *add an SCM action*, users click on the Add button above the list of already available SCM action items, and adjust the default settings of the newly created object.

To *delete an SCM action*, users click on the Delete button that is displayed at the right-hand side of each SCM action items data block.

For each SCM action, users may edit the following properties:

Action
Users have to pick from the available set of actions:
- Restart service
- Reboot computer
- Run command
- No action

Delay
The milliseconds to wait before the action defined above is actually executed.

ODBC

This view allows you to manipulate ODBC resources that will be installed on the package target system. The ODBC standard developed by Microsoft allows a program to access data in any compliant relational database language. Open Database Connectivity (ODBC) related registry information is stored at `HKCU\Software\ODBC` and `HKLM\Software\ODBC`. Data objects stored within the Installer database tables starting with ODBC (`ODBCAttribute`, `ODBCDataSource`, `ODBCDriver`, `ODBCSourceAttribute`, and `ODBCTranslator`) are organizational helpers for proper registry handling at package run-time.

There are three different object types related to ODBC management:

- **Drivers**
  An ODBC driver provides an API that allows to access and control a specific DBMS (Database Managemen System). In order to externally access a specific database type, the matching ODBC driver has to be installed on the active device.
  Attributes for ODBC drivers are stored within the Installer database table `ODBCAttribute`.

- **Data Sources**
  Data sources are required to establish connections to specific database instances via their parent DBMS. The data source string format depends on the target database type, and therefore on the driver used to communicate with that database type.
  Attributes for ODBC data sources are stored within the Installer database table `ODBCSourceAttribute`.

- **Translators**
  Translators are used to provide control intelligence, e. g. for code page translation between system objects that take part in the database communication.
  Attributes for ODBC translators are stored within the Installer database table `ODBCAttribute`.

During package run-time, the standard sequence actions InstallODBC and RemoveODBC are responsible for transferring ODBC related data from the Installer database tables into the right target system areas.

ODBC View Areas

The ODBC view is divided into a list of already existing data objects (drivers, data sources, and translators) on the left and a details pane on the right, which shows the properties of the currently selected object from the item list.
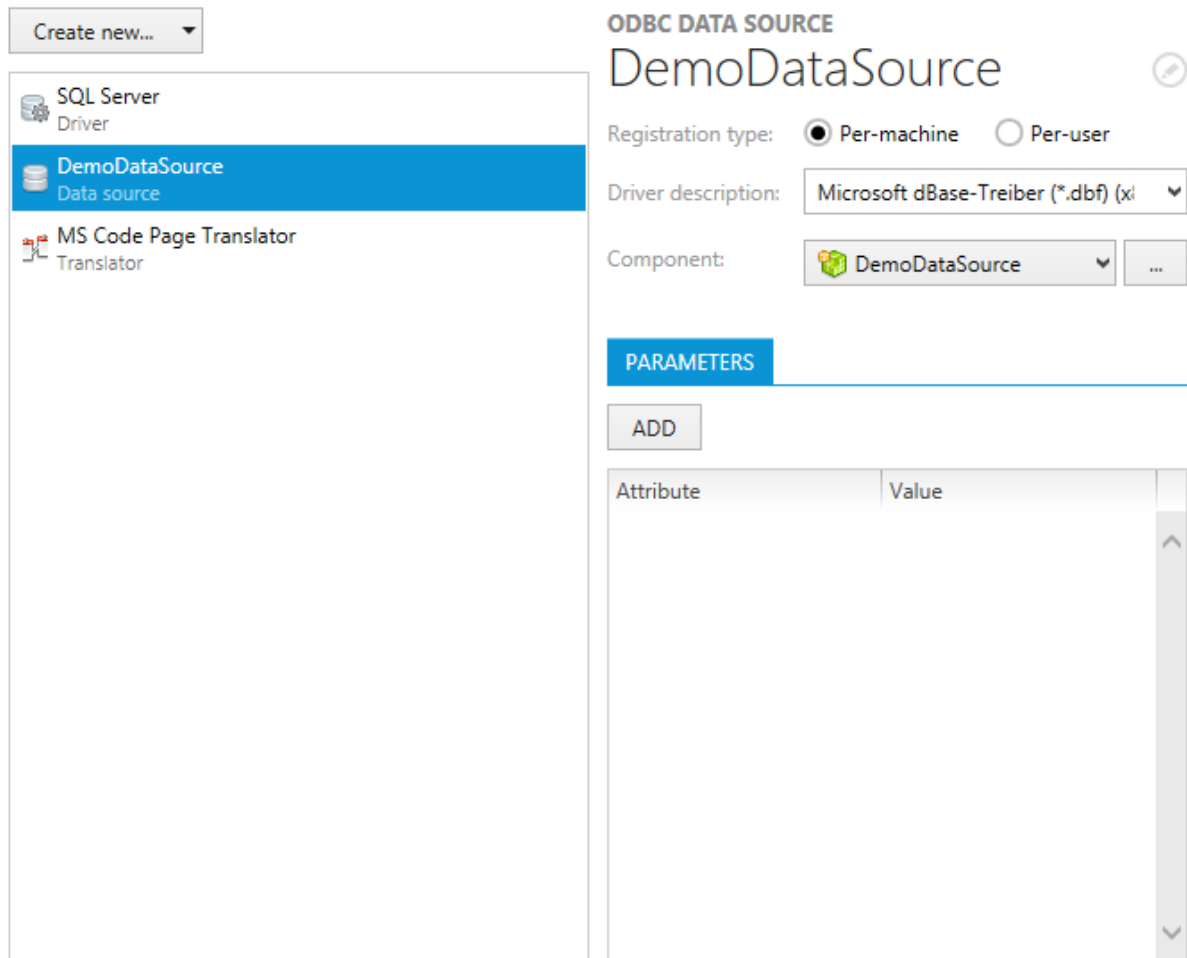
The details pane of the view is used to display and edit ODBC item properties. Please refer to *Edit an ODBC item* for a detailed description.

To access the Installer database counterpart of an ODBC item, users have to right-click its list object, and select **Go to row** from the context menu. The type specific database table is loaded, with a highlight set on the matching data row.

Standard ODBC Management Procedures

Common management procedures for ODBC related objects are described within the following topics:

- *Add an ODBC driver*
- *Add an ODBC data source*

- *Add an ODBC translator*
- *Import of ODBC objects*
- *Rename an ODBC item*
- *Remove an ODBC item*
- *Edit an ODBC item*

Add an ODBC Driver

In order to add an ODBC driver to a packaging project, users go to the ODBC view of the Visual Designer mode.

With a click on the **Create new...** button, an option menu is displayed. Select Driver to launch the required wizard interface.

Work your way through the steps of the wizard to define all required properties for the new item.

At any time, using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, allows to navigate within the already processed steps.

To exit the wizard without creating a new object, use the Cancel button, also located at the bottom of the wizard dialog.

Step 1: Driver Type

Select the type of the new driver object:

- **Custom** drivers have to be defined by providing the DLL with the driver logic. It is the recommended choice for quite special ODBC driver requirements.
- **Predefined** drivers are offered for integration into the packaging project from a list of drivers which are available on the current packaging machine. It is the recommended choice for common driver addition.

Click on the tile that represents the driver type which has to be added to the packaging project, and click **Next** to proceed.

Step 2: Type Specific Information

Component

When a new ODBC driver is added to a packaging project, the DLL resource has to be added to a specific component. RayPack usually creates a new component for the driver resources, as indicated by the default activation for the "Automatically create a new component" radio selector option. However, by selecting the other option "Select existing component", a drop-down menu with available components becomes visible. Select one of them, or manually define the properties of a new component if required.

Custom Driver

ODBC Driver name

The name has to be given as an alphanumerical string, which may contain the ASCII characters A-Z (a-z), digits (0-9), underscores (_), or periods (.). It must begin with either a letter or an

underscore. The name must be unique within the ODBC items of the same type. (e. g. there may not be several ODBC driver items with the same name, whilst there may be a translator and a driver item with the same name)

ODBC Driver file
The DLL with the driver logic. It will be added to the objects stored within the packaging projects `File` table, and referenced in the `ODBCDriver` table via its identifier.

ODBC Driver setup file
The DLL with the logic required to install the driver. It will be added to the objects stored within the packaging projects `File` table, and referenced in the `ODBCDriver` table via its identifier. The setup file is optional, and only required if additional driver installation logic is needed to actually use the driver on the later target machines.

Predefined Driver

Driver
Select one of the drivers from the predefined list. The required resources will automatically be derived from the local machine and added to the packaging project.
If the required driver is not part of the list, use the Back button, select the type Custom driver, and upload the driver resources manually in the following wizard step.
Please keep in mind that the driver architecture must match the target machine architecture. Therefore, select the driver with the matching property ("x86" or "x64") displayed within the labels of the driver option list.

Step 3: Summary

Use the summary page to check the correctness of the driver properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the ODBC item.
- If changes are due, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the summary page again.

Step 4: Finished

Once the new ODBC driver has been created, the wizard can be closed by using the **Finish** button at its lower right corner. The ODBC view is updated, and the list of existing items contains the newly created object at the lowest position.

Add an ODBC Data Source

In order to add an ODBC data resource to a packaging project, users go to the ODBC view of the Visual Designer mode.
With a click on the **Create new...** button, an option menu is displayed. Select Data source to launch the required wizard interface.

Work your way through the steps of the wizard to define all required properties for the new item.

At any time, using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, allows to navigate within the already processed steps.
To exit the wizard without creating a new object, use the Cancel button, also located at the bottom of the wizard dialog.

Step 1: New Data Source

Name
The name has to be given as an alphanumerical string, which may contain the ASCII characters A-Z (a-z), digits (0-9), underscores (_), or periods (.). It must begin with either a letter or an underscore. The name must be unique within the ODBC items of the same type. (e. g. there may not be several ODBC data source items with the same name, whilst there may be a data source and a driver item with the same name)

Registration type
Decide whether the data source should be available for any user on the target machine (activate the Per-machine option), or exclusively for a specific user (activate the Per-user option).

Driver description
Select the driver used in combination with the new data source. The predefined list contains all drivers that are available on the local packaging machine and also all custom drivers that have previously been added to the packaging project.

Component
When a new ODBC data source is added to a packaging project, it has to be added to a specific component. RayPack usually creates a new component for the data source, as indicated by the default activation for the "Automatically create a new component" radio selector option. However, by selecting the other option "Select existing component", a drop-down menu with available components becomes visible. Select one of them, or manually define the properties of a new component if required.

Step 2: Summary

Use the summary page to check the correctness of the data source properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the ODBC item.
- If changes are due, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the summary page again.

Step 3: Finished

Once the new ODBC data source has been created, the wizard can be closed by using the **Finish** button at its lower right corner. The ODBC view is updated, and the list of existing items contains

the newly created object at the lowest position.

Add an ODBC Translator

In order to add an ODBC translator to a packaging project, users go to the ODBC view of the Visual Designer mode.
With a click on the **Create new...** button, an option menu is displayed. Select Translator to launch the required wizard interface.

Work your way through the steps of the wizard to define all required properties for the new item.

At any time, using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, allows to navigate within the already processed steps.
To exit the wizard without creating a new object, use the Cancel button, also located at the bottom of the wizard dialog.

Step 1: New Translator

Component
When a new ODBC translator is added to a packaging project, it has to be added to a specific component. RayPack usually creates a new component for the translator, as indicated by the default activation for the "Automatically create a new component" radio selector option.
However, by selecting the other option "Select existing component", a drop-down menu with available components becomes visible. Select one of them, or manually define the properties of a new component if required.

Translators
Please select the new translator from the predefined list. The list consists of translators, which are defined for the current packaging machine.

If the desired translator is not available, it has to be added to the local packaging machine (by manipulating the following registry location: `HKLM\SOFTWARE\ODBC\ODBCINST.INI`), or uploaded into the packaging project manually. The latter requires to add the source file via Files and Folders, and create a new row for the `ODBCTranslator` table from the **TABLES** view of the Advanced mode, which refers to the translator source file.

Step 2: Summary

Use the summary page to check the correctness of the translator properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the ODBC item.
- If changes are due, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the summary page again.

Step 3: Finished

Once the new ODBC translator has been created, the wizard can be closed by using the **Finish** button at its lower right corner. The ODBC view is updated, and the list of existing items contains the newly created object at the lowest position.

Import of ODBC Entries

In order to import ODBC drivers, data sources, and translators from a `.rpodbc` into a packaging project, users go to the ODBC view of the Visual Designer mode.
With a click on the **Create new...** button, an option menu is displayed. Select Import to launch the required wizard interface.

Step 1: Select File

Click on the **Browse** button [**...**] and browse to the directory where the file is saved.
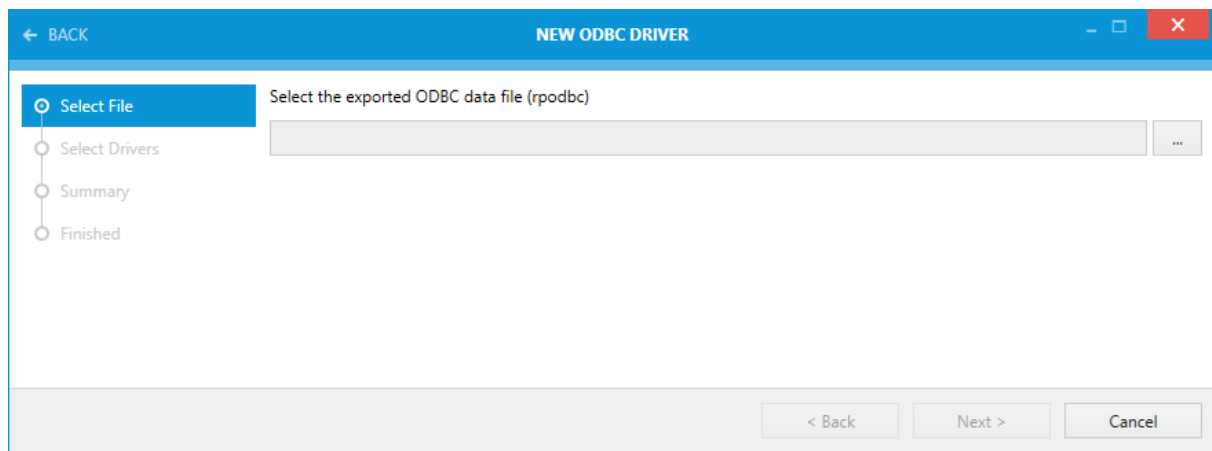
After the file has been selected, click on the **Next >** button to continue to the next step.

Step 2: Select Drivers

Select those entries for drivers, data sources, and translators from the file, that are to be imported into the packaging project.
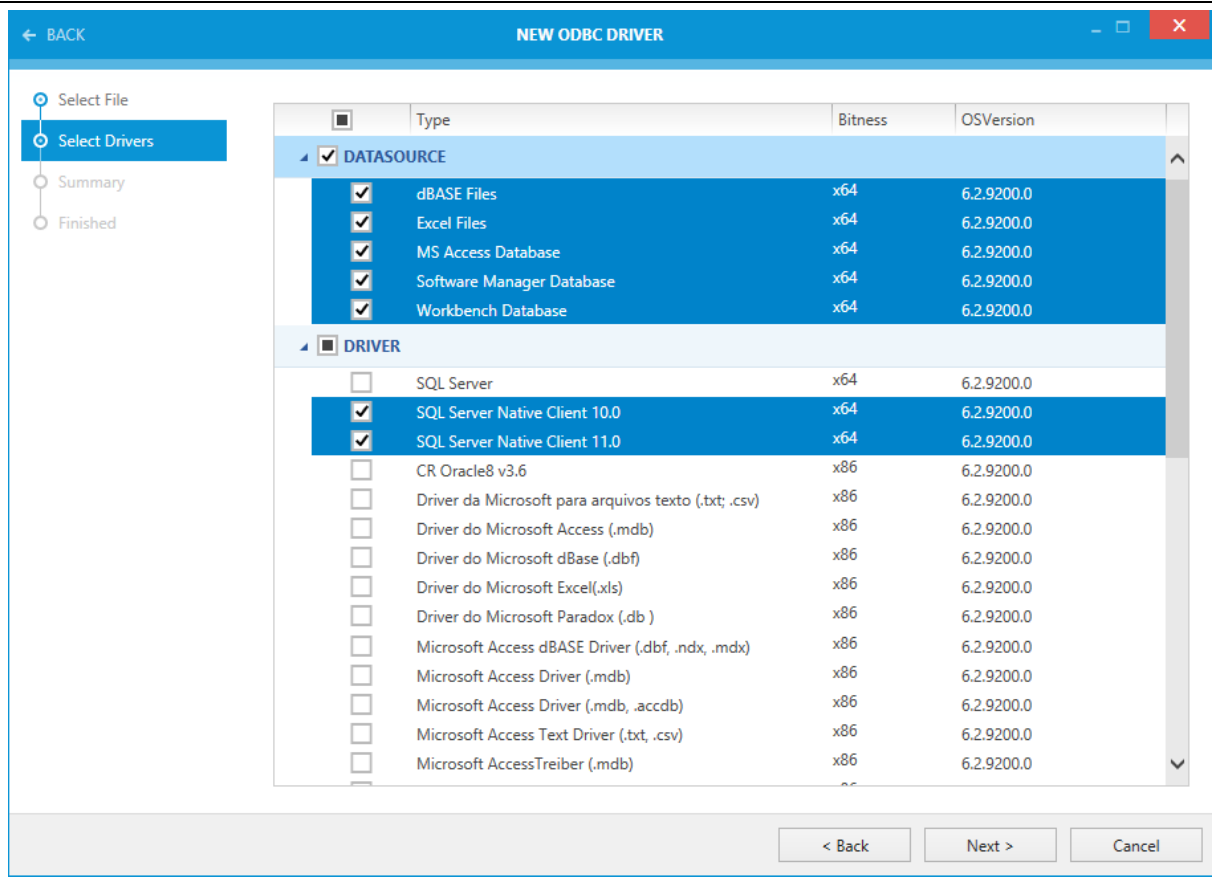
When all entries that are to be imported have been selected, click on the **Next >** button to continue.

Step 3: Summary

Use the summary page to check the correctness and completeness of the entries that were selected during the previous wizard steps.
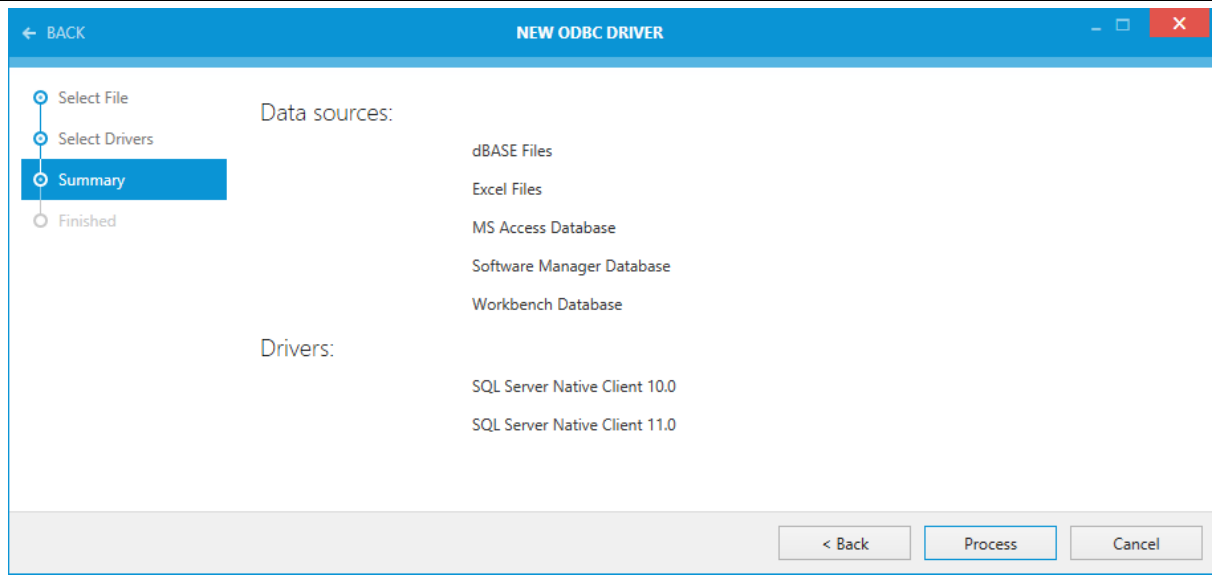
- If all properties are set as required, click **Process** to import the ODBC entries.
- If changes are due, click < **Back** until the wizard step is displayed where the modifications are to be made.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst clicking on **Next >** until the summary page is reached again.

If **Process** has been clicked a progress bar will be shown until all selected entries have been imported into the packaging project.

Step 4: Finished

Once all ODBC entries have been imported, the wizard can be closed by using the **Finish** button at its lower right corner.
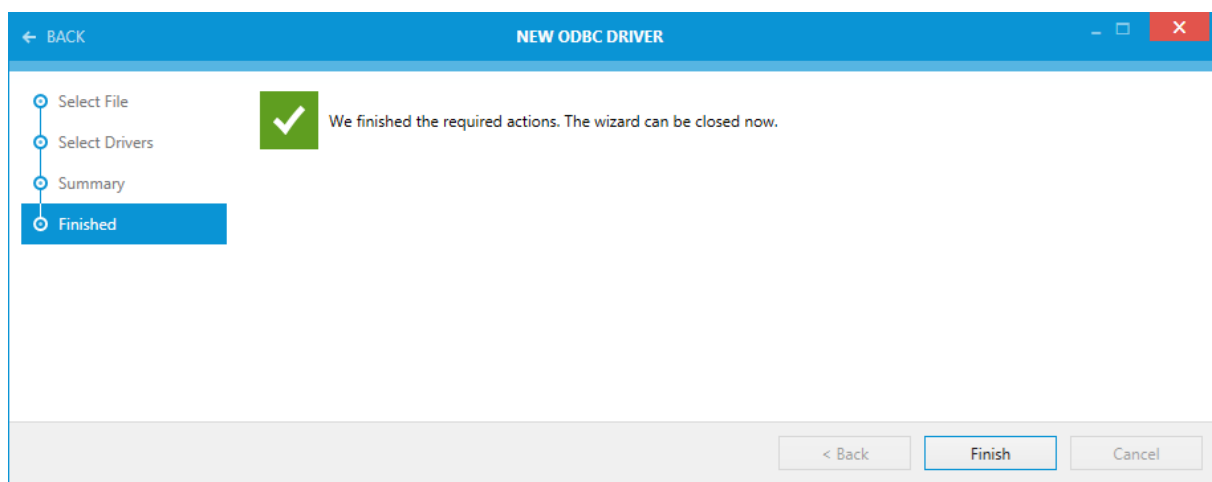
The ODBC view is updated, and the list of existing items contains the newly imported entries in

alphabetical order at the lowest position of the respective entry types.

Rename an ODBC Item

1. To rename an ODBC item, users load the list of existing objects by calling the ODBC view within the Visual Designer mode of PackDesigner.

2. Clicking on the list item of the object that has to be renamed displays its details in the editor panel on the right hand side of the RayPack application screen.

3. The object name is displayed above the form elements of the edit area.

4. The name becomes editable with a click on either the text itself, or the edit icon on the right hand side. A *direct value editor dialog* is displayed, allowing to enter the new value.

   The name of the ODBC item has to be given as an alphanumerical string, which may contain the ASCII characters A-Z (a-z), digits, underscores (_), or periods (.). It must begin with either a letter or an underscore. The name must be unique within the ODBC items of the same type. (e.g. there may not be several ODBC translator items with the same name, whilst there may be a translator and a driver item with the same name)

5. By hitting Enter or clicking on the save icon within the direct value editor interface the new value is saved.

Remove an ODBC Item

To remove an ODBC item from a packaging project by using the PackDesigner Visual Designer mode interface functionality:

1. Users have to call the ODBC view. From the list of items on the left hand side**, right-click** an item and select **Remove** from the **context menu**

2. A dialog is displayed, requesting the user to confirm the object deletion.

3. The ODBC object is immediately deleted from the packaging project when the user clicks on REMOVE to confirm.

Be aware:
As soon as an item is deleted from the ODBC view, its resembling data row as well is deleted from the affected Installer database tables.
Directly depending objects, such as attributes, are deleted along with the basic ODBC object.
Please be aware that object relations, such as a link between driver and data source, may get broken due to the deletion of one of the related objects. It is recommended to check for references within the ODBC object pool before an ODBC item is actually deleted.

Edit an ODBC Item

To edit an ODBC item, users load the list of existing objects by calling the ODBC view within the Visual Designer mode of PackDesigner. Clicking on the list item of the object that has to be edited displays its details in the editor panel on the right hand side of the RayPack application screen.

The amount of changeable properties depends on the specific ODBC item type:

Edit an ODBC Driver

Name

To edit the driver name, users have to either click on the name or the edit icon displayed at the right-hand side of the current name. A *direct inline value editor dialog* is displayed. Please refer to the common dialogs section for details on how to handle this type of editor interface.

The name has to be given as an alphanumerical string, which may contain the ASCII characters A-Z (a-z), digits (0-9), underscores (_), or periods (.). It must begin with either a letter or an underscore. The name must be unique within the ODBC items of the same type (e. g. there may not be several ODBC driver items with the same name, whilst there may be a translator and a driver item with the same name).

Component

Each ODBC driver has to be related to a component. To change the current relation, users may either select another component from the drop-down list, or create a new component with a click on the **[...]** button at the right-hand side of the current component value. The *select a component* dialog is displayed in this case. Please refer to the common dialogs section for details on how to handle this type of editor interface.

File and Setup File

These two properties are not editable via the interface provided by the Visual Designer. If a driver does not have the right values within these fields, it has to be removed from the packaging project, and added again with the correct properties.

> **Be aware:**
> However, all packaging project properties, which are stored within the Installer database in the background, may be edited directly via the **TABLES** view of the Advanced mode. Even though manual changes may lead to severe inconsistency within the packaging project, experienced packagers might want to perform them anyway.

Parameters

To add a new parameter to the ODBC driver, users have to click on the **Add** button, which is available within the PARAMETERS tab of the details pane, displayed at the right-hand side of the ODBC dialog as soon as a driver item has been selected from the item list on the left. A new parameter is created, with the default string "`attribute`" set for the attribute column, and an empty value. Attributes for ODBC drivers are stored within the Installer database table `ODBCAttribute`.

To change either the attribute or the value column content of a specific parameter row, users have to double-click the cell and start to type the new value. Hitting enter saves the new content. As an alternative, it is also possible to click the cell that has to be modified, and hit F2 on

the keyboard. The cell is set into direct inline edit mode, ready for user input to replace the existing content.

A parameter can be removed by right-clicking the parameter row, and selecting **Remove** from the context menu.

Edit an ODBC Data Source

Name

To edit the data source name, users have to either click on the name or the edit icon displayed at the right-hand side of the current name. A *direct inline value editor dialog* is displayed. Please refer to the common dialogs section for details on how to handle this type of editor interface.

The name has to be given as an alphanumerical string, which may contain the ASCII characters A-Z (a-z), digits (0-9), underscores (_), or periods (.). It must begin with either a letter or an underscore. The name must be unique within the ODBC items of the same type. (e. g. there may not be several ODBC data source items with the same name, whilst there may be a translator and a data source item with the same name)

Registration Type

Data sources may either be available per-machine, which enables all users with valid credentials for the machine to use it, or per-user, which restricts the usage for any other user than the one the parent package is installed for. The recommended (and default) setting is per-machine.
To change the current setting, users switch the radio selector control to activate the option that has to be set.

Component

Each ODBC data source has to be related to a component. To change the current relation, users may either select another component from the drop-down list, or create a new component with a click on the **[...]** button at the right-hand side of the current component value. The *select a component* dialog is displayed in this case. Please refer to the common dialogs section for details on how to handle this type of editor interface.

Parameters

To add a new parameter to the ODBC data source, users have to click on the **Add** button, which is available within the PARAMETERS tab of the details pane, displayed at the right-hand side of the ODBC dialog as soon as a data source item has been selected from the item list on the left. A new parameter is created, with the default string "`attribute`" set for the attribute column, and an empty value. Attributes for ODBC data sources are stored within the Installer database table `ODBCSourceAttribute`.

To change either the attribute or the value column content of a specific parameter row, users

have to double-click the cell and start to type the new value. Hitting enter saves the new content. As an alternative, it is also possible to click the cell that has to be modified, and hit F2 on the keyboard. The cell is set into direct inline edit mode, ready for user input to replace the existing content.

A parameter can be removed by right-clicking the parameter row, and selecting **Remove** from the context menu.

> **Be aware:**
> When a parameter is deleted, there is no intermediate confirm dialog. Therefore, clicking Remove from the context menu as described above will immediately remove the data object.

Edit an ODBC Translator

Name

To edit the translator name, users have to either click on the name or the edit icon displayed at the right-hand side of the current name. A *direct inline value editor dialog* is displayed. Please refer to the common dialogs section for details on how to handle this type of editor interface.

The name has to be given as an alphanumerical string, which may contain the ASCII characters A-Z (a-z), digits (0-9), underscores (_), or periods (.). It must begin with either a letter or an underscore. The name must be unique within the ODBC items of the same type. (e. g. there may not be several ODBC translator items with the same name, whilst there may be a translator and a data source item with the same name)

Component

Each ODBC translator has to be related to a component. To change the current relation, users may either select another component from the drop-down list, or create a new component with a click on the **[...]** button at the right-hand side of the current component value. The *select a component* dialog is displayed in this case. Please refer to the common dialogs section for details on how to handle this type of editor interface.

File & Setup File

These two properties are not editable via the interface provided by the Visual Designer. If a translator does not have the right values within these fields, it has to be removed from the packaging project, and added again with the correct properties.

> **Be aware:**
> However, all packaging project properties, which are stored within the Installer database in the background, may be edited directly via the TABLES view of the Advanced mode. Even though manual changes may lead to severe inconsistency within the packaging project, experienced packagers might want to perform them anyway.

Drivers

This view allows you to manipulate drivers that will be installed on the target system.
The drivers view is divided into a list of already existing data objects on the left, and a details pane on the right, which shows the properties of the currently selected object from the item list.

The details pane of the view is used to display and edit driver item properties. Please refer to *Edit a driver* for a detailed description. The object list on the left-hand side is ordered according to the actual installation order of the available driver items. To reorder them, users have to right-click any of the present items, and select one of the move options from the context menu:

- Install first - moves the driver object to the first position of the installation order.
- Install earlier - moves the driver object one position up within the installation order.
- Install later - moves the driver object one position down within the installation order.
- Install last - moves the driver object to the last position of the installation order.

Please keep in mind that the order of driver installation steps may be crucial where dependencies and system requirements are concerned. It is recommended to check the actual order once all driver packages have been added to a packaging project. Since this order is of vital importance for successful package run-times, the list of drivers cannot be re-ordered by manually clicking on the column headers, as it is known as usual interface control option for most

other list views within RayPack.

To access the Installer database counterpart of a driver item, users have to right-click its list object, and select **Go to row** from the context menu. The type specific database table is loaded, with a highlight set on the matching data row.

Standard Driver Management Procedures

Common management procedures for driver objects are described within the following topics:

- *Add a driver*
- *Remove a driver*
- *Edit a driver*

Add a Driver

In order to add a driver to a packaging project, users go to the Drivers view of the Visual Designer mode.

> **Note:**
> Using the New Driver wizard from the Visual Designer View assumes that the driver data is not a part of the current package and that it is supposed to be imported as a part of driver creation. If you already have a component having necessary data and a valid KeyPath that points to the `.inf` file, then use the Component View inside the Advanced Designer, which allows a fine control over the existing components.

With a click on the **Create new...** button, the New Driver wizard is invoked.

Work your way through the steps of the wizard to define all required properties for the new item.

At any time, using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, allows to navigate within the already processed steps.
To exit the wizard without creating a new object, use the **Cancel** button, also located at the bottom of the wizard dialog.

Step 1: INF File

INF file
Use the browse button at the right-hand side of the INF file path info field to browse for the required driver file.
Select the INF file and click Open to use it as driver data source.

> **Note:**
> The system browser dialog is by default opened with a type restriction, allowing to select `INF files (*.inf)` only. However, it is possible to add driver files of other types here as well once the file type filter is set to `All files (*.*)`. Even though RayPack allows to do that, it is recommended for experts use only. The mechanisms of the driver wizard are optimized for inf-based drivers, and may deliver unexpected

results for other driver types.

Target platform

The target platform may be changed for drivers that are added to packaging projects with a 32-bit target platform. As soon as a package has a global target platform with a 64-bit architecture, this option is activated by default and read-only.

> **Note:**
> It is highly recommended to keep the target platform of the driver object and the target platform of the packaging project itself closely aligned. If there are reasons to set them to different values, these circumstances should be validated and documented carefully.

Click **Next** to proceed.

Step 2: Supporting Files

Automatically Determined Supporting Files

When a driver file is added, RayPack automatically reads the dependencies determined per INF file. If there is readable information about required `*.sys`, `*.cat`, `*.drv`, or `*.dll` files, this automatism adds them to the packaging project as files that depend on the newly created driver object.

Manually Add Supporting Files

If there are additional files known to be required for the driver to operate, users may add them manually. To do so, the **Add** button above the list of already added supporting files has to be clicked. A system browser dialog is displayed, enabling the user to select any **`*.sys, *.cat, *.drv,`** or **`*.dll`** file, and add it to the list of supporting files by clicking on the **Open** button.

Each new file is added to the end of the list of already added supporting files. Please be aware that the supporting files will be added to the packaging project exactly as given in the list of supporting files. Restrictions regarding required installation orders have to be meet by adding them in the correct order. Later re-adjustments are error-prone and should be avoided.

Remove Supporting Files

If a file should not be installed along with the new driver object, but has been added to the list of supporting files either manually or automatically, users have the option to remove them from that list. To do so, users select one or more files from within the list, and click on the Remove button above the list area. The Remove button is inactive as long as no file has been selected, but becomes available as soon as at least one file is currently selected.

The selected files will be removed from the list - immediately and without any intermediate confirm step.

Create Separate Components for Supporting Files

According to Best Practices, it is recommended to create a component per supporting file of a driver. These files are typically executables, which should be key path of their very own component. The checkbox for automated component creation is active by default in order to allow effortless Best Practice compliance.

Step 3: Options

Create the Add / Remove Programs entry for a driver package
If this checkbox is active, the driver will be visible within the ARP dialog of the packages target system. In order to provide transparency on the target system, it is recommended to do so.

Show the Connect Device dialog on install
It is possible to suppress the Connect Device dialog by deactivating this checkbox. If it is left active (as it is per default), the installing user will be ordered to connect a matching Plug and Play device at package run-time if the device recognition has not detected a matching device yet.

Always overwrite existing driver
If a similar driver is already present on the target system, the new driver brought along within the package will not be installed automatically. To force the installation in such a situation, this option needs to be activated manually.

Remove the binaries from system folder when the driver is being uninstalled (not recommended)
It is part of Best Practice to keep files that have been created at specific system folders during package installation, even beyond the package uninstallation. If other (vital) parts of the target device rely on these system files, their removal could cause severe harm. Therefore, it is recommended to keep this checkbox inactive. Please double-check the necessity of removing system folder files before it is actually activated.

Allow the installation of unsigned / incomplete drivers
It is not common to allow the installation of unsigned or incomplete driver packages on a device. However, there may be circumstances where it is required to do so. Activating this checkbox overrules standard system settings, which would suppress the installation of usually unfit driver packages. Please double-check the necessity of allowing this type of resource installation before this checkbox is actually activated.

Step 4: Summary

Use the summary page to check the correctness of the driver properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the driver item.
- If changes are due, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the summary page again.

Step 5: Finished

Once the new driver has been created, the wizard can be closed by using the **Finish** button at its lower right corner. The Drivers view is updated, and the list of existing items contains the newly created object at the lowest position of the installation order.

Remove a Driver

To remove a driver item from a packaging project by using the PackDesigner Visual Designer mode interface functionality:

1. Users have to call the Drivers view. From the list of items on the left hand side:

    • **Right-click** an item and select **Remove** from the **context menu**

2. The driver object is immediately deleted from the packaging project.

Be aware:
As soon as an item is deleted from the Drivers view, its resembling data row as well is deleted from the affected Installer database tables.

Edit a Driver

1. To edit a driver object, users load the list of existing items by calling the Drivers view within the Visual Designer mode of PackDesigner.

2. Clicking on the list item of the object that has to be modified displays its details in the editor panel on the right hand side of the RayPack application screen.

The following properties of the driver object may be manipulated:

User Interaction

Prompt for device on install
It is possible to suppress the Connect Device dialog by de-activating this checkbox. If it is active (as it is per default), the installing user will be ordered to connect a matching Plug and Play device at package run-time if the device recognition has not detected a matching device yet.

Create Add / Remove Programs entry for a driver
If this checkbox is active, the driver will be visible within the ARP dialog of the packages target system. In order to provide transparency on the target system, it is recommended to do so.

Installation and Uninstallation

Allow installation of missing or unsigned files
It is not common to allow the installation of unsigned or incomplete driver packages on a device. However, there may be circumstances where it is required to do so. Activating this checkbox overrules standard system settings, which would suppress the installation of usually unfit driver packages. Please double-check the necessity of allowing this type of resource installation before this checkbox is actually activated.

Always overwrite existing driver
If a similar driver is already present on the target system, the new driver brought along within

the package will not be installed automatically. To force the installation in such a situation, this option needs to be activated manually.

Remove binary files on uninstall (not recommended)
It is part of Best Practice to keep files that have been created at specific system folders during package installation, even beyond the package de-installation. If other (vital) parts of the target device rely on these system files, their removal could cause severe harm. Therefore, it is recommended to keep this checkbox inactive. Please double-check the necessity of removing system folder files before it is actually activated.

Condition

It is possible to control the driver installation by the definition of conditions. These conditions have to be fulfilled in order to trigger the driver installation. Conditions may either be defined by typing the clause manually into the conditions input field, or by using the condition builder.
To use the assisting condition builder, users have to click on the button "edit in condition builder..." which is displayed below the input field. Please refer to the *Common dialogs* section *Condition builder* to get details on the options available within this special assistance interface.

Database and Server

The Database & Server view allows you to make changes to the SQL Databases, SQL Scripts, IIS, and Scheduled Tasks settings.

> **Note:**
> If any of the items listed above is not available within your RayPack installation, check for the actual set of features covered by your license. To do so, visit the About section and expand the license details with the full listing of licensing options.

SQL Databases

This view allows you to install and manipulate SQL databases on the target system. The provided functionality is designed to support SQL database management only. Other database types have to be installed and manipulated by individual custom action measures.

> **Note:**
> Another view of the Visual Designer, SQL scripts, bases on the SQL database objects created within this view. Identifying the target database for SQL script execution is only possible, if a database object itself has been provided with connection information before. Therefore, adding SQL database objects in RayPack is not only required if a new database has to be created on the later target machines of the package, but as well if SQL scripts have to run on already existing SQL databases on those target machines. SQL database objects may be used as mere database connection definition references.

Since database handling is not part of the MSI Standard, RayPack utilized special Installer database tables to manage SQL database information until a package is built from the project:

- `RPSqlDatabase` - Contains core information about the database object, such as the parent server.

- `RPSqlFileSpec` - Contains data about SQL database files handling options, such as log file specifications
- `RPUser` - Contains user profiles required for SQL database connections. User objects are required for SQL database and *SQL script* handling.

If these tables have not been present within the current packaging project before, they are automatically added with the first SQL Database object created within the project. Please refer to the section about *Custom RayPack installer database tables* for further details and table specifications.

The SQL Databases view is separated into two main areas:
- A list of already available SQL database objects on the left-hand side
- A details pane for viewing and editing the properties of already existing SQL database objects at the right-hand side
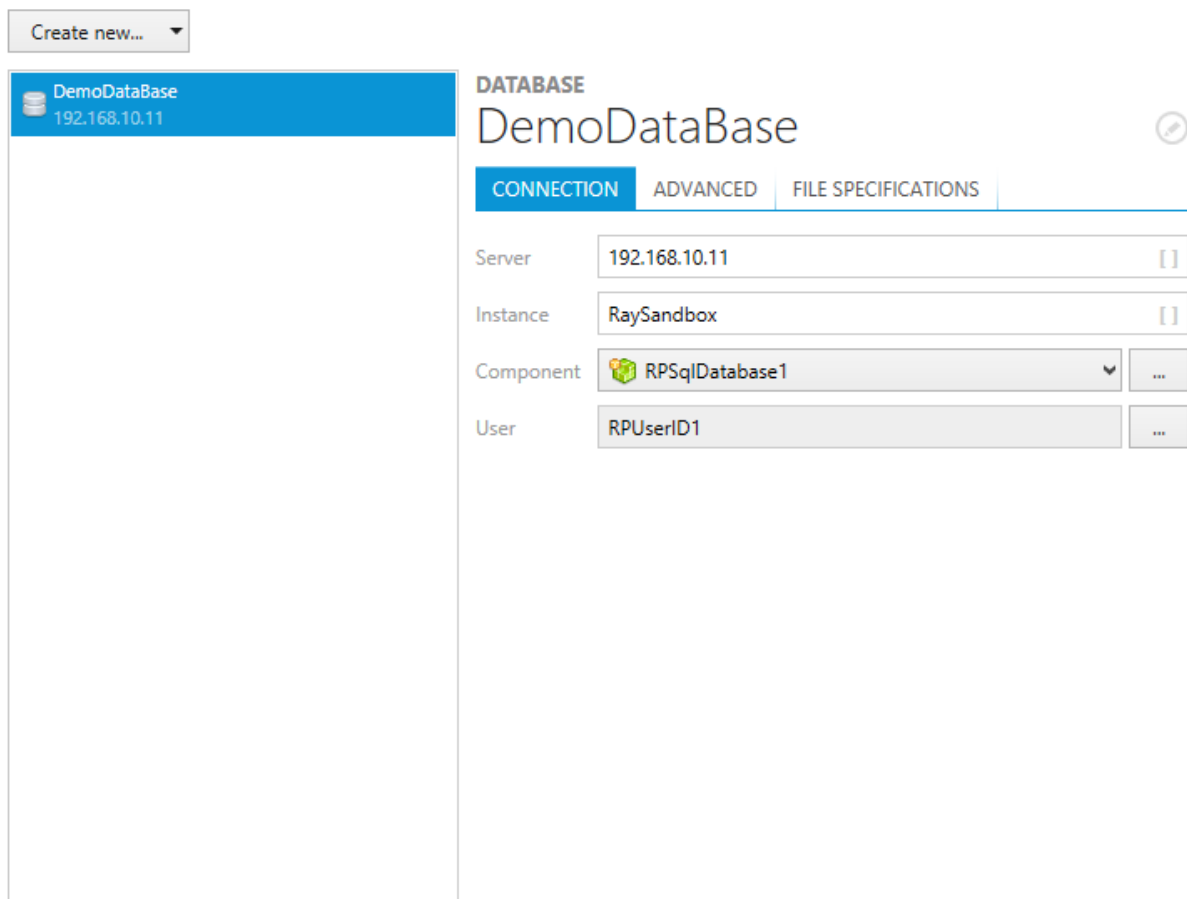
To access the Installer database counterpart of an SQL database item, users have to right-click its list object, and select **Go to row** from the context menu. The database table `RPSqlDatabase` is loaded, with a highlight set on the matching data row.

Standard SQL Database Management Procedures

Common management procedures for SQL database objects are described within the following topics:

- *Add an SQL database*
- *Remove an SQL database*
- *Edit an SQL database*

Add an SQL Database

> 📎 **Note:**
> Adding SQL database objects in RayPack is not only required if a new database has to be created on the later target machines of the package, but as well if SQL scripts have to run on already existing SQL databases on those target machines. SQL database objects may be used as mere database connection definition references.

In order to add an SQL database to a packaging project, users go to the **SQL Databases** view of the Visual Designer mode.
With a click on the **Create new...** button, the options menu is displayed. Click on **SQL database** to invoke the New SQL Database Wizard.

Work your way through the steps of the wizard to define all required properties for the new item.

At any time, using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, allows to navigate within the already processed steps.
To exit the wizard without creating a new object, use the Cancel button, also located at the bottom of the wizard dialog.

Step 1: General

Server
The physical address of the database server. This property is mandatory. It may either be defined by directly entering a string value of max. 255 characters length, or by using one of the variables available within the packaging project (e. g. a property). To trigger support for dynamic content input, users have to type an opening square bracket. The dialog for *MSI formatted string input fields* is displayed. Please refer to the *common dialogs* section for further details on how to handle this type of input control.

Instance
The instance name of the target SQL database object. If this optional value is not set, the default instance of the specified database server will be addressed.

The square brackets at the right-hand side of the input field indicate, that it is a special input field, allowing packagers to use Properties and other variable packaging project information for the value definition. Please refer to the *MSI formatted string field* section for further details on how to handle this typical kind of user interface control in RayPack.

Database

This mandatory field has to contain the name of the target database itself. It may neither be empty, nor exceed a length of 255 characters.

Tip:
There are some official Microsoft recommendations regarding the restrictions database schema objects should follow. It may be handy to apply them to the database name at this point:

In SQL Server 2012, an object name can be up to 128 characters long. Non-quoted identifier names must follow these rules:

- The first character must be alphanumeric, an underscore (_), an at sign (@), or a number sign (#).
- Subsequent characters can include alphanumeric characters, an underscore (_), an at sign (@), a number sign (#), or a dollar sign ($).
- The identifier must not be a Transact-SQL reserved word.
- Embedded spaces or special characters are not allowed.

Identifiers that start with an at sign (@) or a number sign (#) have special meanings. Identifiers starting with @ are local variable names. Those that start with a # are temporary table names.

Excerpt derived from *SQL Server 2012 - Guide to Migrating from MySQL to SQL Server 2012*

User

Accessing databases usually requires credentials. These have to be given in the form of a RayPack user object. To select or create a user object, click on the browse button **[...]** at the right-hand side of the user input field. The **Select User** dialog is displayed. Please refer to the *Common dialogs* section for further details on how to *manage user data objects*.

However, the user property of an SQL database object is optional, which allows packagers to define connection settings wherever and however they prefer to.

Be aware:
User objects may very likely be used at several places within SQL database, script, and IIS management structures. When the same user item is referred to from different locations, changing the properties of that user item actually takes effect for all related objects. Please double-check the correctness of user profile changes, and make sure all related objects are still valid and operational with the modified set of properties.

Step 2: Attributes Info

Run-time behavior
This steps allows to determine the activities performed during the several run-times of a typical Windows Installer package:

- On install
- On uninstall

- On reinstall

It is possible to independently determine if the database object should be created, dropped, or not affected at all during a specific run-time type. A typical setting would be to create the database on install, execute no action on uninstall, and create on reinstall.

Continue on error
When this checkbox is activated, errors that occur during the activities defined above do not lead to a package run-time execution to fail. It is recommended to keep it deactivated (as it is per default), if a database is vital for the operability of other package resources.

Confirm updates of existing databases
If the database already exists on the target device, activating this checkbox demands user confirmation for any action that manipulates the existing database at package run-time.

If the checkbox is not active, manipulations of existing databases will be executed without explicit user confirmation. It is highly recommended to double-check the expected and actual run-time behavior of the target package in order to prevent data loss and negative user experience.

Step 3: File Specification Info

If the SQL database object is actually used to create a new database on the package target machines, it is possible to define properties for the database files and log files:

Create File Specification

If this option is active, the following file related settings become effective and will be considered during the SQL database creation phase.

Destination folder
The destination of the newly created database file has to be defined by selecting one of the already given directories within the packaging project, or by creating a new one. Both procedures may be done by clicking the browse button [...] at the right-hand side of the destination folder property control item, and using the interface of the *common dialog Select directory*.

File name
The name of the database file has to be given with the Microsoft SQL database standard extension `*.mdf`. A default name is derived from the value entered as database name, and the suffix "`_DB`". It is recommended to follow a definitive file name convention to prevent accidental removal or manipulation of vital resources.

Size
The initial size of the file. The entered value may be extended with the suffixes KB, MB, and GB. If no suffix is provided, the value will be considered to be given in MB. If this optional property is not defined, a default of 1MB initial size is actually applied to the file.

Growth size

When the initial file size does no longer suffice to manage the required amount of data, the growth size determines the steps for file enlargements:

- If a value is given without one of the possible suffixes KB, MB, GB, or %, the value is considered to be given in MB.
- If no value is given, the default growth step is 10%.
- A growth step may not be smaller than 64KB
- A growth step may not be larger than the max. file size (see below)

Max size

In order to prevent the database from overpopulating the parent device, it is recommended to define a maximal size for the database content. The entered value may be extended with the suffixes KB, MB, and GB. If no suffix is provided, the value will be considered to be given in MB. If this optional property is not defined, the file may grow until the available disk space is fully consumed.

Create Log File Specification

If this option is active, the following log file related settings become effective and will be considered during the SQL database creation phase.

Destination folder
See *above*.

File name
The name of the log file has to be given with the Microsoft SQL database standard extension `*.ldf`. A default name is derived from the value entered as database name, and the suffix "`_DBLog`". It is recommended to follow a definitive file name convention to prevent accidental removal or manipulation of vital resources.

Size
See *above*.

Growth size
See above.

Max size
See *above*.

Step 4: Summary

Use the summary page to check the correctness of the SQL database properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the SQL database item.
- If changes are due, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the

summary page again.

Step 5: Finished

Once the new SQL database control object has been created, the wizard can be closed by using the **Finish** button at its lower right corner. The SQL Databases view is updated, and the list of existing items contains the newly created object at the lowest position.

Remove an SQL Database

The PackDesigner mode Visual Designer contains an **SQL Database** view with a list of all objects stored within a packaging projects `RPSqlDatabase` table. To remove an SQL database by using this view interface, users have the following options at hand:

* Within the list of SQL databases, **right-click** any item, and select **Delete** from the **context menu**.

The remove procedure is triggered. Once an SQL database is removed, it is lost from the packaging project.

> **Be aware:**
> If user objects were created during the **Add SQL database** wizard execution, these additional packaging project contents are not automatically removed when the SQL database itself is deleted. Please make sure to remove them manually if required.

Confirm Object Removal

Since an accidental object removing could lead to substantial issues, RayPack displays a **Confirm** dialog before an essential project content object is actually removed. Use the **REMOVE** button within that dialog to confirm the activity. If **DO NOT REMOVE** or **CANCEL** are clicked, the object is not deleted and the user returns to the parent view without any changes.

Edit an SQL Database

1. To edit an SQL database object, users load the list of existing items by calling the **SQL Databases** view within the Visual Designer mode of PackDesigner.

2. Clicking on the list item of the object that has to be modified displays its details in the editor panel on the right-hand side of the RayPack application screen.

The following properties of the SQL database object may be manipulated:

Database Name

The database name is displayed above the three view tabs of the edit area. It becomes editable with a click on either the name itself, or the edit icon on the right-hand side. A *direct value editor dialog* is displayed, allowing to enter the new name value. By hitting **Enter** or clicking on the **save** icon within the direct value editor interface the new name is saved.

Tab: CONNECTION

Server
The physical address of the database server. This property is mandatory. It may either be defined by directly entering a string value of max. 255 characters length, or by using one of the variables available within the packaging project (e. g. a property). To trigger support for dynamic content input, users have to type an opening square bracket. The dialog for *MSI formatted string input fields* is displayed. Please refer to the *common dialogs* section for further details on how to handle this type of input control.

Instance
The instance name of the target SQL database object. If this optional value is not set, the default instance of the specified database server will be addressed.

The square brackets at the right-hand side of the input field indicate, that it is a special input field, allowing packagers to use Properties and other variable packaging project information for the value definition. Please refer to the *MSI formatted string field* section for further details on how to handle this typical kind of user interface control in RayPack.

Component
Each SQL database object has to be related to a component item of the packaging project. RayPack automatically creates a new component when an SQL database item is created.

However, it is possible to change the predefined component structure.

To select another existing component as parent object of the SQL database item, users click on the downwards-pointing arrow at the right-hand side of the component selector control. A list of existing components is displayed. The new setting is established with a click on any of the given components.

To create a new component for the SQL database object, users click on the browse button **[...]** at the right-hand side of the component control. The *common dialog* type **Select Component** is displayed, ready for creating and selecting components.

User
The credentials used to connect to the specified SQL database may be provided by denoting a user profile. To change the current user profile relation, packagers click on the browse button **[...]** at the right-hand side of the user property control object. The *common dialog* **Select User** is displayed, ready for creating, editing, removing and mere selecting of user profiles. Please refer to the section *User object manager* to get more details about the handling of this interface type.

> **Be aware:**
> User objects may very likely be used at several places within SQL database, script, and IIS management structures. When the same user item is referred to from different locations, changing the properties of that user item actually takes effect for all related objects. Please double-check the correctness of user profile changes, and make sure all related objects are still valid and operational with the modified set of properties.

Tab: ADVANCED

Run-time behavior
This steps allows to determine the activities performed during the several run-times of a typical Windows Installer package:

- On install
- On uninstall
- On reinstall

It is possible to independently determine if the database object should be created, dropped, or not affected at all during a specific run-time type. A typical setting would be to create the database on install, execute no action on uninstall, and create on reinstall.

Continue on error
When this checkbox is activated, errors that occur during the activities defined above do not lead to a package run-time execution to fail. It is recommended to keep it de-activated (as it is per default), if a database is vital for the operability of other package resources.

Confirm updates of existing databases
If the database already exists on the target device, activating this checkbox demands user confirmation for any action that manipulates the existing database at package run-time.

If the checkbox is not active, manipulations of existing databases will be executed without explicit user confirmation. It is highly recommended to double-check the expected and actual run-time behavior of the target package in order to prevent data loss and negative user experience.

Tab: FILE SPECIFICATIONS

If the SQL database object is actually used to create a new database on the package target machines, it is possible to define properties for the database files and log files:

Create File Specification

If this option is active, the following file related settings become effective and will be considered during the SQL database creation phase.

Destination folder

The destination of the newly created database file has to be defined by selecting one of the already given directories within the packaging project, or by creating a new one. Both procedures may be done by clicking the browse button [...] at the right-hand side of the destination folder property control item, and using the interface of the *common dialog* **Select directory**.

File name

The name of the database file has to be given with the Microsoft SQL database standard extension `*.mdf`. A default name is derived from the value entered as database name, and the suffix "`_DB`". It is recommended to follow a definitive file name convention to prevent accidental removal or manipulation of vital resources.

Size

The initial size of the file. The entered value may be extended with the suffixes KB, MB, and GB. If no suffix is provided, the value will be considered to be given in MB. If this optional property is not defined, a default of 1MB initial size is actually applied to the file.

Growth size

When the initial file size does no longer suffice to manage the required amount of data, the growth size determines the steps for file enlargements:

- If a value is given without one of the possible suffixes KB, MB, GB, or %, the value is considered to be given in MB.
- If no value is given, the default growth step is 10%.
- A growth step may not be smaller than 64KB
- A growth step may not be larger than the max. file size (see below)

Max size

In order to prevent the database from overpopulating the parent device, it is recommended to define a maximal size for the database content. The entered value may be extended with the suffixes KB, MB, and GB. If no suffix is provided, the value will be considered to be given in MB. If this optional property is not defined, the file may grow until the available disk space is fully consumed.

Create Log File Specification

If this option is active, the following log file related settings become effective and will be considered during the SQL database creation phase.

Destination folder
See *above*.

File name
The name of the log file has to be given with the Microsoft SQL database standard extension `*.ldf`. A default name is derived from the value entered as database name, and the suffix "`_DBLog`". It is recommended to follow a definitive file name convention to prevent accidental removal or manipulation of vital resources.

Size
See *above*.

Growth size
See *above*.

Max size
See *above*.

SQL Scripts

This view allows you to create SQL script which allow to manipulate SQL databases on the target system. The provided functionality is designed to support SQL database management only. Other database types have to be manipulated by individual custom action measures.

> **Note:**
> Another view of the Visual Designer, SQL databases, provides the management functions to establish connections to SQL database objects which may be manipulated by the SQL script object handled within this view. Identifying the target database for SQL script execution is only possible, if a database object itself has been provided with connection information before.

Since database handling is not part of the MSI Standard, RayPack utilized special Installer database tables to manage SQL script information until a package is built from the project:

- `RPSqlScript` - Contains basic data about SQL scripts
- `RPSqlReplace` - Contains information about search and replace procedures for SQL scripts

If these tables have not been present within the current packaging project before, they are automatically added with the first SQL script object created within the project. Please refer to the section about *Custom RayPack installer database tables* for further details and table specifications.

The SQL Scripts view is separated into two main areas:
- A list of already available SQL script objects on the left-hand side
  This list is grouped by the SQL databases the scripts operate on. With a click on the arrow icon in front of the database group header it is possible to expand and collapse the script list group.
- A details pane for viewing and editing the properties of already existing SQL script objects at the right-hand side

To access the Installer database counterpart of an SQL script item, users have to right-click its list object, and select **Go to row** from the context menu. The database table `RPSqlScript` is loaded, with a highlight set on the matching data row.

Standard SQL Script Management Procedures

Common management procedures for SQL script objects are described within the following topics:

- *Add an SQL script*
- *Remove an SQL script*
- *Edit an SQL script*

Add an SQL Script

> **Note:**
> Please make sure to use the **SQL Databases** view for database object creation before the wizard for SQL script creation is used. Each SQL script needs to operate on a specific database connection, which is established by the definition of an SQL database object.

It is not possible to create SQL database objects directly via the Create SQL Script wizard interface.

In order to add an SQL script to a packaging project, users go to the **SQL Scripts** view of the Visual Designer mode.

With a click on the **Create new...** button, the options menu is displayed. Click on **SQL database** to invoke the New SQL Database Wizard.

Work your way through the steps of the wizard to define all required properties for the new item.

At any time, using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, allows to navigate within the already processed steps.

To exit the wizard without creating a new object, use the Cancel button, also located at the bottom of the wizard dialog.

Step 1: Database

Database

Please select the database object the new SQL script has to operate on. The database object has to be provided as row within the `RPSqlDatabase` table, since this table is read to provide the list of databases users can select from within this wizard step.

Once the database object is selected from the list, the matching server address is automatically displayed as additional informative reference.

User

RayPack allows to run SQL scripts either via Windows Authentication, or as a specific user.

The default setting is Windows Authentication, indicated by the active checkbox Use Windows Authentication. To enable the usage of individual user credentials, the checkbox needs to be deselected. Once it is inactive, the controls for user object management become active for user selection.

Individual user credentials have to be given in the form of a RayPack user object. To select or create a user object, click on the browse button **[...]** at the right-hand side of the user input field. The **Select User** dialog is displayed. Please refer to the *Common dialogs* section for further details on how to *manage user data objects*.

> **Be aware:**
> User objects may very likely be used at several places within SQL database, script, and IIS management structures. When the same user item is referred to from different locations, changing the properties of that user item actually takes effect for all related objects. Please double-check the correctness of user profile changes, and make sure all related objects are still valid and operational with the modified set of properties.

Step 2: Advanced

Name

A default name for the new script object is automatically generated once this wizard step is displayed. It is combined by the Prefix `RPSqlScript` and an automatically calculated and

incremented index value. The script name has to be defined as unique alphanumerical string of max. 72 characters length.

Even though blanks and special characters (e. g. question mark (?) or period (.)) are technically allowed, it is recommended to stick to alphanumerical script names.

Component
Each SQL script object has to be related to a component item of the packaging project. RayPack automatically creates a new component when an SQL database item is created. It is recommended to add SQL scripts to the same component as the parent SQL database object, which is why the component of the SQL database selected during step 1 is pre-selected for the new SQL script.

To select another existing component as parent object of the SQL script item, users click on the downwards-pointing arrow at the right-hand side of the component selector control. A list of existing components is displayed. The new selection is done with a click on any of the given components.

To create a new component for the SQL script object, users click on the browse button **[...]** at the right-hand side of the component control. The *common dialog* type **Select Component** is displayed, ready for creating and selecting components.

Sequence
The sequence defines in which specific order SQL scripts are executed on the target database. The first SQL script defined for a database is always executed first, whilst additional scripts may be assigned to be run either at the beginning of the SQL sequence or after any of the already existing SQL scripts.

Please be aware that the sequence does not consider scripts that are run on other database objects within the packaging project. Adjust the order of database script groups at the SQL scripts list view to manage the sequence of database manipulations at that higher level.

Attributes
The attributes column of the `RPSqlScript` table is equipped to provide the following script execution options:

- Execute on install
  The script will be executed when the package is installed.

- Execute on uninstall
  The script will be executed when the package is uninstalled.

- Continue on error
  If the script execution causes an error, the package run-time execution is not aborted, no matter if it is an installation, repair or uninstallation procedure.

- Rollback on install
  The rollback mechanism for this SQL script is executed when the package is installed.

- Rollback on uninstall
  The rollback mechanism for this SQL script is executed when the package is installed.

Users may activate each execution option by selecting the checkbox at the left-hand side of the option label.

> **Be aware:**
> If neither "Execute on install" nor "Execute on uninstall" are selected, the script may very well never run on the target database during package run-time.

Step 3: Script

The actual content of the SQL script has to be defined during this wizard step. There are three different options for the source type of the script:

Binary Table
The SQL statements are already present as content of a binary stream object within the current packaging project.

Once this option is chosen from the Source Type selector, another selector "Binary" is displayed, waiting for the user to pick the desired binary file from the list of available binaries.
As soon as a binary has been selected, the content of that binary file is displayed within the Script text area below.

> **Be aware:**
> Changing the content of the SQL script text area once a binary object has been selected will cause an update of the content stored within the binary resource itself. The SQL script shown here is not a copy that will be created within the SQL script object, but the actual binary content. Please make sure to double-check the binary usage at other places before the content is changed within this view.

Property Table
The SQL statements are already present as content of a property within the current packaging project.

Once this option is chosen from the Source Type selector, another selector "Property" is displayed, waiting for the user to pick the desired property from the list of available Property table rows.
As soon as a property has been selected, the value of that property is displayed within the Script text area below.

> **Be aware:**
> Changing the content of the SQL script text area once a binary object has been selected will cause an update of the content stored within the property itself. The SQL script shown here is not a copy that will be created within the SQL script object, but the actual property content. Please make sure to double-check the property usage at other

places before the content is changed within this view.

Text

This source type option stores the script as text within the `RPSqlScript` table column `Target`. Even if the same SQL string is used several times within the packaging project as text source for SQL script objects, there will be a separate source copy for each script object.

> **Be aware:**
> The source type of the SQL script may not be changed via the Visual Designer user interface at a later time. If Property has been selected and the SQL script object has been saved, the source type setting will always be Property. However, it is possible to edit which property the script content is stored within. (Or to change the binary that is read for the Binary source type option).
>
> It is possible to directly manipulate the `Source_` column of the `RPSqlScript` Installer database table via the TABLES view of the Advanced mode, but it is not recommended to do so for packagers with a beginner level experience.

Load from file...

No matter which source type has been selected, it is always possible to replace the current content of the SQL script input field with a string retrieved from any file within reach. Users simply click on the Load from file button, navigate to the desired `*.sql` file, and click Open. The file content immediately replaces the existing SQL script content string. Please keep in mind, that this function reads the file content and writes the string into one of the source types references named above (a binary, a property, the Target column of the SQL script database row). However, the file itself is not saved as physical resource within the packaging project. This means it will not be available for direct access, e. g. via the Files & Folders view.

Switch off '[', ']' and Switch on '[', ']'

Square brackets may either be considered to be triggers for special SQL script notations, such as delimited identifiers, or as simple text signs without special functionality. In order to support packagers in their aim for clean and unambiguous source definitions, the switch for square brackets has been added to the SQL script text area.

Switching square brackets off leads to the explicit escaping of square bracket, which means that each opening and closing square bracket is prefixed with escape triggers. The escaped bracket signs are no longer considered to be special notations, but as a mere opening / closing bracket signs without additional functionality.

Example:
The original text "`This is [sample] text`" turns into "`This is [\[]sample[\]] text`" once square brackets are turned off.

Switching square brackets on removes escaping from already existing escaped opening / closing bracket signs. They are considered to be special characters (SQL delimited identifiers) again.

Example:

The original text "`This is [sample] text with different [\[]square[\]] bracket usages.`" turns into "`This is [sample] text with different [square] bracket usages.`" once square brackets are turned on.

Step 4: Replacement

If search & replace activities on SQL scripts are required for execution at package run-time, this is the interface to create them. With a click on the **Add** button, a new replacement task is generated with default values. To change any of these values, users have to double-click the cell value of the list item they want to modify. The current value is marked and ready for direct inline modification:

Sequence
The sequence value is an automatically incremented integer value, indicating the order of replacement execution. The item with the lowest sequence value is run first.

Search
The search string has to be given as plain text. There is no support for wild card usage or regular expression based search strings.

Please keep in mind, that the standard search algorithm is case insensitive. To change that, the attributes value 2 has to be set for the replacement job.

Replace
The string that will be put instead of the matched keyword from Search.

Please keep in mind, that the standard search algorithm replaces all matches. To stop the replacement after the first match, the attributes value 4 has to be set for the replacement job.

Attributes
Users may either enter any combination of the available attribute bit value representations manually, or use the helping hand of the Attributes Editor for this property. To display the Attributes Editor, users have to double-click the current value of the attributes column, and use the downwards pointing arrow button. Checkboxes become visible, allowing to individually enable or disable each attribute option:

1 = Match whole word only
2 = Match case
4 = Replace once only

The default attribute value for new replacement tasks is 1, which means that the search is resolved word by word.

To **remove a replacement task**, users have to select it from the list of existing replacements, and use the **Remove** button below the list. The task is removed immediately, without an intermediate confirm dialog.

Step 5: Summary

Use the summary page to check the correctness of the SQL script properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the SQL script item.
- If changes are due, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the summary page again.

Step 6: Finished

Once the new SQL script object has been created, the wizard can be closed by using the **Finish** button at its lower right corner. The SQL Scripts view is updated, and the list of existing items contains the newly created object at the lowest position within the group of scripts saved for the same target database.

Remove an SQL Script

The PackDesigner mode Visual Designer contains an SQL Scripts view with a list of all objects stored within a packaging projects `RPSqlScript` table. To remove an SQL script by using this view interface, users have the following options at hand:

- Within the list of SQL scripts, **right-click** any item, and select **Delete** from the **context menu**.

The remove procedure is triggered. Once an SQL script is removed, it is lost from the packaging project.

Be aware:
Once the Remove option is selected from the context menu, there is no intermediate confirm dialog - SQL script objects are deleted immediately.
Also be aware:
If user objects were created during the **Add SQL script** wizard execution, these additional packaging project contents are not automatically removed when the SQL script itself is deleted. Please make sure to remove them manually if required.

Edit an SQL Script

1. To edit an SQL script object, users load the list of existing items by calling the **SQL Scripts** view within the Visual Designer mode of PackDesigner.

2. Clicking on the list item of the object that has to be modified displays its details in the editor panel on the right-hand side of the RayPack application screen.

The following properties of the SQL script object may be manipulated:

Script Name

The script name is displayed above the three view tabs of the edit area. It becomes editable with a click on either the name itself, or the edit icon on the right-hand side. A direct value editor dialog is displayed, allowing to enter the new name value. By hitting **Enter** or clicking on the **save** icon within the direct value editor interface the new name is saved.

Tab: GENERAL

Database
Select one of the SQL database objects that have been created for the current packaging project. If the required option is not available yet, the Create new SQL Database wizard has to be used to add one to the project. This wizard is available from the SQL Databases view.

User
RayPack allows to run SQL scripts either via Windows Authentication, or as a specific user. If the checkbox Use Windows Authentication is active, this option is used for the current SQL script.

To enable the usage of individual user credentials, the checkbox needs to be deselected. Once it is inactive, the controls for user object management become active for user selection.

Individual user credentials have to be given in the form of a user object. To select or create a user object, click on the browse button **[...]** at the right-hand side of the user input field. The **Select User** dialog is displayed. Please refer to the Common dialogs section for further details on how to manage user data objects.

> **Be aware:**
> User objects may very likely be used at several places within SQL database, script, and IIS management structures. When the same user item is referred to from different locations, changing the properties of that user item actually takes effect for all related objects. Please double-check the correctness of user profile changes, and make sure all related objects are still valid and operational with the modified set of properties.

Component
Each SQL script object has to be related to a component item of the packaging project. RayPack automatically creates a new component when an SQL database item is created. It is recommended to add SQL scripts to the same component as the parent SQL database object.

To select another existing component as parent container of the SQL script item, users click on the downwards-pointing arrow at the right-hand side of the component selector control. A list of existing components is displayed. The new selection is done with a click on any of the given components.

To create a new component for the SQL script object, users click on the browse button **[...]** at the right-hand side of the component control. The common dialog type Select Component is displayed, ready for creating and selecting components.

Sequence

The integer value is displayed in a read-only mode. The Visual Designer interface does not allow to modify the sequence value directly, but indirectly by the re-ordering options available from the context menu displayed when SQL script list items are right-clicked. (Execute first and Execute earlier reduce the sequence value, whilst Execute later and Execute last raise it.)

> **Tip:**
> Any value that cannot be manipulated by dedicated Visual Designer user interface controls is nonetheless available for modification by the tools provided within the TABLES view of the Advanced mode. Even though there is a clear recommendation to leave the Advanced mode for experienced users, it is freely accessible for all RayPack users.

Attributes

The attributes column of the `RPSqlScript` table is equipped to provide the following script execution options:

- Execute on install
  The script will be executed when the package is installed.

- Execute on uninstall
  The script will be executed when the package is uninstalled.

- Continue on error
  If the script execution causes an error, the package run-time execution is not aborted, no matter if it is an installation, repair or uninstallation procedure.

- Rollback on install
  The rollback mechanism for this SQL script is executed when the package is installed.

- Rollback on uninstall
  The rollback mechanism for this SQL script is executed when the package is installed.

Users may activate each execution option by selecting the checkbox at the left-hand side of the option label.

> **Be aware:**
> If neither "Execute on install" nor "Execute on uninstall" are selected, the script may very well never run on the target database during package run-time.

Tab: SQL SCRIPT

The interface options for the SQL SCRIPT tab depend on the script source type that has been defined during SQL script object creation. However, if the displayed script has to be modified, users have to click on the Edit the SQL script... button above the actual script content display area. A new dialog window is opened, ready for manipulation according to the source type:

Binary

Users may either

- select another binary as source for the script string
- manually edit the content of the currently selected binary by simply editing the string shown in the SQL script text area
- replace the binary's content by loading new textual content from an external `*.sql` file

Property

Users may either

- select another property as source for the script string
- manually edit the content of the currently selected property by simply editing the string shown in the SQL script text area
- replace the properties content by loading new textual content from an external `*.sql` file

Text

Users may either

- manually edit the content of the `Target` column of the currently active `RPSqlScript` database table row reference by simply editing the string shown in the SQL script text area
- replace the `Target` column content of the very same `RPSqlScript` database table row reference by loading new textual content from an external `*.sql` file

Switch off '[', ']' and Switch on '[', ']'

Square brackets may either be considered to be triggers for special SQL script notations, such as delimited identifiers, or as simple text signs without special functionality. In order to support packagers in their aim for clean and unambiguous source definitions, the switch for square brackets has been added to the SQL script text area.

Switching square brackets off leads to the explicit escaping of square bracket, which means that each opening and closing square bracket is prefixed with escape triggers. The escaped bracket signs are no longer considered to be special notations, but as a mere opening / closing bracket signs without additional functionality.

Example:
The original text "`This is [sample] text`" turns into "`This is [\[]sample[\]] text`" once square brackets are turned off.

Switching square brackets on removes escaping from already existing escaped opening / closing bracket signs. They are considered to be special characters (SQL delimited identifiers) again.

Example:
The original text "`This is [sample] text with different [\[]square[\]] bracket usages.`" turns into "`This is [sample] text with different [square] bracket usages.`" once square brackets are turned on.

Changes to the SQL script content are saved with a click on the **OK** button at the bottom of the editor. The OK button is not available as long as the SQL script input field is empty, since the script content is a mandatory information. Hitting Cancel discards any unsaved changes and closes the dialog window.

Be aware:
Saving changes to SQL script content that comes from a binary or property actually modifies the content of that specific packaging project item. If other functions or options rely on the very same binary or object, they will have to interact with the modified content as well.

Tab: REPLACEMENT

If search & replace activities on SQL scripts are required for execution at package run-time, this is the tab to manage them. With a click on the **Add** button, a new replacement task is generated with default values. It is automatically set to the end of the current replacement task sequence (which means at the lowest position of the task item list).

To change a replacement task property value, users have to double-click the cell value of the list item they want to modify. The current value is marked and ready for direct inline modification:

Sequence
The sequence value is an integer value, indicating the order of replacement execution. The item with the lowest sequence value is run first.

Search

The search string has to be given as plain text. There is no support for wild card usage or regular expression based search strings.

Please keep in mind, that the standard search algorithm is case insensitive. To change that, the attributes value 2 has to be set for the replacement job.

Replace

The string that will be put instead of the matched keyword from Search.

Please keep in mind, that the standard search algorithm replaces all matches. To stop the replacement after the first match, the attributes value 4 has to be set for the replacement job.

Attributes

Users may either enter any combination of the available attribute bit value representations manually, or use the helping hand of the Attributes Editor for this property. To display the Attributes Editor, users have to double-click the current value of the attributes column, and use the downwards pointing arrow button. Checkboxes become visible, allowing to individually enable or disable each attribute option:

1 = Match whole word only
2 = Match case
4 = Replace once only

The default attribute value for a replacement task is 1, which means that the search is resolved word by word.

To **remove a replacement task**, users have to select it from the list of existing replacements, and use the **Remove** button below the list. The task is removed immediately, without an intermediate confirm dialog.

IIS

This view allows you to manipulate IIS web sites, as well as configuration and options of IIS that will be installed on the target system.

Since IIS handling is not part of the MSI Standard, RayPack utilized special Installer database tables to manage IIS related information until a package is built from the project:

- `RPIIsAppPool` - Application pool property definitions
- `RPIIsWebAddress` - Relation between web sites and their addresses on the IIS
- `RPIIsWebApplication` - Registered web applications
- `RPIIsWebApplicationExtension` - Mapping between web applications and extensions
- `RPIIsWebDirProperties` - Directories on the IIS
- `RPIIsWebLog` - References of available log file formats
- `RPIIsWebServiceExtension` - Mapping between web services and extensions
- `RPIIsWebSite` - Websites on the IIS
- `RPIIsWebVirtualDir` - Virtual directories for websites on the IIS

If these tables have not been present within the current packaging project before, they are automatically added with the first requirement by an ISS object created within the project. Please refer to the section about *Custom RayPack installer database tables* for further details and table specifications.

The IIS view is separated into two main areas:

- A tree view structure of already available IIS objects on the left-hand side
  This hierarchy is structured by the available main IIS object types: Application pools, Web service extensions and Web sites.

- A details pane for viewing and editing the properties of already existing IIS objects at the right-hand side
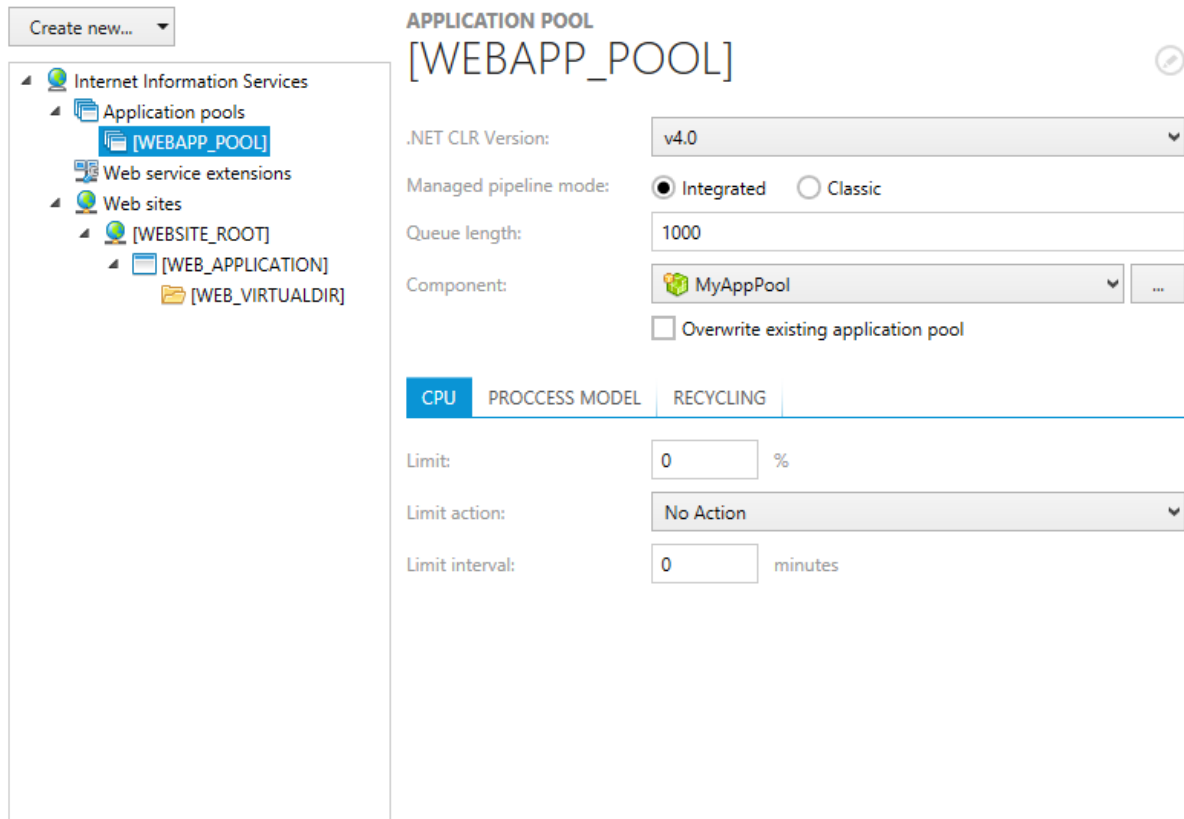
To access the Installer database counterpart of an IIS item, users have to right-click its list object, and select **Go to row** from the context menu. The required database table is loaded, with a highlight set on the matching data row.

Standard IIS Management Procedures

Common management procedures for IIS objects are described within the following topics:

- *Add an IIS application pool*
- *Add an IIS web service extension*
- *Add an IIS web site*
- *Remove an IIS object*
- *Edit an IIS application pool*
- *Edit an IIS web service extension*
- *Edit an IIS web site*

Add an IIS Application Pool

In order to add an IIS application pool to a packaging project, users go to the IIS view of the Visual Designer mode.

- With a click on the **Create new...** button, the options menu is displayed. Click on **Application pool** to invoke the New Application Pool Wizard.

- As an alternative access to the very same wizard, users may click on the **Application pools node** within the IIS object tree structure on the left hand side of the IIS view, and click on the **Add new application pool** button that is displayed within the details pane at the right-hand side.

Work your way through the steps of the wizard to define all required properties for the new item.

At any time, using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, allows to navigate within the already processed steps.
To exit the wizard without creating a new object, use the Cancel button, also located at the bottom of the wizard dialog.

Step 1: General

.NET CLR version
This property specifies the .NET Framework version to be used by the application pool. Please select the required version from the predefined set of available options.

Managed Pipeline mode
Please select one of the predefined settings:

- Classic
  Is the traditional mode as known from IIS6 and earlier. Run this way, IIS operates directly on ISAPI extensions and ISAPI filters.

- Integrated
  was introduced as new mode in IIS7. The IIS pipeline is equal to the ASP.NET request pipeline.

Whilst the classic mode treats ASP.NET as external plugin, without knowledge about and access to internal processes, the integrated mode has ASP.NET fully integrated into IIS.

Name
The name of the app pool as it will be displayed within the IIS application pools overview. It is an alphanumerical string of max. 72 characters length. It is recommended to make sure each application pool has a unique and informative name, as this helps to support object management for the created IIS structure on the target machine.

Queue length
This value defines the maximum number of requests that Http.sys queues for the application pool. When the queue is full, new requests receive a 503 "Service Unavailable" response. The default of 1000 is already given as a recommendation when the wizard is invoked.

Component

Each application pool object has to be related to a component item of the packaging project.

To select an existing component as parent object of the app pool item, users click on the downwards-pointing arrow at the right-hand side of the component selector control. A list of existing components is displayed. The actual selection is done with a click on any of the given components.

To create a new component for the app pool object, users click on the browse button **[...]** at the right-hand side of the component control. The *common dialog* type **Select Component** is displayed, ready for creating and selecting components.

Overwrite existing application pool

This checkbox triggers a check on the target machine IIS, searching for application pools with the same name. If such an object exists, it is either fully overwritten with the properties defined for the application pool contained in the package (when the checkbox is active), or left totally unchanged (when the checkbox is inactive).

Step 2: CPU

Limit

Configures the maximum percentage of CPU time that the worker processes in an application pool are allowed to consume over a period of time as indicated by the Limit Interval setting. Setting this value to 0 disables limiting the worker processes to a percentage of CPU time.

Limit action

Please select the action that has to be taken when the limit is reached:

- **NoAction** - an event log entry is generated.
- **KillW3WP** - the application pool is shut down for the duration of the reset interval and an event log entry is generated.
- **Throttle** - the CPU consumption is limited to the value set in Limit. The Limit interval is not used and an event log entry is generated.
- **ThrottleUnderLoad** - the CPU consumption is limited only when there is contention on the CPU. The Limit interval is not used and an event log entry is generated.

Limit interval

Specifies the reset period for CPU monitoring and throttling limits on the application pool. When the number of minutes elapsed since the last process accounting reset equals the number specified by this property, IIS resets the CPU timers for both the logging and limit intervals. Setting this value to 0 disables CPU monitoring.

Step 3: Process Model

Identity

Configures the application pool to run as a built-in account, such as

- Network Service (recommended)
- Local Service

- Local System
- ApplicationPoolIdentity
- As a specific user identity (Other)

User

This set of control becomes active as soon the the Identity property is set to **Other**.

Accessing system parts in the name of an explicit user identity usually requires credentials. These have to be given in the form of a user object. To select or create a user object, click on the browse button **[...]** at the right-hand side of the user input field. The **Select User** dialog is displayed. Please refer to the *Common dialogs* section for further details on how to *manage user data objects*.

> **Be aware:**
> User objects may very likely be used at several places within SQL database, script, and IIS management structures. When the same user item is referred to from different locations, changing the properties of that user item actually takes effect for all related objects. Please double-check the correctness of user profile changes, and make sure all related objects are still valid and operational with the modified set of properties.

Idle timeout

Amount of time a worker process remains idle before it shuts down. A worker process is idle if it is not processing requests and no new requests are received. The default of 20 minutes is already given as a recommendation when the wizard step is displayed.

Maximum worker processes

The maximum number of worker processes permitted to service requests for the application pool. If this number is greater than the recommended default value of 1, the application pool is called a "web garden".

Step 4: Recycling

Private memory limit

The maximum amount of private memory a worker process can consume before causing the application pool to recycle. A value of 0 means there is no limit.

Regular time interval

The period of time after which an application pool recycles. A value of 0 means the application pool does not recycle at a regular interval.

Request limit

The maximum number of requests an application pool can process before it is recycled. A value of 0 means the application pool can process an unlimited number of requests.

Specific times

A set of specific local times, in 24 hour format, when the application pool is recycled.

Virtual memory limit

The maximum amount of virtual memory a worker process can consume before causing the application pool to recycle. A value of 0 means there is no limit.

Step 5: Summary

Use the summary page to check the correctness of the application pool properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the application pool item.
- If changes are due, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the summary page again.

Step 6: Finished

Once the new application pool object has been created, the wizard can be closed by using the **Finish** button at its lower right corner. The IIS view is updated, and the tree structure of existing items contains the newly created object.

Add an IIS Web Service Extension

In order to add an IIS web service extension to a packaging project, users go to the IIS view of the Visual Designer mode.

- With a click on the **Create new...** button, the options menu is displayed. Click on **Web service extension** to invoke the New Web Service Extension Wizard.
- As an alternative access to the very same wizard, users may click on the **Web service extensions node** within the IIS object tree structure on the left hand side of the IIS view, and click on the **Add new web service extension** button that is displayed within the details pane at the right-hand side.

Work your way through the steps of the wizard to define all required properties for the new item.

At any time, using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, allows to navigate within the already processed steps.
To exit the wizard without creating a new object, use the Cancel button, also located at the bottom of the wizard dialog.

Step 1: Description

Description
The name of the new web service extension file has to be given as an identifier string: it has to be a unique, not empty alphanumeric string of max. 72 characters length. It may contain the ASCII characters A-Z (a-z), digits, underscores (_), or periods (.), and must begin with either a letter or an underscore.

Step 2: Component

Component

Each web service extension object has to be related to a component item of the packaging project.

To select an existing component as parent object of the web service extension item, users click on the downwards-pointing arrow at the right-hand side of the component selector control. A list of existing components is displayed. The actual selection is done with a click on any of the given components.

To create a new component for the web service extension object, users click on the browse button **[...]** at the right-hand side of the component control. The *common dialog* type **Select Component** is displayed, ready for creating and selecting components.

Step 3: File

File

Please provide the full path to the extension file. Usually a property is provided, resolving to the short file name path.

The square brackets at the right-hand side of the input field indicate, that it is a special input field, allowing packagers to use **Properties** and other variable packaging project information for the value definition. Please refer to the *MSI formatted string field* section for further details on how to handle this typical kind of user interface control in RayPack.

Group

This string of max. 255 characters length is used to identify groups of extensions. IIS offers specific management functions which are executable on groups rather than single object, which makes it easier to maintain a clear and stable overal system state.

Attributes

- Allow - activate the checkbox to set the web service extension to Allow
- Deletable - activate the checkbox in order to allow that the web service extension file can be deleted from the IIS via the IIS Manager interface

Step 4: Summary

Use the summary page to check the correctness of the web service extension properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the new item.
- If changes are due, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the summary page again.

Step 5: Finished

Once the new web service extension object has been created, the wizard can be closed by using the **Finish** button at its lower right corner. The IIS view is updated, and the tree structure of existing items contains the newly created object.

Add an IIS Web Site

In order to add an IIS web site to a packaging project, users go to the IIS view of the Visual Designer mode.

- With a click on the **Create new...** button, the options menu is displayed. Click on **Web site** to invoke the New Web Site Wizard.
- As an alternative access to the very same wizard, users may click on the **Web sites node** within the IIS object tree structure on the left hand side of the IIS view, and click on the **Add new website** button that is displayed within the details pane at the right-hand side.

Work your way through the steps of the wizard to define all required properties for the new item.

At any time, using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, allows to navigate within the already processed steps.
To exit the wizard without creating a new object, use the Cancel button, also located at the bottom of the wizard dialog.

Step 1: Site Name

Site name
The name of the new web size has to be given as an identifier string: it has to be a unique, not empty alphanumeric string of max. 72 characters length. It may contain the ASCII characters A-Z (a-z), digits, underscores (_), or periods (.), and must begin with either a letter or an underscore.

Step 2: General

Component
Each web site object has to be related to a component item of the packaging project.

To select an existing component as parent object of the web site, users click on the downwards-

pointing arrow at the right-hand side of the component selector control. A list of existing components is displayed. The actual selection is done with a click on any of the given components.

To create a new component for the web site, users click on the browse button **[...]** at the right-hand side of the component control. The *common dialog* type **Select Component** is displayed, ready for creating and selecting components.

State
- Auto start
- Start on install

Connection timeout
Specifies the time (in seconds) that IIS waits before it disconnects a connection that is considered inactive.

Step 3: Basic Settings

Physical path
The destination of the content provision for the new website has to be defined by selecting one of the already given directories within the packaging project, or by creating a new one. Both procedures may be done by clicking the browse button [...] at the right-hand side of the Physical path property control item, and using the interface of the *common dialog* **Select directory**.

Step 4: Binding

Type
Please select one of the predefined types:
- HTTP - HyperText Transfer Protocol as stateless communication protocol for web contents
- HTTPS - HyperText Transfer Protocol Secure as communication protocol for web contents with special, certificate-based security functions (keyword SSL / TSL)

IP
The IP address for direct access to the web site. If the wild cart sign asterisk (*) is set, all requests to the web server will be directed to the web site.

Port
The port address as entrance filter for allowed incoming requests. If the wild cart value 80 is set, all ports are open for communication.

Header
The URL used as public alias for the web site content is another filter element for incoming communication requests.

> **Tip:**
> The combination of IP, port, and header is a quite powerful security and communication tunneling mechanism. Please refer to further chapters of the TechNet online contents

for further details.

Step 5: Advanced Settings

Web Application
Select one of the already available web applications stored within the current packaging project by clicking on the downwards pointing arrow at the right-hand side of the Web Application selector control. If the required application is not present within the list, use the browse button [...] on the right to open the **Web Application Manager** dialog. Please refer to the dedicated help topic for details on how to accomplish the required steps.

Web Dir Properties
Select one of the already available web directories stored within the current packaging project by clicking on the downwards pointing arrow at the right-hand side of the Web Dir selector control. If the required option is not present within the list, use the browse button [...] on the right to open the **Web Directories Manager** dialog. Please refer to the dedicated help topic for details on how to accomplish the required steps.

Web Log
Please select one of the predefined log options:
- IIS - fixed ASCII format
- NCSA - fixed ASCII format, specialized for web sites
- NONE - log files will not be written
- ODBC - record of a fixed set of data properties in a database
- W3C - customizable ASCII format

For further details regarding IIS log file formats, please visit MSDN.

Step 6: Summary

Use the summary page to check the correctness of the application pool properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the application pool item.
- If changes are due, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the summary page again.

Step 7: Finished

Once the new application pool object has been created, the wizard can be closed by using the **Finish** button at its lower right corner. The IIS view is updated, and the tree structure of existing items contains the newly created object.

Importing an IIS application

In order to add an IIS application pool to a packaging project, users go to the IIS view of the Visual Designer mode.

- With a click on the **Create new...** button, the options menu is displayed. Click on **Import IIS** to invoke the import dialog
- Select an `.rpiis` file containing the definition of the imported IIS website
- After clicking on **OK** button, RayPack will import all the necessary entries to recreate the imported website

The `.rpiis` file can be created by using the standalone IIS Scanner tool.

Remove an IIS Object

The PackDesigner mode Visual Designer contains an IIS view with a tree structure view on all IIS related objects stored within a packaging project. To remove an item by using this view interface, users have the following options at hand:

- Within the tree of IIS object, **right-click** any item, and select **Delete** from the **context menu**.

The remove procedure is triggered. Once an IIS item is removed, it is lost from the packaging project.

	Be aware:
☞	If an object has child nodes when it is marked for removal, those child elements are deleted along with their parent.

Confirm Object Removal

Since an accidental object removing could lead to substantial issues, RayPack displays a **Confirm** dialog before an IIS object is actually removed. Use the **REMOVE** button within that dialog to confirm the activity. If **DO NOT REMOVE** or **CANCEL** are clicked, the object is not deleted and the user returns to the IIS view without any changes.

The display of the Confirm dialog may manually be abandoned for future remove procedures by activating the "In future do not show this confirmation for delete operations" checkbox and using the REMOVE or DO NOT REMOVE button next. (Using the CANCEL button does not save any changes made to the status of that checkbox.) However, it is not recommended to skip the confirmation step, since it may prevent users from severe data loss.

Edit an IIS Application Pool

1. To edit an IIS application pool object, users load the tree structure of existing IIS items by calling the IIS view within the Visual Designer mode of PackDesigner.

2. The existing application pool objects may be brought to expanded display with a click on the arrow icon left of the "Application pools" node within the IIS object tree structure.

3. Clicking on the leaf representation of the app pool object that has to be modified displays its details in the editor panel on the right-hand side of the RayPack application screen.

The following properties of the application pool object may be manipulated:

General Properties

App Pool Name

The application pool name is displayed above the three view tabs of the edit area. It becomes editable with a click on either the name itself, or the edit icon on the right-hand side. A *direct value editor dialog* is displayed, allowing to enter the new name value. By hitting **Enter** or clicking on the **save** icon within the direct value editor interface the new name is saved.

Please refer to the section about *general application pool properties* within the help topic *Add an IIS application pool* for further details about the available general application pool settings.

Tab: CPU

Please refer to the section about *CPU settings* within the help topic *Add an IIS application pool* for further details.

Tab: PROCESS MODEL

Please refer to the section about *process model settings* within the help topic *Add an IIS application pool* for further details.

Tab: RECYCLING

Please refer to the section about *recycling settings* within the help topic *Add an IIS application pool* for further details.

Edit an IIS Web Service Extension

1. To edit an IIS web service extension object, users load the tree structure of existing IIS items by calling the IIS view within the Visual Designer mode of PackDesigner.

2. The existing web service extension objects may be brought to expanded display with a click on the arrow icon left of the "Web service extensions" node within the IIS object tree structure.

3. Clicking on the leaf representation of the web service extension object that has to be modified displays its details in the editor panel on the right-hand side of the RayPack application screen.

The following properties of the web service extension object may be manipulated:

Extension Description

The extension description becomes editable with a click on either the text itself, or the edit icon on the right-hand side. A *direct value editor dialog* is displayed, allowing to enter the new description value. By hitting **Enter** or clicking on the **save** icon within the direct value editor interface the new content is saved.

Please refer to the help topic *Add an IIS web service extension* for further details about the available object settings and how they have to be handled.

Edit an IIS Web Site

1. To edit an IIS web site object, users load the tree structure of existing IIS items by calling the IIS view within the Visual Designer mode of PackDesigner.

2. The existing web site objects may be brought to expanded display with a click on the arrow icon left of the "Web site" node within the IIS object tree structure.

3. Clicking on the leaf representation of the web site object that has to be modified displays its details in the editor panel on the right-hand side of the RayPack application screen.

The following properties of the web site object may be manipulated:

Site Name

The extension description becomes editable with a click on either the text itself, or the edit icon on the right-hand side. A *direct value editor dialog* is displayed, allowing to enter the new description value. By hitting **Enter** or clicking on the **save** icon within the direct value editor interface the new content is saved.

General Web Site Properties

Please refer to the help topic *Add an IIS web site* for further details about the available object settings and how they have to be handled.

Beyond the direct web site property settings themselves, there are further actions users may take on web sites:

- *Add an application to a web site* and *Add a virtual directory to a web site*

- *Remove an application or virtual directory from a web site*

- Edit a web site application & Edit a web site virtual directory
  To edit a web site application or virtual directory, users have to navigate through the child nodes of a web site, and select the object they want to edit with a left-click on the tree item. The details pane on the right is immediately loaded with the object properties, ready for direct manipulation. The properties that are available for modification are the same as for the creation of an application or virtual directory. Therefore, please refer to the sections about *adding applications* and *adding virtual directories* to web sites for further details.

In RayPack, IIS applications and virtual directories are used to manage content and functionality for IIS web sites. The IIS object manager tree on the left-hand side of the Visual Designer view IIS shows the nested structure of directories and applications per website.

Once a website has been created, users may add applications to the web site root, or to an already existing application or directory node, by right-clicking the desired parent of the new application and selecting **Add Application** from the context menu.

The **Add Application** dialog is displayed, waiting for user input regarding the properties described below. The **OK** button at the bottom of the dialog window becomes active as soon as all mandatory properties are defined. Clicking it saves the new application and closes the Add Application dialog. The Cancel button does the usual: discard the entered values and close the dialog without saving the new application.

Component
Each application object has to be related to a component item of the packaging project.

To select an existing component as parent object of the application item, users click on the downwards-pointing arrow at the right-hand side of the component selector control. A list of existing components is displayed. The actual selection is done with a click on any of the given components.

To create a new component for the application object, users click on the browse button **[...]** at the right-hand side of the component control. The *common dialog* type **Select Component** is displayed, ready for creating and selecting components.

Description
This value is the visible reference for the new application, which will be displayed within the IIS MMC applet as well as within RayPack. It has to be given as a non-empty string of max. 255 characters length.

Directory
The application directory has to be defined by selecting one of the already given directories within the packaging project, or by creating a new one. Both procedures may be done by clicking the browse button **[...]** at the right-hand side of the Directory control item, and using the interface of the *common dialog* **Select directory**.

Web dir property

Select one of the already available web directories stored within the current packaging project by clicking on the downwards pointing arrow at the right-hand side of the Web Dir selector control. If the required option is not present within the list, use the browse button **[...]** on the right to open the **Web Directories Manager** dialog. Please refer to the dedicated help topic for details on how to accomplish the required steps.

Web application

Select one of the already available web applications stored within the current packaging project by clicking on the downwards pointing arrow at the right-hand side of the Web Application selector control. If the required application is not present within the list, use the browse button **[...]** on the right to open the **Web Application Manager** dialog. Please refer to the dedicated help topic for details on how to accomplish the required steps.

In RayPack, IIS applications and virtual directories are used to manage content and functionality for IIS web sites. The IIS object manager tree on the left-hand side of the Visual Designer view IIS shows the nested structure of directories and applications per website.

Once a website has been created, users may add virtual directories to the web site root, or to an already existing application or directory node, by right-clicking the desired parent of the new application and selecting **Add Virtual Directory** from the context menu.

The **Add Virtual Directory** dialog is displayed, waiting for user input regarding the properties described below. The **OK** button at the bottom of the dialog window becomes active as soon as all mandatory properties are defined. Clicking it saves the new virtual directory and closes the Add Virtual Directory dialog. The **Cancel** button does the usual: discard the entered values and close the dialog without saving the new application.

Component

Each virtual directory object has to be related to a component item of the packaging project.

To select an existing component as parent object of the virtual directory item, users click on the downwards-pointing arrow at the right-hand side of the component selector control. A list of existing components is displayed. The actual selection is done with a click on any of the given components.

To create a new component for the virtual directory object, users click on the browse button **[...]** at the right-hand side of the component control. The *common dialog* type **Select Component** is displayed, ready for creating and selecting components.

Directory

The directory has to be defined by selecting one of the already given directories within the packaging project or by creating a new one. Both procedures may be done by clicking the browse button **[...]** at the right-hand side of the Directory control item and using the interface of the *common dialog* **Select directory**.

Web dir property

Select one of the already available web directories stored within the current packaging project by clicking on the downwards pointing arrow at the right-hand side of the Web Dir selector

control. If the required option is not present within the list, use the browse button **[...]** on the right to open the **Web Directories Manager** dialog. Please refer to the dedicated help topic for details on how to accomplish the required steps.

In RayPack, there is a dedicated interface for managing IIS web applications. It is displayed whenever users try to manipulate the web application property of another IIS data object. Information stored for web applications can be reviewed directly via accessing the `RPIIsWebApplication` table.

The Web Applications Overview

The IIS web applications manager is launched with an initial list view of already existing web applications. Above the list, there is a button **NEW**, which allows to *define the properties of a new web application object* within another dialog window.

The list view also offers a context menu for each existing web application, which is displayed whenever a user performs a right-click on one of the web application objects. The context menu allows to trigger the **edit** and **remove** procedures for web application objects.

To Add a New Web Application

To trigger the dialog for adding web applications, users have to click on the **NEW** button displayed within the web application properties overview. From the options menu, **Web Application Property** has to be clicked. The displayed dialog is separated into several tabs to provide a structured approach to the complex data object.
At any time and from any tab, users may use the **OK** button to save the new object and return to the updated web application properties overview. Clicking the **Cancel** button discards the object creation.

Tab: GENERAL

Name
The name of the application as it will be visible within the IIS Manager interface.

Isolation
Sets the application isolation level. Possible values are:

- low: the application executes within the IIS process. (= 0)
- medium: the application executes pooled in a separate process. (= 1)
- high: the application executions in a separate process. (= 2)

App pool
Defines which application pool the application actually belongs to. The selector offers a set of already defined application pools of the current packaging project. If the desired application pool is not available, it has to be added to the project via the *Add an IIS application pool* procedure.

Tab: BEHAVIOR

Script timeout
Sets the timeout value for executing ASP scripts in seconds.

Enable Buffering
Sets the option that enables response buffering in the application, which allows ASP script to set response headers anywhere in the script.

Enable Parent Path
Sets the parent paths option, which allows a client to use relative paths to reach parent directories from this application.

Tab: COMPILATION

Script language
The default script language may either be "`VBScript`" or "`JScript`".

Debugging
Use the provided checkboxes to control
- server-side script debugging
  and
- client-side script debugging

Active checkboxes indicate that the debugging option is active as well.

Tab: SERVICES

Enable Session State
Allows to use the session timeout attribute.

Session timeout
Sets the timeout value for sessions in minutes.

To Edit a Web Application

When an existing web application has to be edited, users need to define the very same properties they had to define for the initial web application creation itself. Therefore, to add a new web application please refer to the section above for further details about these properties.

However, editing a web application means to edit an object that may very well be referenced by several IIS objects at the same time. This means, that changing the unique set of properties of a specific web application may have influence on the settings and behavior of several web sites, web site applications, and web site virtual directories. Therefore, please double-check any change to web application settings and make sure that all referenced objects are still fully functional after the modifications.

To Remove a Web Application

The Remove procedure is triggered as outlined above: by a click on any web application objects context menu item **Remove**. Once an item is removed, it is lost from the packaging project.

> **Be aware:**
> If an object has child nodes when it is marked for removal, those child elements are deleted along with their parent.

Confirm Object Removal

Since an accidental object removal could lead to substantial issues, RayPack displays a **Confirm**

dialog before a web application object is actually removed. Use the **REMOVE** button within that dialog to confirm the activity. If **DO NOT REMOVE** or **CANCEL** are clicked, the object is not deleted and the user returns to the web application manager view without any changes.

The display of the **Confirm** dialog may manually be abandoned for future remove procedures by activating the "In future do not show this confirmation for delete operations" checkbox and using the **REMOVE** or **DO NOT REMOVE** button next. (Using the **CANCEL** button does not save any changes made to the status of that checkbox.) However, it is not recommended to skip the confirmation step, since it may prevent users from severe data loss.

In RayPack, there is a dedicated interface for managing IIS web directory properties. It is displayed whenever users try to manipulate the web directory property of another IIS data object. Information stored for web directories can be reviewed directly via accessing the `RPIIsWebDirProperties` table.

The Web Directory Properties Overview

The IIS web directory property manager is launched with an initial list view of already existing web directory objects. Above the list, there is a button **NEW**, which allows to *define the properties of a new web directory property object* within another dialog window.

The list view also offers a context menu for each existing web application, which is displayed whenever a user performs a right-click on one of the web application objects. The context menu allows to trigger the **edit** and **remove** procedures for web application objects.

Be aware:
There may be issues with the edition of RPP projects, which have originally been created with prior RayPack versions. These issues are an outcome of required changes to the database table schema for IIS management. Please refer to the section about *custom installer database tables* for further details.
If any issues occur, please contact our *support team*, who will gladly assist on upgrading the project files.

To Add a New Web Directory Property Object

To trigger the dialog for adding web directories, users have to click on the **NEW** button displayed within the web directory properties overview. From the options menu, **Web Directory Property** has to be clicked. The displayed dialog is separated into several tabs to provide a structured approach to the complex data object.

At any time and from any tab, users may use the **OK** button to save the new object and return to the updated web directory properties overview. Clicking the **Cancel** button discards the object creation.

Tab: AUTHENTICATION

RPUser
Accessing system parts in the name of an explicit user identity usually requires credentials. These have to be given in the form of a user object. To select or create a user object, click on the browse button **[...]** at the right-hand side of the user input field. The **Select User** dialog is

displayed. Please refer to the *Common dialogs* section for further details on how to *manage user data objects*.

> **Be aware:**
> User objects may very likely be used at several places within SQL database, script, and IIS management structures. When the same user item is referred to from different locations, changing the properties of that user item actually takes effect for all related objects. Please double-check the correctness of user profile changes, and make sure all related objects are still valid and operational with the modified set of properties.

Authentication providers
Comma delimited list, in order of precedence, of Windows authentication providers that IIS will attempt to use: NTLM, Kerberos, Negotiate, and others.

Authorization
Authorization policy to web server (anonymous access, NTLM, etc.)

IIS controlled password
Sets whether IIS should control the password used for the Windows account specified in the AnonymousUser attribute. The defaults state for this checkbox is inactive.

Tab: SSL SETTINGS

Require SSL
An active checkbox indicates that file access requires SSL file permission processing with or without a client certificate. This corresponds to AccessSSL flag for AccessSSLFlags IIS metabase property.

Require 128bit SSL
If this option is set, simple SSL does not suffice to access files since 128-bit SSL is requried.

Tab: OUTPUT CACHING

Cache control custom
Custom HTTP 1.1 cache control directives.

Cache control max. age
Integer value specifying the cache control maximum age value in seconds.

Tab: DOCUMENT

Default doc
The list of default documents to set for this web directory in comma-delimited format.

Http expires
Value to set the HttpExpires attribute to for a Web Dir in the metabase.

Index
Specifies whether IIS searches the directory.

ASP detailed error
Sets the option for whether to send detailed ASP errors back to the client on script error. Default is inactive.

No custom error
Specifies whether or not IIs will return custom errors for this directory.

Log visits
Sets whether visits to this site should be logged. Default is set to inactive.

To Edit a Web Directory Property Object

When an existing web directory property object has to be edited, users need to define the very same properties they had to define for the initial web directory property object creation itself. Therefore, please refer to the section *To add a new web directory property object* for further details about these properties.

However, editing a web directory property object means to edit an object that may very well be referenced by several IIS objects at the same time. This means, that changing the unique set of properties of a specific web directory property object may have influence on the settings and behavior of several web sites, web site applications, and web site virtual directories. Therefore, please double-check any change to web directory property object settings and make sure that all referenced objects are still fully functional after the modifications.

To Remove a Web Directory Property Object

The Remove procedure is triggered as outlined above: by a click on any web directory property objects context menu item **Remove**. Once an item is removed, it is lost from the packaging project.

> **Be aware:**
> If an object has child nodes when it is marked for removal, those child elements are deleted along with their parent.

Confirm Object Removal

Since an accidental object removal could lead to substantial issues, RayPack displays a **Confirm** dialog before a web application object is actually removed. Use the **REMOVE** button within that dialog to confirm the activity. If **DO NOT REMOVE** or **CANCEL** are clicked, the object is not deleted and the user returns to the web directory properties manager view without any changes.

The display of the **Confirm** dialog may manually be abandoned for future remove procedures by activating the "In future do not show this confirmation for delete operations" checkbox and using the **REMOVE** or **DO NOT REMOVE** button next. (Using the **CANCEL** button does not save any
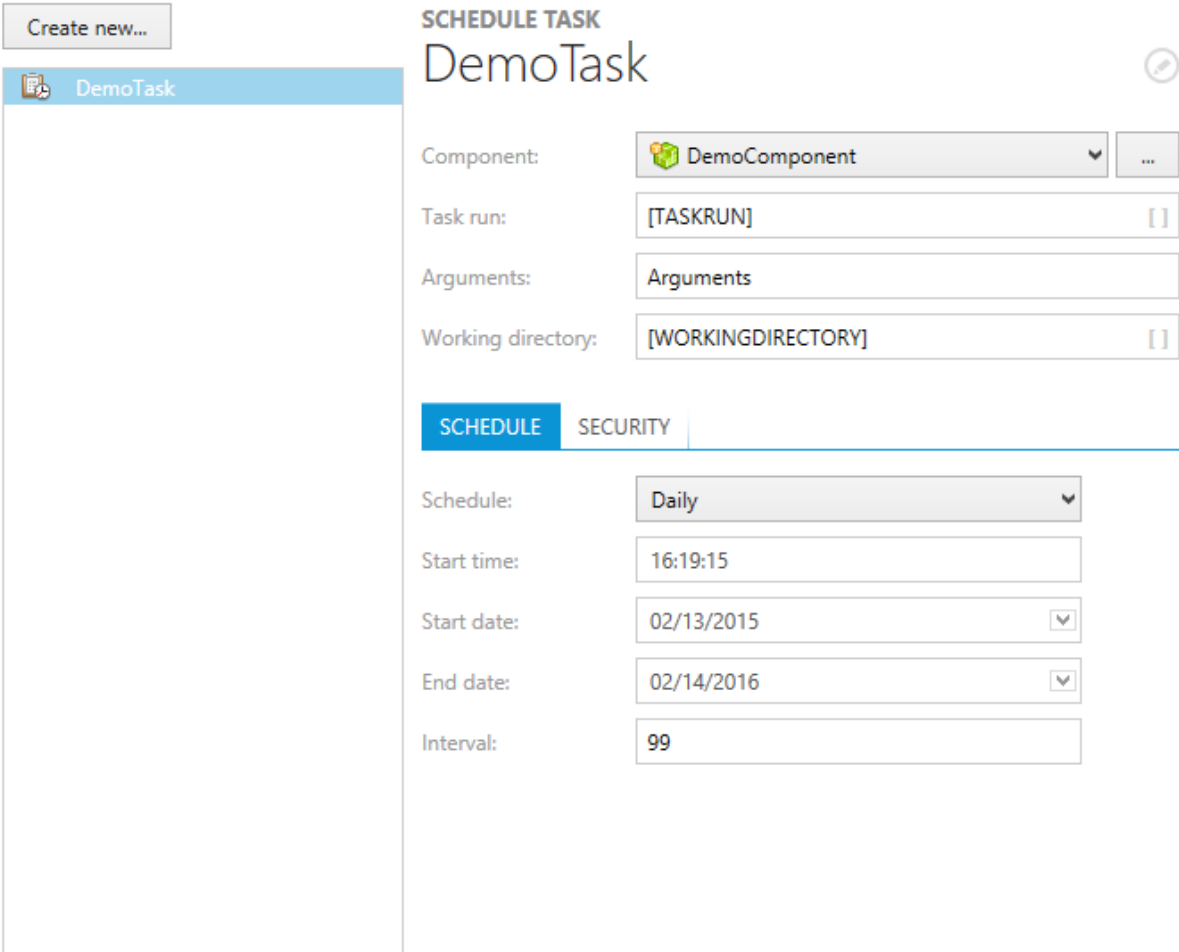
changes made to the status of that checkbox.) However, it is not recommended to skip the confirmation step, since it may prevent users from severe data loss.

Scheduled Tasks

This view allows to manipulate the **Scheduled Tasks** on a target system.

The **Scheduled Tasks** view is separated into two main areas:
- A list of already available tasks on the left-hand side
- A details pane for viewing and editing the properties of already existing tasks at the right-hand side

The Scheduled tasks view allows packagers to execute the following set of standard functions:

- Add a scheduled task
- Rename a scheduled task
- Remove a scheduled task
- Edit a scheduled task

As soon as a scheduled task object is right-clicked, the context menu offers the option **Go to row**, which enables to switch to the TABLES editor of the Advanced mode, with the data row of the currently displayed scheduled task object focused within the custom RPScheduledTasks table.

Add a Scheduled Task

In order to add an IIS application pool to a packaging project, users go to the IIS view of the Visual Designer mode.

- With a click on the **Create new...** button, the options menu is displayed. Click on **Application pool** to invoke the New Application Pool Wizard.

- As an alternative access to the very same wizard, users may click on the **Application pools node** within the IIS object tree structure on the left hand side of the IIS view, and click on the **Add new application pool** button that is displayed within the details pane at the right-hand side.

Work your way through the steps of the wizard to define all required properties for the new item.

At any time, using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, allows to navigate within the already processed steps.
To exit the wizard without creating a new object, use the Cancel button, also located at the bottom of the wizard dialog.

Step 1: Scheduled Task

Task name
The name of the task as it will be displayed within the target systems task scheduler interface. The name should be unique, and has to be given as a string of max. 255 characters.

Step 2: Component

Component
Each scheduled task has to be related to a component item of the packaging project.

To select an existing component as parent object of the scheduled task, users click on the downwards-pointing arrow at the right-hand side of the component selector control. A list of existing components is displayed. The actual selection is done with a click on any of the given components.

To create a new component for the scheduled task, users click on the browse button **[...]** at the right-hand side of the component control. The *common dialog* type **Select Component** is displayed, ready for creating and selecting components.

Step 3: Task Run

Task run
A value that specifies the path and file name of the task to be run at the scheduled time.

Arguments

The parameter arguments that have to be passed to the executable defined as task run.

Working directory

An optional value that specifies the the working directory for the executable defined as task run.

> 📎 **Note:**
> As indicated by the square brackets ("`[]`") at the right hand side of the input fields for task run and working directory, it is possible to make use of syntax suggestions to define their values. Simply enter an opening square bracket ("`[`") to display the list of available suggestions. Select the desired value from the list to add it to the already entered value of the input field. The actual value at run time will be used for the scheduled task definition.

Step 4: Schedule Type

Select one of the predefined schedule types:
- **Once** - the task will run once, at a specific date and time
- **Daily** - the task will run every day at a specific time
- **Day of month** - the task will run on a specified list of months and days during a defined calendar period, each at the generally specified time
- **Weekly** - the task will run on a selection of weekdays at a specific time
- **Week of month** - the task will run on a specified list of months, weeks and days during a defined calendar period, each at the generally specified time
- **Idle** - the task will run after a specified period of idle time
- **System Start** - the task runs each time the computer starts up
- **Logon** - the task runs each time a user logs on to the computer

The upcoming step Schedule Details will be adjusted to request exactly the scheduling settings required for the type selected within this step. Since System Start and Logon do not need no further specifications, RayPack skips the details steps for these types.

Step 5: Schedule Details

Once

Start time

The start time is defined as `HH:MM:SS` for the local target machine time zone (e. g. 16:43:00 equals 4pm 43 minutes and 0 seconds).

Start date

The start date is defined as `MM/DD/YYYY` for the local target machine time zone (e. g. 03/02/2015 equals the 2nd of March 2015).

Daily

Start time - *see above*

Start date - *see above*

End date
The end date is defined as `MM/DD/YYYY` for the local target machine time zone (e. g. 05/06/2015 equals the 6th of May 2015).

Interval
An integer value that refines the schedule type to allow for more detailed control over the schedule recurrence. If it is set to 1, the scheduled task will be executed exactly once. If it is set to 99, the scheduled task execution will be triggered 99 times (which equals 99 days in a row for a daily schedule type).

Day of Month

Start time - *see above*
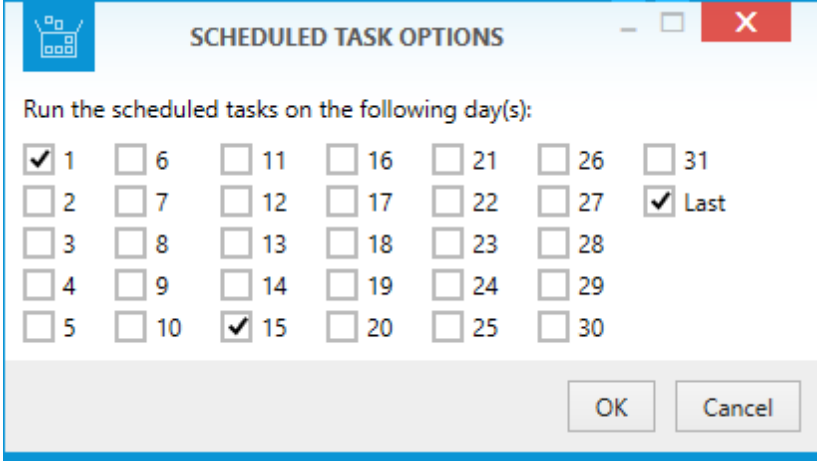
Start date - *see above*

End date - *see above*

Days
A value that specifies the day(s) of the month to run the task on. A comma (,) has to be used to separate a list of month days.

Please use the browse button **[...]** at the right-hand side of the input field to get support for the definition of more complex day selection.

Within the displayed Scheduled Task Options dialog, activate the checkbox at the left-hand side of the month day number. Note that due to the irregular number of days per month (28,29,30,31), there is the additional option "Last", which makes sure that the task is run at the last day of the month, no matter how many days the month actually has.

Hit **OK** to copy the defined set of days into the parent scheduled task wizard step and close this dialog. Hit Cancel to discard the selection without taking it over to the parent scheduled task wizard step.

Months

A value that specifies the month(s) of the year to run the task on. A comma (,) has to be used to separate a list of month names.

Please use the browse button **[...]** at the right-hand side of the input field to get support for the definition of more complex month selection.
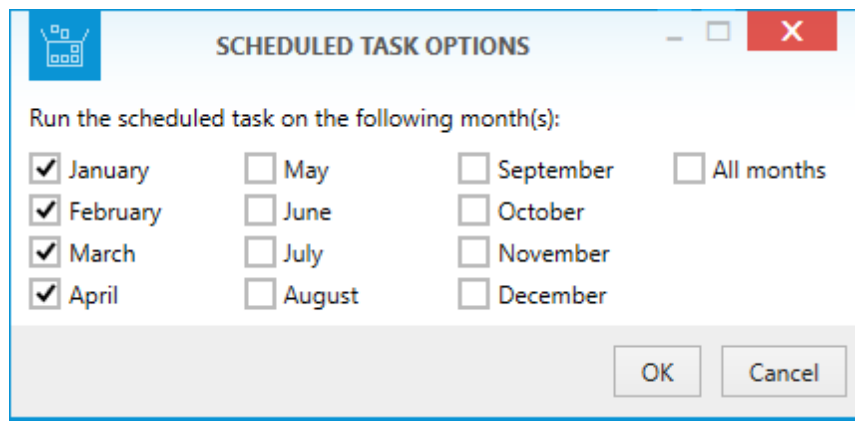
Within the displayed Scheduled Task Options dialog, activate the checkbox at the left-hand side of the month name. Note that due to the tendency of people who work with IT to be lazy, there is the additional option "All months", which enables / disables all checkboxes at once.

Hit **OK** to copy the defined set of months into the parent scheduled task wizard step and close this dialog. Hit Cancel to discard the selection without taking it over to the parent scheduled task wizard step.

Weekly

Start time - *see above*

Start date - *see above*

End date - *see above*

Interval - *see above*

Days
A value that specifies the day(s) of the week to run the task on. A comma (,) has to be used to separate a list of day names.

Please use the browse button **[...]** at the right-hand side of the input field to get support for the definition of more complex day selection.
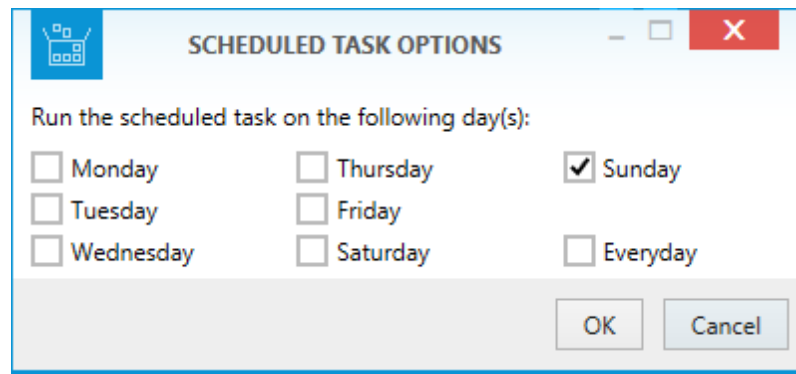
Within the displayed Scheduled Task Options dialog, activate the checkbox at the left-hand side of the day names. Note that due to the tendency of people who work with IT to be lazy, there is the additional option "Everyday", which enables / disables all checkboxes at once.

Hit **OK** to copy the defined set of week days into the parent scheduled task wizard step and close this dialog. Hit Cancel to discard the selection without taking it over to the parent scheduled task wizard step.

Week of Month

Start time - *see above*

Start date - *see above*

End date - *see above*

Week
Select one of the predefined options:
- First

- Second
- Third
- Fourth
- Last

Note that due to the irregular number of weeks per month, there is the additional option "Last", which makes sure that the task is run at the last week of the month, no matter how many weeks the month actually spreads across.

Days - *see above*

Months - *see above*

Idle

Idle time
The idle time has to be given as integer number of minutes, calculated from the moment of screen saver activation.

Step 6: Security

User
RayPack allows to run scheduled tasks either via Windows Authentication, or as a specific user. The default setting is Windows Authentication, indicated by the active checkbox Use Windows Authentication. To enable the usage of individual user credentials, the checkbox needs to be deselected. Once it is inactive, the controls for user object management become active for user selection.

Individual user credentials have to be given in the form of a user object. To select or create a user object, click on the browse button **[...]** at the right-hand side of the user input field. The **Select User** dialog is displayed. Please refer to the **Common dialogs** section for further details on how to *manage user data objects*.

	Be aware:
	User objects may very likely be used at several places within SQL database, script, and IIS management structures. When the same user item is referred to from different locations, changing the properties of that user item actually takes effect for all related objects. Please double-check the correctness of user profile changes, and make sure all related objects are still valid and operational with the modified set of properties.

Step 7: Summary

Use the summary page to check the correctness of the scheduled task properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the scheduled task item.

- If changes are due, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the summary page again.

Step 8: Finished

Once the new scheduled task object has been created, the wizard can be closed by using the **Finish** button at its lower right corner. The Scheduled Tasks view is updated, and the list of existing items contains the newly created object.

Rename a Scheduled Task

1. To rename a scheduled task, users load the list of existing tasks by calling the **Scheduled Tasks** view within the Visual Designer mode of PackDesigner.

2. Clicking on the list item of the object that has to be renamed displays its details in the editor panel on the right hand side of the RayPack application screen.

3. The task name is displayed above the form elements of the edit area.

4. The current value becomes editable with a click on either the name text, or the edit icon on the right hand side. A *direct value editor dialog* is displayed, allowing to enter the new content.

   The name should be unique, and has to be given as a string of max. 255 characters.

5. By hitting **Enter** or clicking on the **save** icon within the direct value editor interface the new task name is saved.

Remove a Scheduled Task

The PackDesigner mode Visual Designer contains a Scheduled Task view with a list of all objects stored within a packaging projects `RPScheduledTasks` table. To remove an item by using this view interface, users have the following options at hand:

- Within the list of scheduled tasks, **right-click** any item, and select **Delete** from the **context menu**.

The remove procedure is triggered. Once a scheduled task is removed, it is lost from the packaging project.

> **Be aware:**
> If user objects were created during the **Add scheduled task** wizard execution, these additional packaging project contents are not automatically removed when the scheduled task itself is deleted. Please make sure to remove them manually if required.

Confirm Object Removal

Since an accidental object removing could lead to substantial issues, RayPack displays a **Confirm** dialog before an essential project content object is actually removed. Use the **REMOVE** button within that dialog to confirm the activity. If **DO NOT REMOVE** or **CANCEL** are clicked, the object is not deleted and the user returns to the parent view without any changes.

Edit a Scheduled Task

1. To edit a scheduled task object, users load the list of existing items by calling the **Scheduled Tasks** view within the Visual Designer mode of PackDesigner.

2. Clicking on the list item of the object that has to be modified displays its details in the editor panel on the right-hand side of the RayPack application screen.

The following properties of the scheduled task object may be manipulated:

Task Name

Please refer to the topic *Rename a scheduled task* for details.

Basic Scheduled Task Properties

Component
Each scheduled task has to be related to a component item of the packaging project.

To select an existing component as parent object of the scheduled task, users click on the downwards-pointing arrow at the right-hand side of the component selector control. A list of existing components is displayed. The actual selection is done with a click on any of the given components.

To create a new component for the scheduled task, users click on the browse button **[...]** at the right-hand side of the component control. The *common dialog* type **Select Component** is displayed, ready for creating and selecting components.

Task run
A value that specifies the path and file name of the task to be run at the scheduled time.

Arguments
The parameter arguments that have to be passed to the executable defined as task run.

Working directory
An optional value that specifies the the working directory for the executable defined as task run.

> **Note:**
> As indicated by the square brackets ("[ ]") at the right hand side of the input fields for task run and working directory, it is possible to make use of syntax suggestions to define their values. Simply enter an opening square bracket ("[") to display the list of available suggestions. Select the desired value from the list to add it to the already entered value

of the input field. The actual value at run time will be used for the scheduled task definition.

Tab: SCHEDULE

Please refer to the topic about the *scheduling type* and *details* steps of the *Add new scheduled task* chapter for details about the available options and the settings each of them requires.

Tab: SECURITY

User
RayPack allows to run scheduled tasks either via Windows Authentication, or as a specific user. The default setting is Windows Authentication, indicated by the active checkbox Use Windows Authentication. To enable the usage of individual user credentials, the checkbox needs to be deselected. Once it is inactive, the controls for user object management become active for user selection.

Individual user credentials have to be given in the form of a user object. To select or create a user object, click on the browse button **[...]** at the right-hand side of the user input field. The **Select User** dialog is displayed. Please refer to the Common dialogs section for further details on how to manage user data objects.

Be aware:
User objects may very likely be used at several places within SQL database, script, and IIS management structures. When the same user item is referred to from different locations, changing the properties of that user item actually takes effect for all related objects. Please double-check the correctness of user profile changes, and make sure all related objects are still valid and operational with the modified set of properties.

Setup Options

The **Setup options** view allows you to make changes to the launch condition, system search, installer option, and administrator option settings.

If an MSI file is opened for manipulation in PackDesigner, an additional view for file compression options is part of the Setup options section.

Note:
If any of the items listed above is not available within your RayPack installation, check for the actual set of features covered by your license. To do so, visit the About section and expand the license details with the full listing of licensing options.

Launch Conditions

This view allows to manipulate the **Launch Conditions** for a package run-time phase to be executed on a target machine.

The **Launch Conditions** view is separated into two main areas:

- A list of already available conditional statements on the left-hand side
- A details pane for viewing and editing the properties of already existing conditions at the right-hand side
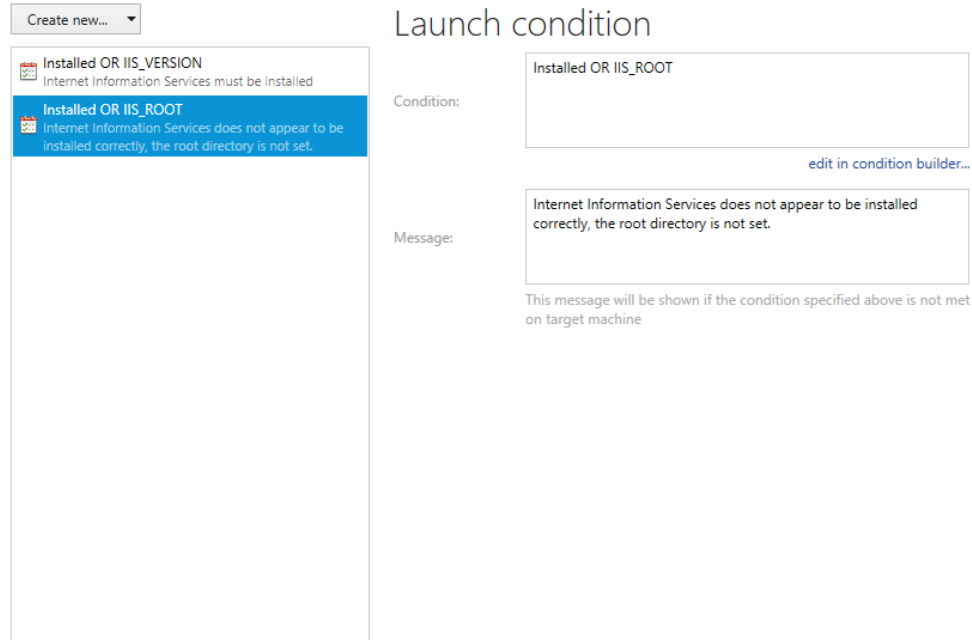
The **Launch Conditions** view allows packagers to execute the following set of standard functions:

- *Add a launch condition*
- *Duplicate a launch condition*
- *Remove a launch condition*
- *Edit a launch condition*

As soon as a launch condition object is right-clicked, the context menu offers the option **Go to row**, which enables to switch to the **TABLES** editor of the Advanced mode, with the data row of the currently displayed scheduled task object focused within the `LaunchCondition` table.

Add a Launch Condition

In order to add a new launch condition to a packaging project, users go to the **Launch Conditions** view of the Visual Designer mode.
With a click on the **Create new...** button, the options menu is displayed. Click on **Create launch condition** to invoke the **New Row in Launch Condition Table** dialog.

The **OK** button at the bottom of the dialog window becomes active as soon as all mandatory properties are defined. Clicking it saves the new launch condition and closes the current dialog. The Cancel button does the usual: discard the entered values and close the dialog without saving the new data object.

Condition
A conditional statement that needs to evaluate to true in order to process the package run-time phase on the target machine.

Description
A textual description of the message displayed on the target machine during package run-time if the conditional statement defined above evaluates to false.

Duplicate a Launch Condition

Since the conditional statements and messages defined for a launch condition may very likely be quite similar to the ones required for another launch condition, the RayPack user interface offers the option to copy an existing launch condition.

To do so:

1. Users have to **right-click** the existing launch condition that has to be duplicated, and select **copy** from the **context menu**.

2. **Right-click** the list area at the left-hand side of the Launch Conditions view, and select **paste** from the **context menu**.

3. The duplicate launch condition is created with a slightly modified conditional statement: an incrementing index value is automatically added to the condition string. When the paste option is triggered the first time, a "0" is added. When it is used a second time, a 1 is added, and so on.

4. Please **adjust the conditional statement** of the newly created duplicate by the interface options provided for *launch condition edition*.

Remove a Launch Condition

To remove a launch condition from a packaging project by using the PackDesigner Visual Designer mode interface functionality:

Users have to call the **Launch Conditions** view. From the list of conditions on the left hand side, **right-click** a condition item and select **Remove** from the **context menu**

The launch condition is immediately deleted from the packaging project.

Be aware:
As soon as an item is deleted from the Launch Conditions view, its resembling data row as well is deleted from the affected Installer database table `LaunchCondition`.

Edit a Launch Condition

1. To edit a launch condition object, users load the list of existing items by calling the **Launch Conditions** view within the Visual Designer mode of PackDesigner.

2. Clicking on the list item of the object that has to be modified displays its details in the editor panel on the right-hand side of the RayPack application screen.

The following properties of the launch condition object may be manipulated:

Condition

The text area displays the current conditional statement defined for the launch condition. Users may either edit it by manually typing the updated phrase segments. The more handy editor interface for conditional statement manipulation in RayPack is the condition builder. Please refer to the common dialogs section for further details on how to use this standard interface type.

Message

A textual description of the message displayed on the target machine during package run-time if the conditional statement defined above evaluates to false.

System Search

This view allows to manipulate system search tasks for a package run-time phase to be executed on a target machine.

The System Search view is separated into two main areas:
- A list of already available search tasks on the left-hand side
- A details pane for viewing and editing the properties of already existing searches at the right-hand side

The System Search view allows packagers to execute the following set of standard functions:

- *Add a predefined search*
- *Add a custom search*
- *Remove a system search*
- *Edit a system search*

As soon as a system search object is right-clicked, the context menu offers the option **Go to row**, which enables to switch to the **TABLES** editor of the Advanced mode, with the data row of the currently displayed system search object focused within the respective table.

Depending on the type of search, each task may consist of a varying set of objects. Additionally to the main system search item, stored within the Installer database table `AppSearch`, there may be others, stored within tables such as `IniLocator`, `RegLocator`, `DrLocator`, `CompLocator`, or `Signature`. Which ones are available per search, depends on the type of searched object on the target device.

System Search Task Properties

Within the list view of tasks, each system search task summary is displayed accompanied with a specific icon, at the left-hand side of the tasks root or child item(s). The icon indicates which type of search criterion the item defines:

File
Properties are handled within the `Signature` Installer database table.

Path
Properties are handled within the `DrLocator` Installer database table.

Component
Properties are handled within the `CompLocator` Installer database table.

Registry
Properties are handled within the `RegLocator` Installer database table.

INI
Properties are handled within the `IniLocator` Installer database table.

General system properties
This icon is used to indicate general system search root criteria, which is stored within the `AppSearch` table.

- Each search task has a **root item**, defining the target property that will save the search result for later re-use during the package run-time.
  - If a **file** has to be searched on the target device, that parent root has to have **two child elements**:
    - One file item that defines key properties required to determine if a found file matches the search
    - One other item that defines the path to the searched file: registry, component, ini or path.
  - If the existence of a **path, registry value, component or INI value** has to be determined, the root item of the system search task has to have exactly **one child element**, defining where to find the object of interest.

> **WARNING**
> RayPack actually allows to define system search tasks with more or less criteria definition items than specified above. It may be necessary to extend the standard system search operations with more complex structures in order to fulfill more complex searches. However, to provide high-level package quality, any non-standard system search definition needs to be tested thoroughly before it is used in a productive software package.

Add a Predefined Search

In order to add a new predefined system search to a packaging project, users go to the **System Search** view of the Visual Designer mode.
With a click on the **Create new...** button, the options menu is displayed. Click on **Predefined search** to invoke the **Select Predefined Searches** dialog.

Once the dialog is displayed, it presents a list of searches, grouped by target application or system property a user might want to search for on the target system. The group titles are equipped with arrow icons at their left-hand side. If the group details are not visible when the dialog is displayed, a click on those arrow icons expands them group wise.

To add one of the predefined searches to the current packaging project, the checkbox at the left-hand side of the predefined search title has to be activated.

As soon as at least one search is selected this way, the **OK** button at the bottom of the dialog window becomes available. Hit it to add the selected predefined search(es) to the current project. Hit Cancel to discard the selection and close the dialog without adding a search to the packaging project.

Note:

The Select Predefined Searches dialog does not reflect the current existence of a predefined search within a packaging project. This means that whenever the dialog is opened in RayPack, all checkboxes are inactive, and all searches can be added to the project. Well, this may lead to duplicate searches on the one side, but at the same time allows users to take any predefined search and used it as starting point for further individual fine tuning.

However, please make sure to define a set of unique and required system search tasks for the final package, since searching a system may take noticeable time. Performing the same search twice is unnecessary and by all means does not prove high package quality.

As soon as the new system search task has been added to the packaging project according to the predefined settings, there is a new root item within the list of system search tasks for the current packaging project. As outlined before, there may be one of more child-nodes attached to the root. If at all, and how many exactly there are, depends on the type of search.

Please refer to the *Edit a system search* topic for further details regarding possible child-nodes and their modification options.

Add a Custom Search

In order to add a new predefined system search to a packaging project, users go to the **System Search** view of the Visual Designer mode.
With a click on the **Create new...** button, the options menu is displayed. Click on **Custom search** to invoke the **New System Search Wizard**.

Work your way through the steps of the wizard to define all required properties for the new item.

At any time, using the **Next** or **Back** buttons, which are displayed at the bottom of the wizard dialog, allows to navigate within the already processed steps.
To exit the wizard without creating a new object, use the Cancel button, also located at the bottom of the wizard dialog.

Step 1: Property

Property
Please select the property that will store the result of the system search. It has to be a public (Uppercase) property, which is already present in the current packaging project. If the desired one is not shown within the property options, exit the wizard and create it by the interface available from the **Properties** view.

To actually pick a property, click on the option of the selector controls option list, an use the Next button at the bottom of the dialog, which becomes active as soon as a search type is selected.

Step 2: Search Type

There is a predefined set of typical search types, users may select from:

- File - search for a file on the target machine
- Path - search for a directory on the target machine
- Registry - search for a registry value on the target machine
- Component - search for a component on the target machine
- INI - search for a specific INI file value on the target machine

To actually pick a search type, click on the icon at the left-hand side of its name, an use the Next button at the bottom of the dialog, which becomes active as soon as a search type is selected.

Step 2a: Search Location Type [Only for Search Type = File]

If searching for a file has been selected during step 2, this additional step is required to define where the path to the requested file will be stored:

- Path - the path is exactly the same as for the given file source
- Registry - the path is stored within a registry value
- Component - the path is defined by a component key path
- INI - the path is defined within an INI file

To actually pick a search location type, click on the icon at the left-hand side of its name, an use the Next button at the bottom of the dialog, which becomes active as soon as a search type is selected.

Step 2b: File Options [Only for Search Type = File]

If searching for a file has been selected during step 2, this additional step is required to define some key properties used to determine whether a found file matches the search:

File name
Use the browse button at the right-hand side of the input field to select a file from the local system. The properties provided by the selected file will automatically be read to fill in the following properties as well as possible:

Minimum / Maximum version
Enter a value to determine the minimum / maximum version of the searched file. If the file on the target machine does not have a version between minimum and maximum, it is not regarded as a match for the system search task.
If only one version limit is entered, any file version above (if a minimum was given) or below (if a maximum was given) will be valid for a match.

Minimum / Maximum size
Enter a value to determine the minimum / maximum size of the searched file in bytes. If the file on the target machine does not have a file size between minimum and maximum, it is not

regarded as a match for the system search task.

If only one size limit is entered, any file size above (if a minimum was given) or below (if a maximum was given) will be valid for a match.

Not older / newer than

Activate the checkbox at the right-hand side of the option label to enable the controls for minimum and / or maximum file age limitation.

Click on the arrow of the active date limitation control to select a date via the interface provided by the calendar helper, or enter a date directly, respecting the format demand of `DD.MM.YYYY`.

When the calendar helper with its three columns day, month, and year is displayed, move the mouse pointer over the column that has to be changed (e. g. over the day column), and use the mouse wheel to spin it up or down. As an alternative, it is also possible to click on the current column value, and use the up and down arrow keys on the keyboard to change the selection. Using the right and left arrow keys allows to switch between the three rows of the calendar selector dialog.

The values that are currently presented at the middle row of the calender selector helper are immediately taken over into the triggering date field. To close the dialog, users have to either hit Escape on the keyboard, or left-click somewhere outside the calendar selector dialog.

Once the date has been selected, a time index needs to be defined as well. Please make sure to use the right format `HH:MM`.

Languages

Use the browse button at the right-hand side of the input field to open the language selector dialog. Activate the checkbox at the left-hand side of the language name to add it to the list of possible language matches. Use the OK button at the bottom of the language selector interface to copy the selection into the triggering language version input field (as a comma separated list of language ID's, or use the Cancel button to close the dialog without taking over any new selections.

Step 3: Location Options

Depending on the type selected in step 2, there are some location specifications due.

Folder Location Options

Path

The path of the searched file has to be defined by selecting one of the already given directories within the packaging project, or by creating a new one. Both procedures may be done by clicking the browse button [...] at the right-hand side of the destination folder property control item, and using the interface of the *common dialog* **Select directory**.

Depth

Please enter an integer value for the number of levels below the defined path that will be searched for the key object.

Registry Location Options

Root
Select one of the predefined registry hive options:

- `HKEY_CLASSES_ROOT`
- `HKEY_CURRENT_USER`
- `HKEY_LOCAL_MACHINE`
- `HKEY_USERS`

The searched registry content must be present within the selected hive.

Key
Enter the full key name that leads to the searched value. Make sure to use backslashes to define the several levels of the requested registry branch.

Name
Enter the name of the registry value to search for. Please make sure to give only the name, without leading or trailing blanks, without hierarchy information (such as parent key levels, etc.) It is recommended to match the case as well.

Type

Please pick one of the predefined options:

- Key path is a directory
- Key path is a file
- Key path is a registry value

Search 64bit portion of the registry

If this checkbox is active, the system search task will be executed on the 64bit branch of the registry. If it is not active, the 32bit branch is scanned.

Component Location Options

Component

To select an existing component as search object, users click on the downwards-pointing arrow at the right-hand side of the component selector control. A list of existing components is displayed. The new setting is established with a click on any of the given components.

To create a new component as search object, users click on the browse button **[...]** at the right-hand side of the component control. The *common dialog* type **Select Component** is displayed, ready for creating and selecting components.

Type

Please pick one of the predefined options:

- Key path is a directory
- Key path is a file

INI Location Options

FileName

Please enter the file name of the INI file that has to be scanned.

Section

Once the INI file is found, a specific section within that file has to be searched for.

Key

Within the specified INI file and section, the key name entered here will be searched for.

Field

Please enter the numeric index value of the field that has to contain the searched value.

Type

Please pick one of the predefined options:

- .ini value is a directory location
- .ini value is a file location
- Use raw .ini value

Step 4: Summary

Use the summary page to check the correctness of the system search properties that were defined during the previous wizard steps.

- If all properties are set as required, click **Process** to finally create the system search item.
- If changes are due, click **Back** until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.
  Please note that changes in an early step may lead to different defaults or options in any later step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the summary page again.

The wizard is closed automatically as soon as the new system search task has been created. The list of existing system search tasks within the **System Search** view is updated to contain the newly created task.

Remove a System Search

The PackDesigner mode Visual Designer contains an **System Search** view with a list of all objects required for the execution of search tasks within the current packaging project. To remove a whole search task by using this view interface, users have the following options at hand:

- Within the list of search tasks, **right-click** any root item, and select **Remove** from the **context menu**.

The remove procedure is triggered. Once a search task is removed, it is lost from the packaging project.

Be aware:
If additional objects were created during the *Add custom search* or *Add predefined search* execution, these additional packaging project contents are not automatically removed when the search task itself is deleted. Please make sure to remove them manually if required.

Confirm Object Removal

Since an accidental object removing could lead to substantial issues, RayPack displays a **Confirm** dialog before an essential project content object is actually removed. Use the **REMOVE** button within that dialog to confirm the activity. If **DO NOT REMOVE** or **CANCEL** are clicked, the object is not deleted and the user returns to the parent view without any changes.

Remove a Single System Search Criterion

Since each system search task consists of a root element and a varying number of children with additional search criterion definitions, it is not only possible to remove a search task in general,

but as well to remove each child individually.

To remove a specific child object of a system search task, but leave the search task itself as part of the packaging project, users have to right-click the child object itself, and not the parent search task root object. With a click on the Remove option displayed within the context menu, the child object is immediately removed - there is no intermediate confirm dialog step in between.

> **Be aware:**
> Once the child object is removed, the search task will need some further detail definitions to be actually operational. Please refer to the *Edit a system search* topic to read more about how to add further detail definitions to a system search task.

Edit a System Search

1. To edit a system search object, users load the list of existing items by calling the **System Search** view within the Visual Designer mode of PackDesigner.

2. To edit the root element of a system search task, users left-click that root object, and access its details, which are displayed within the edit pane at the right-hand side of the application screen

3. To edit a child element of a system search task, users left-click the specific child of the root task object. There may be one or more child objects per root item, since each search type required different key criteria to operate.

System Search - Root Object Properties

When a system search root object is selected for edition, users may change the property which will be used to store the search result at run time.
Please enter the name of an existing, public property of the current packaging project.

System Search - Criteria Object Properties

File

If a search task is defined for a specific file, the root item should always consist two child items: One file and one path, registry, component, or ini item. The file item defines the criteria required to look up the matching file itself, whilst the other item defines where to find the path to the file users want to search for.

Please refer to the *Add a custom search* topic (steps 2a and 2b) for further details about the settings available for target file property definitions.

Path

Please refer to the *Add a custom search* topic (step 3: *Folder location options*) for further details about the settings available for folder related system search criteria.

Registry

Please refer to the *Add a custom search* topic (step 3: *Registry location options*) for further details about the settings available for registry related system search criteria.

Component

Please refer to the *Add a custom search* topic (step 3: *Component location options*) for further details about the settings available for component related system search criteria.

INI

Please refer to the *Add a custom search* topic (step 3: *INI location options*) for further details about the settings available for INI related system search criteria.

> **Be aware:**
> Each system search tasks needs at least one child item to define what exactly the search task has to execute. If there are too many or too few of these child objects defined per system search root task, there may be unforeseen search results during package run-time.

To Add a Child Element to a System Search Task Root Object

Once a system search root element has been created by either adding a predefined or a custom system search, users may add further search term child elements of not already added types to that root element.

1. To do so, users have to **right-click the root** item of the affected search task, and select **Add** from the **context menu**.

2. The **New System Search** wizard is displayed, with the property already present as read-only information derived from the root system search object.

   With a click on the **Next** button, users get the overview of system search types.

3. If the root item has already been extended with a child item of a specific search type, that type is no longer available for an additional selection. This means that each parent search item can have only one child per search type. To proceed, users have to select one of the available search types, and click **Next**.

4. Finish the wizard by entering the requested search criteria as described within the help topic Add a custom system search.

> **WARNING**
> RayPack actually allows to define system search tasks with more or less criteria definition items than recommended. It may be necessary to extend the standard system search operations with more complex structures in order to fulfill more complex searches. However, to provide high-level package quality, any non-standard system search definition needs to be tested thoroughly before it is used in a productive software package.

Installer Options

This view allows you to influence the behavior of the Windows Installer Engine on the target system.

The Installer Options view is separated into two main areas:

- the *Restart manager* on the left
  and
- the interface for defining *Logging options* at the right-hand side

Restart Manager

RayPack allows to determine the value of the `MSIRESTARTMANAGERCONTROL` property by the options presented within this area:

Activate the checkbox **Use the restart manager** to actually take control over the restart behavior of the target package.
Once it is active, the following settings may be defined:

Restart applications that were shutdown by the Restart Manager
If the checkbox for this option is active, a post-installation routine is enabled, that restarts all applications, that have been running on the target machine, but were shutdown by the restart manager during the installation run-time.

If there are applications that need to be shutdown by the Restart Manager, RayPack offers three options for their handling:

- Attempt to shutdown all affected applications
- Shutdown all affected applications
- Shutdown affected applications only if they are registered for restart

Select the option that fits the current packaging project needs, and be sure to test the actual target system behavior for several application constellations.

Logging Options

At MSI package run-time, msiexec is used to launch the package installation. The default command line options for creating and populating log files are internally stored within the property `MsiLogging`, and may be defined by this RayPack dialog.

If the optional MsiLogging property is present in the `Property` table, the installer generates a log file named `MSI*.LOG`. The full path to the log file is given by the value of the MsiLogFileLocation property.

To activate one of the predefined options by the interface provided by the Installer Options view, users have to activate the checkbox left of the option label. If the standard options do not suffice the effective needs, there are further options, which may be displayed with a click on the more options slider.

Once the extended logging options content is visible, there is also a preview field for the effective logging string as it is currently defined by the active options. The interface does not only allow to define that string by checkbox manipulation, but as well by direct manual input. Using the logging options area this way, it is actually possible to easily analyze an existing logging options string by simply pasting it into the Logging string input field, and taking a look at the updated set of enabled and disabled checkboxes above. Any sign or letter that does not trigger one of the listed logging options is automatically cleared from the entered logging string.

Further details about logging options and the `MsiLogging` property are available on *MSDN*.

Administrator Options

This view allows you to set how a package is installed on the target system, and the behavior during installation, repair and de-installation.

Installation Context

The installation context determines whether the package will be installed on user or per-machine basis. It affects the registration information written during installation and the location of various folder, including:

- `Desktop` folder
- `Program menu` folder
- `Start menu` folder
- `Startup` folder
- `ProgramFiles` folder
- `CommonFiles` folder

There are three available options:

- **Per-machine**
  The package will be installed for all users. `Global shortcuts`, `Start Menu`, and `ProgramFiles folders` will be used. This is the default setting for repackaged applications. User installing the package has to have the permissions to write to the global locations. This setting corresponds to the `ALLUSERS` property value of 1.

- **Per-user**
  The package will be installed for the current user. `Private shortcuts`, `Start Menu`, and `ProgramFiles` folders will be used. If the package is not writing to any protected areas, the user installing the package does not have to have administrator permissions. This setting corresponds to the missing value of `ALLUSERS` property.

- **Determine by user privileges**
  The installation context will be determined automatically based on who is installing the package. This settings corresponds to the `ALLUSERS` property of value 2

Reinstall Options

These settings control the way files will be handled in case of reinstallation of the MSI package.

Files (only one of the options can be selected):

- **P** – reinstall if missing
  Reinstalls the file only if it is missing.

- **O** – Reinstall if missing or is in older version
  Reinstalls the file if it is missing or is an older version.

- **E** – Reinstall if missing or is in equal or older version
  Reinstalls the file if it is missing, or is an equal or older version.

- **D** – Reinstall if missing or with different version
  Reinstalls the file if it is missing or a different version is present.

- **A** – always reinstall
  Forces all files to be reinstalled, regardless of checksum or version.

Registry

- **Reinstall all user registry entries**
  Rewrites all required registry entries from the Registry table that go to the
  `HKEY_CURRENT_USER` or `HKEY_USERS`

- **Reinstall all machine registry entries**
  Rewrites all required registry entries from the Registry table that go to the
  `HKEY_LOCAL_MACHINE` or `HKEY_CLASSES_ROOT`

Shortcuts

- **Reinstall shortcuts**
  Reinstalls all shortcuts and recaches all icons overwriting any existing shortcuts and icons.

Recaching Policy

Use to run from the source package and recache the local package.

Reboot Options

These settings control the way the reboot is handled on a target machine

- **Never reboot**
  The installer suppresses all restarts and restart prompts initiated by `ForceReboot` during the installation amd all restarts and restart prompts at the end of the installation.

- **Reboot if required**
  The installer suppresses prompts for a restart at the end of the installation. The installer still prompts the user with an option to restart during the installation whenever it encounters the ForceReboot action.

- **Always reboot**
  The installer will always prompt for a restart at the end of the installation. The UI always prompts the user with an option to restart at the end.

- **Do not display any reboot prompts**
  If this option is checked, no reboot prompt will be shown even if a reboot is required. In that case, the installer will initiate reboot automatically, without waiting for any interaction from user.

Rollback Options

These settings control the usage of Windows Installer rollback feature

- **Disable the rollback**
  If this option is checked, the rollback option will be disabled for the current configuration. The installer is then prevented from generating a rollback script and saving copies of deleted files during the installation.

If the rollback is enabled, the combobox below can be used to select the strategy when there is not enough space on hard drive to initiate the rollback copy:

- **Ask whether to continue without a rollback**
  The user will be prompted that subsequent installation will be potentially unsafe, as no rollback script can be generated.

- **Disable rollback and continue without asking user**
  The rollback will be disabled and the installation will continue without waiting for user interaction

- **Fail with the out-of-disk space error prompt**
  The installation will return error when no rollback script can be saved due to lack of hard disk space.

Admin Properties

This list determines which MSI properties should be treated as "Admin properties". Windows Installer saves their value at the time of administrative installation, and restores these values during actual installation from the admin image.

- Click *Add* to add a new property to the list.
- Click *Remove selected...* to remove the currently selected property.
- To rename the focused property, press `F2` or double click the name to trigger the editing mode. The name of the property must be a valid MSI identifier.

Advanced Mode

Please refer to the following help sections for details regarding the specific view functionality:

- **TABLES**
- **FEATURES**
- **COMPONENTS**
- **CUSTOM ACTIONS**
- **SEQUENCING**
- **UPGRADES**
- **USER INTERFACE**

Tables

The tables editor is one of the center pieces of advanced packaging activity. The editor interface provides a sound set of handy features and helpers besides the plain access to Installer database tables packagers are used to have.
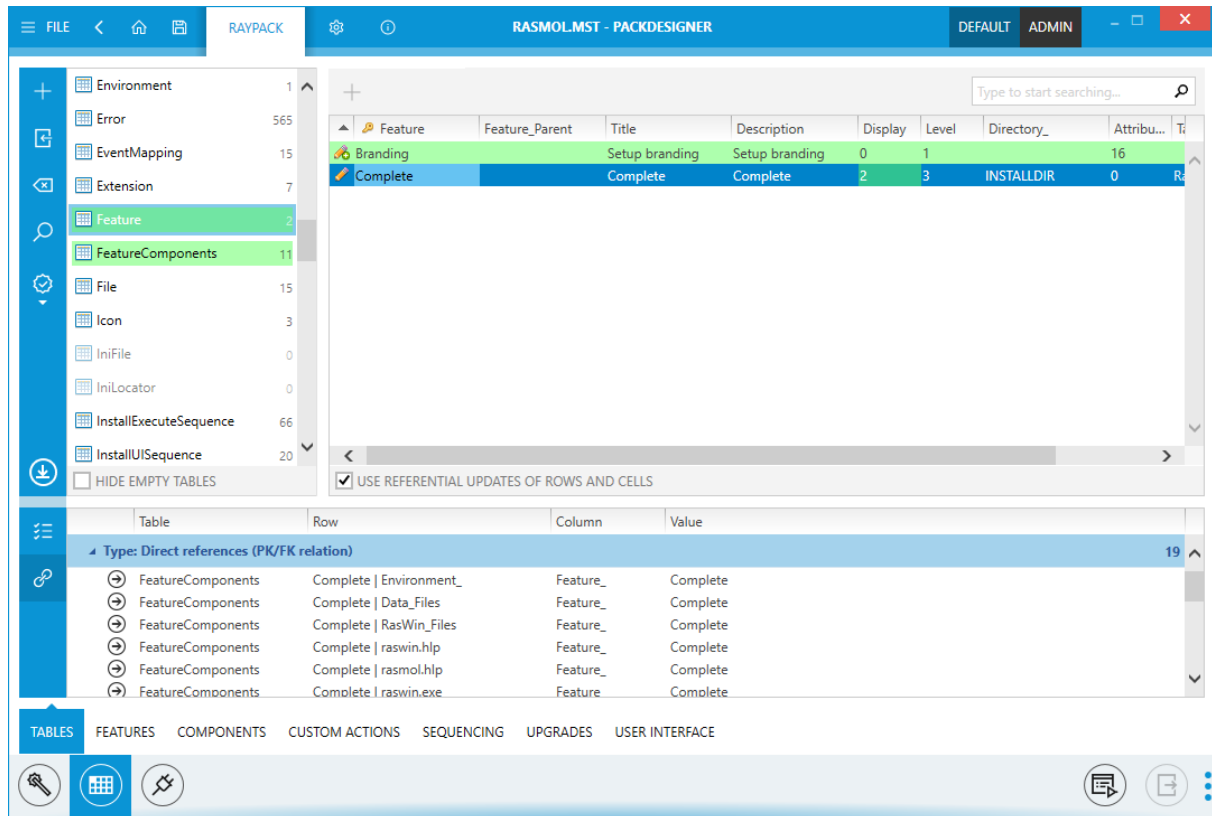
The screenshot above shows the **TABLES** view with the list of available database tables already added to the packaging project on the left-hand side, details of the currently selected table on the right, the action bar above these sections to call core functionality executable on the database, and finally the view navigation to call other views of the Advanced Mode close to the bottom of the application window.

The green and grey highlights displayed above are the result of changes applied to the database contents. Please refer to the help section on *Highlighting & Color Codes* for a detailed description. The grayed out table names indicate, that these tables are empty. They may be *hidden* to be able to focus on active database areas. Icons displayed at the left-hand side of table rows provide quick optical recognition of manipulated table contents, such as edited, added or removed rows.

The action bar above the actual database content allows to:

- *Add tables*
- *Clear the change history*
- *Search and replace within the database*

- *Validate against ICE & Windows Logo Checks*
- *Add rows to the currently selected table*
- *Search the currently selected table*

Installer Database Contents

The primary objective of the tables editor is to provide direct access to the Installer Database contents. The set of tables contained in an MSI based package is defined within the package structure itself. When a new packaging project (`*.rpp`) is created, a standard set of database tables is automatically added and pre-populated with default values.

Beyond the already existing set of tables within a packaging project or an MSI based package that is opened for edition in PackDesginer, packagers may add further tables from a pre-defined stock of standard tables.

The whole set of pre-definitions for database tables can be controlled by RayPack settings profile management. Packagers may edit the template for MSI package generation via the settings section - individually per profile and fully towards their requirements. Please refer to the settings help section to get details on how the applied MSI template may be edited.

Standard MSI Tables

When the RayPack default profile is used, new packaging projects always have the same structure and set of MSI standard tables, either ready for direct editing or for addition from stock.

If opened package has storages, a special table `_Storage` is shown. Note that this table has limited editing possibilities.

Custom RayPack Tables

RayPack requires special database tables to establish functionality such as TXT replacements, source file handling, service manipulation and the like. These custom, non-standard MSI tables are all marked with the table name prefix "RP". Please refer to the help section about *custom database tables* to access information regarding detailed table definitions.

> **Note:**
> Please be aware that, due to ongoing product development, in the future additional custom tables may be added to the already existing stock. Therefore it is highly recommended to name any individually generated Installer Database table without the "RP" prefix.

Main Features Available from the Tables View

- *Database manipulation*
- *Change tracking*
- *Package validation*

- *Search and replace*

Advanced Editor Interface Features

- *Hide empty tables*
- *Highlighting & color codes*
- *Attributes editor*
- *Reference integrity tracking*
- *Input validation*

Database Manipulation

Packagers require full control over the actual database contents of any packaging project. Therefore, database manipulation via direct table access must be available in a simple, yet comprehensive way. An elaborated set of database manipulation options has been molded into RayPack's direct table editor. Besides many advanced features that support the maintenance of data integrity and validity, basic manipulation features, such as adding and removing tables and rows as well as editing of cells, are fully supported.

To Add a Table

Within RayPack, and to be even more precise, within RayPack packaging projects, there are different types of tables available for integration into the active database. Users may manually add a table to an existing packaging project by selecting one of the prepared tables from the pool of generally available tables.

As an alternative, RayPack adds *standard tables* from the prepared stock, or *custom tables* for RayPack specific functionality, right when they are required to implement a specific packaging task, such as creating a *TXT changes* job, or *manipulating a service object on the target machine*.

Note:
Adding new tables, which were not represented in the MSI template that has originally been used to initialize the default data structure during the project creation, is not feasible via the table editor. Changes to the default stock of available database tables have to be prepared in the default MSI template used to initiate packaging projects (or to generate MSI packages from RCP files). Please refer to the help section regarding *profile and template management* to get a detailed description on how to accomplish template preparation.

To Manually Add a Table from the Pool of Template-based Standard Tables:

1. Packagers have to open the **TABLES** view in PackDesigner's Advanced Mode.

2. A list of already added tables is displayed on the left-hand side of the application window. Above that list there is the activity bar for the table editor. Within that bar there is an add button (**+**), which has to be clicked.
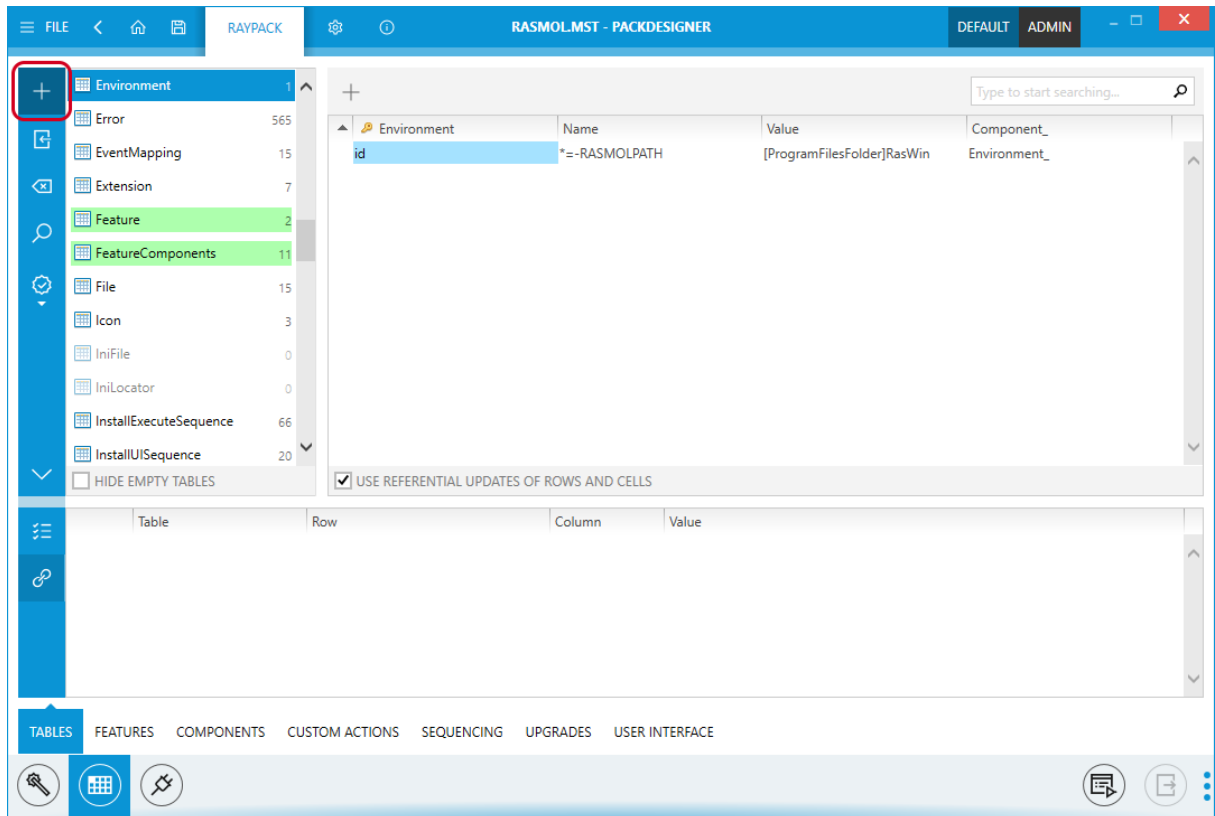
The **ADD TABLE** dialog for adding prepared tables to an Installer database is opened.

3. To select a table from that list, users activate the checkbox on the left hand side of the needed table name.
   The selection may include one or more tables.

4. To create the table(s), users click the **OK** button at the bottom of the dialog window. The marked tables will immediately be added to the project. Additionally, the **TABLES** view is automatically reloaded to show the newly added tables, and the **ADD TABLE** dialog is closed.

   In order to close the **ADD TABLE** dialog without actually adding a table to the project database, the **CANCEL** button has to be clicked.

To Add Rows to a Table

As soon as a table has been added to the projects database, new rows may be added to them directly.

1. First of all, the table that has to be extended with a new row must be loaded into the details pane of the **TABLES** view. To do so, users *click on the table name* from the list of database tables.

   If the name of the desired table is not listed, the table may either be empty and *hidden*, or may not have been *added to the project* yet. Please refer to the specific help topics to find out how to solve these issues.

2. Once the table is selected from the list, its content is loaded into the details pane. For complex tables with voluminous content, the data loading process may take a few moments.

3. As soon as the details pane displays the table content, users may click on the add (**+**) button within the action bar above the details pane.

4.
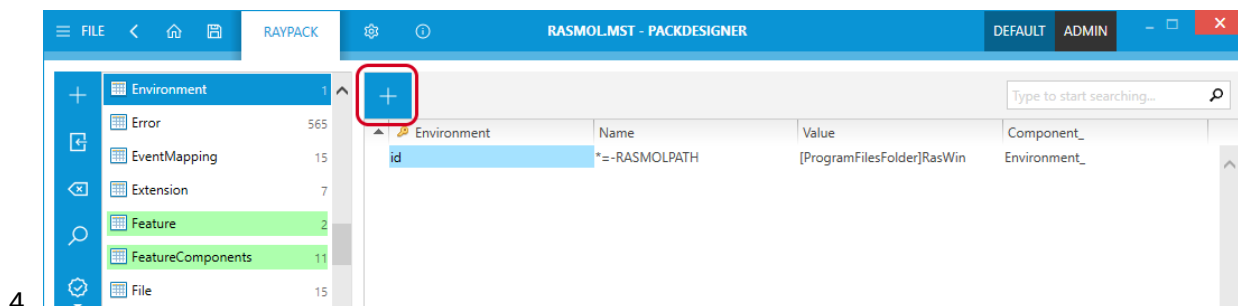
<div align="center">A NEW ROW dialog is displayed.</div>

5. RayPack automatically adjusts the set of displayed input controls to match the column structure of the currently focused table. Adjusting in this case does not only mean to show the right input fields, but also to prepare selector controls for columns that establish relations to other tables (usually marked in the column name with an underscore at the end of the column name, e. g. "Component_" for columns that expect a value that is a primary key of the `Component` table).

6. The labels and expected data types of the table columns are displayed on the left hand side of the **ADD ROW** dialog, the actual input controls are on the right. Inputs that are marked with a red border and background color are mandatory and have to be filled with valid data.

7. If there are missing or invalid inputs, a message bar at the bottom of the **ADD ROW** dialog is displayed, explaining the issue that needs to be solved before the new row can be added to the table.

8. Once all red markers and error messages are gone, users may *add the new row* with a click on the **OK** button, available at the bottom of the dialog window.

To *exit* the dialog without attempting to add the new row, users click on the **CANCEL** button right next to **OK**.

Adding a row to a table is a data manipulation that is temporary until the current packaging project status is saved. Therefore, newly added rows are marked within database tables with a special icon (a pencil with plus sign) and *color coding* (green background color for table and row).

To Copy and Paste Table Rows

Within the **TABLES** editor, RayPack allows to cut, copy and paste single rows of database tables.

To **copy a row** to clipboard, users load the table into the details pane of the **TABLES** editor. With a right-click on any of the displayed rows, the context menu with the copy option is revealed. As an alternative, users may also select a row and use the keyboard shortcut **Control** + **C** to trigger the copy procedure.

To **cut a row** from a table and copy its contents into the clipboard, users load the table into the details pane of the **TABLES** editor.
With a right-click on any of the displayed rows, the context menu with the cut option is revealed. As an alternative, users may also select a row and use the keyboard shortcut **Control** + **X** to trigger the cut procedure.

As soon as a row is copied or cut, users may **paste** it from the clipboard. With a right-click somewhere inside the tables details pane, the context menu with the paste option is revealed. As an alternative, users may also select any of the existing rows within the table and use the keyboard shortcut **Control** + **V** to trigger the cut procedure.

Rows can be pasted back into the original table within the **TABLES** editor:

- If the initial procedure was a copy, a second row will be inserted into the table, matching the properties of the copy master.
- If cut was used to transfer row data to the clipboard, the original row will be replaced by the paste command. This is quite handy if a row has accidentally been removed and has to be restored.

As an alternative, rows may also be pasted into any other file for external usage, e. g. a `.txt` file, or an excel sheet. Pasting into the same table of another packaging project opened in another instance of RayPack is possible as well. However, it is not allowed to copy (or cut) and paste between different tables of a packaging project (e. g. from **File** to **Component**), since the table definitions of these targets are not equal.

To Edit Table Rows

As soon as a table has been added to a projects database and populated with data, users may edit each single cell.

1. First of all, the table that contains the row which has to be edited must be loaded into the details pane of the **TABLES** view. To do so, users **click on the table name** from the list of database tables.

   If the name of the desired table is not listed, the table may either be empty and *hidden*, or has not been *added to the project* yet. Please refer to the specific help topics to find out how to solve these issues.

2. Once the table is selected from the list, its content is loaded into the details pane. For complex tables with voluminous content, the data loading process may take a few moments.

3. In RayPack, users manipulate individual data columns in a direct inline editor. There is no additional dialog that pops out, containing input controls for all columns of the specific table.

   The RayPack table editor interface reacts on double clicks on a specific cell, switching from view into direct inline edit mode. The same can be achieved by clicking a cell and hitting **F2** on the keyboard.

   The current cell content is automatically marked, therefore simply starting to type replaces the existing value. To partially change existing values, users should move the cursor to the desired position first, and then start to enter the additional content. The same should be done for partial deletions. Move the cursor to the right position and hit **Backspace** or **Delete** on the keyboard.

4. Users may hit **Enter** to attempt to save the updated value, or use the **Tab** key to move to the next cell and automatically activate the direct inline edit mode for that one.

   If the changes made to a value are invalid, the associated cell is highlighted with an exclamation mark icon. Hovering over that error icon reveals a tooltip with hints on the issue that prevents the changes from being valid.

   If the changes are actually valid, the modified cell is highlighted with green background color. The table row containing the updated cell is marked with a pencil icon which is displayed left of it.

To Remove Rows from a Table

As soon as a table has been added to a projects database and populated with data, users may manipulate the rows it contains.

1. First of all, the table that contains the row which is about to be deleted must be loaded into the details pane of the **TABLES** view. To do so, users **click on the table name** from the list of database tables.

   If the name of the desired table is not listed, the table may either be empty and *hidden*, or has not been *added to the project* yet. Please refer to the specific help topics to find out how to solve these issues.

2. Once the table is selected from the list, its content is loaded into the details pane. For complex tables with voluminous content, the data loading process may take a few seconds.

3. To remove a row from the currently loaded table, users right-click somewhere inside the desired row, and select **Remove** from the context menu. Of course, selecting the row with a left-click and hitting **Delete** on the keyboard will lead to the very same result.

   It is also possible to multi-select a whole set of table rows and delete them in one step. Keep the **Control** key pressed while clicking on several rows to multi-select them, or use standard

windows selection options such as **Control** + **A** (to select all rows of a table), or **Shift** (to select a group of adjacent rows) to gather the required set of selected rows.

4. Once a table row is removed this way, it is marked as deleted within the table editor interface: It has a gray background color and a pencil with a minus as status icon on the left-hand side. In this state the row is finally deleted from the project as soon as the user closes the project, or *manually clears the change history*.

Until that moment, users may restore deleted rows by right-clicking them, selecting copy from the context menu, and after another right-click somewhere in the details pane of the table, select paste from the context menu. (Left-clicking a row and using the shortcuts **Control** + **C** for copy and **Control** + **V** for paste will work as well) The row is restored, displayed in standard colors and without the removal flag icon.

To Remove a Table

As soon as a table has been added to a projects database, users may remove it from the active pool of tables.

1. First of all, the table that is about to be deleted must be visible in the list of project database tables on the left-hand side of the **TABLES** view.

If the name of the desired table is not listed, the table may either be empty and *hidden*, or has not been *added to the project* yet. Please refer to the specific help topics to find out how to solve these issues.

2. To remove a table from the stock of active project database tables, it has to be *right-clicked* within the listing of database-tables on the left-hand side of the **TABLES** view. The context menu reveals the **Delete** option for tables.

3. Once remove is selected from the context menu, RayPack displays a confirm dialog. Click **YES** to remove the table, or **NO** to keep it as it is.

4. If the user clicks **YES**, the table (and along with it all rows it contains) is removed.

Once a table is removed from the pool of active project database tables, it is available for readding via the **add table** dialog if it is part of the template based stock of prepared database tables for a packaging project. Please refer to the help topic above to get details regarding the *add table* procedure.

	Note:
📎	Please be aware that removing a table cannot be reverted. This means that even if a table will be added to a project again, the data it contained before its removal is and stays gone.

To restore changes to the original state

Once a cell, row or a table is highlighted green, it is an indication that a change has been made and that the original state is stored by RayPack. You can easily revert to the base state by accessing the context menu functionality **Revert to base**. The behavior of this action is different, depending on the context:

- Reverting a newly created row deletes it.
- Reverting a deleted row restores it.
- Reverting changed cells revert their values to the original state.
- Reverting a newly added table removes it.
- Reverting a deleted table restores it.
- Reverting a changed row reverts all changed cells to their original values.
- Reverting a changed table restores all its deleted rows, deletes new rows, and restores the modified row cells to their original state.
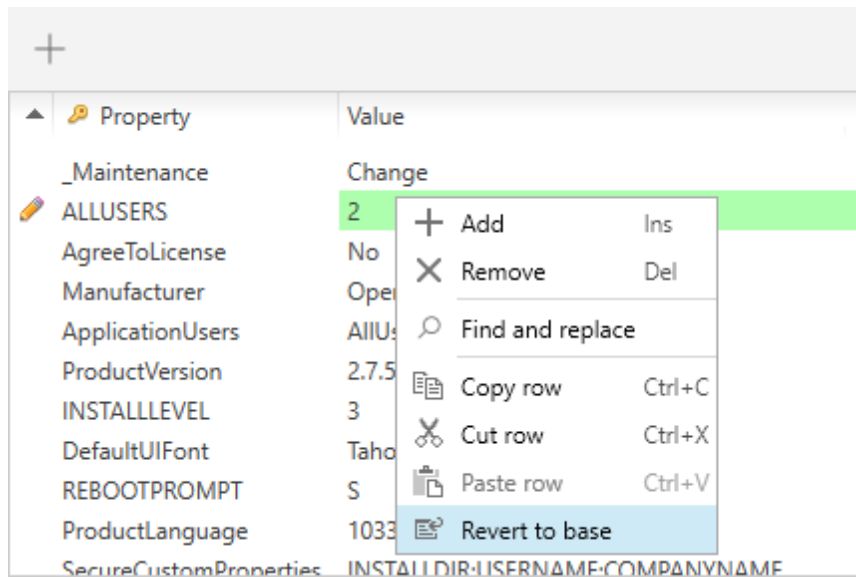
Context menu has an option to revert a cell, row or table to the original state.

Chapter Change Tracking also describes how the change tracking works and how to reset the changes.

To View Help for a Selected Table
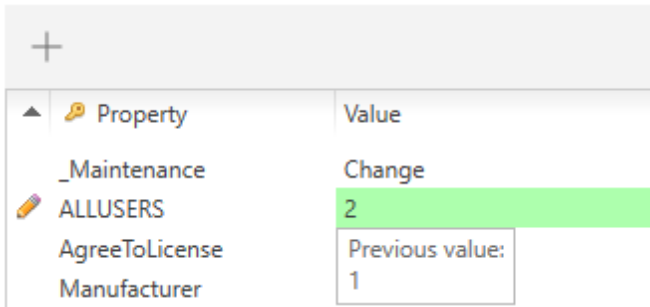
A contextual HTML help for the currently selected table can be shown by pressing `F1` in the Advanced view. This will open an official documentation created by Microsoft and available on MSDN help topic.

Change Tracking

RayPack owns a very elaborate change tracking system for the **TABLES** editor. All manipulations that affect the content of Installer database tables are stored in the temporary change history.

Compared to change history features known from other applications, such as standard office tools, RayPack does not allow to access specific history states per undo or redo. The change history rather allows to add database content highlighting for manipulated contents (see *Highlighting & Color Codes*) during a packaging project work session.

The change history provides access to original values of manipulated table cells. When a user hovers over a cell that is currently highlighted as changed, a tooltip is revealed, displaying the original content of this cell. Original in this case means the content that represents the last permanently saved status of the cell (e. g. the state it had when the project was opened or saved).

Hovering a changed cell reveals the original value.

Clearing the Change History

Clearing the change history may be triggered by several user activities in RayPack:

- **Opening a project**
  When a packaging project is opened (or newly created) in RayPack, the change history is empty by default.

- **Closing a project**
  When a project is closed, the change history is cleared for that project. When it is re-opened, the change history is blank.

- **Manually via TABLES editor interface**
  Within the **TABLES** view, users may click on the **Clear History** button, available in the upper left corner of the application window. Color coding and icon markup are immediately removed from the whole Installer database.
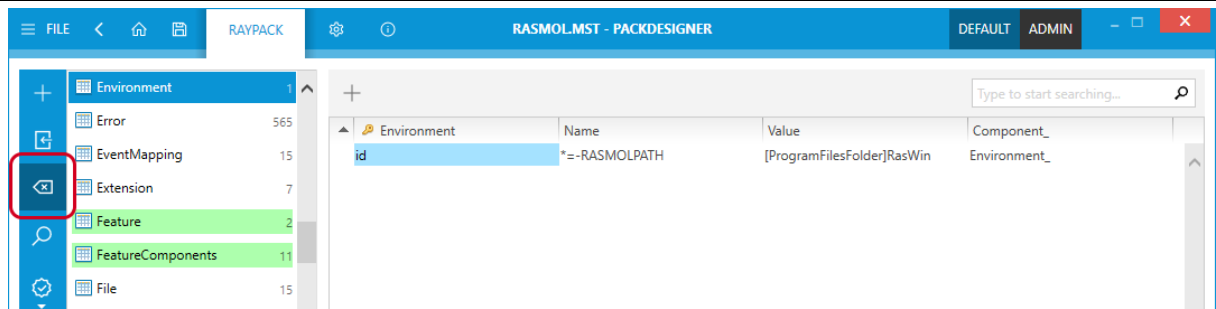
> **Be aware:**
> Clearing the change history does not save the changes, but simply clears the interface from change history markup. This includes hiding old values of updated cells from hover tool tip display, hiding rows that have been marked for permanent removal from display, and the like. Users have to save the changes in order to permanently preserve the current project state.

Package Validation and Testing

This view allows to execute validation rule checks for the currently loaded project. Please note, that this only applies to MSI / MST and RPP based projects.

Package validation includes validation against rules defined by so called Internal Consistency Evaluators (= ICE). It offers full MSI validation as well as Windows Logo checks and Merge Module specific validation. RayPack offers Logo Checks that are specialized for applications targeting Windows XP, Vista and 2000.
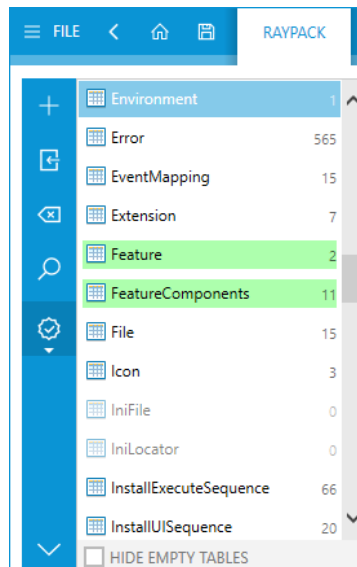
In RayPack, users have access to standard ICE validation and custom ICE validation:

Standard ICE Validation

Standard ICE validation refers to the application of standard ICE rule sets delivered by RayPack, such as Full MSI, Merge Module, Windows XP Logo, Windows Vista Logo, and Windows 2000 Logo. The CUB files containing the ICE rules can be accessed from the `Validation` folder within the RayPack application directory (typically something like `C:\Program Files (x86)` `\RayPack\`).

These sets are by default available in the main list of validation options when the user clicks on the validation button within the `TABLES` editor interface of PackDesigner's Advanced Mode. When a user clicks on one of the entries of this option list, validation according to the ICE rules bundled inside the specific rule set is executed. Please refer to *Working with validation results* for information on how to handle validation results.

Users have several options to modify the actual behavior of standard ICE validation:

- **Ignore ICE rules from the standard validation procedures**
  The settings section allows to edit RayPack profiles. Within the advanced tab of the PackDesigner profile properties section, users may enter a list of ICE rules, packaging projects should not be validated against. If a standard validation rule set contains one or more of the ICE rules that have been added to a profiles ignore list, the standard validation procedure ignores these rules whenever that profile is active.

- **Modify the set of default ICE validation procedures**
  The validation options menu within the `TABLES` editor displays the RayPack standard validation rule sets by default. However, users may manually change the XML file that controls the checks displayed here. To do so, users have to open `ICE.xml` from the Resources folder of the current RayPack installation, usually available here: `C:\Program Files (x86)\RayPack\Resources` `\ICE.xml`.

  The XML file contains a section `<Cubs>`, which lists the RayPack standard ICE validation rule sets. Users may either remove single leaves from this node, or simply change the value for the `IsVisibleInMain` property from `true` to `false`. Either way, the rule set is no longer visible in the main validation rule set list within the RayPack user interface. The set of visible/available Cubs is global, which means that all RayPack users use the list given within the same `ICE.xml` file.

  Another way to modify the standard list, is to add further CUB files to it. Even though CUB files may not be created with RayPack, there are several tools available which can be used for that purpose. Any valid CUB file may be added to the Standard validation list by simply adding a new leaf to the `<Cubs>` node:

  Tag: Cub
  Properties:
  - IsVisibleInMain (true or false)
  - Path (name of CUB file from the default validation source folder without extension or full path to the CUB file)
  - Name (Label for the validation rule set that will be displayed in RayPack)

- Description (Textual description of rule set purpose)

Please note that the Cubs node is read top down for the definition of the Standard Validation Options menu - newly added rule sets will be displayed in RayPack at the very same position the they hold within the `<Cubs>` node.

Each ICE rule may be extended with a description. Therefore, if users manually add further CUB files with individual ICE rule index values, the list of ICE rule descriptions can actually be extended as well. For each ICE rule, there is the option to add a leaf to the `<Description>` node of the `ICE.xml` file:

Tag: Rule
Properties:
- Id (the unique identifier of the rule, e. g. "ICE999")
- Description (A textual description of the issue checked by the specific rule)

Note:
Changes to the `ICE.xml` should not be made while the RayPack application is running. Please close an open RayPack project and exit the application before the `.xml` configuration resource is edited. Finish the changes and launch RayPack to see the changes take effect on the validation behavior and user interface.

- **Modify the CUB files of standard validation procedures**
  Even though it is not recommended to do so, packagers may edit the CUB files delivered along with RayPack, and directly manipulate the scripted logic behind each and every single rule. Since this kind of manipulation is neither visible in RayPack nor noticeable for other packagers working with the same packaging machine, this kind of manipulation should be handled with explicit care and consideration. If a standard CUB has to be changed, it would be better to make a copy of the RayPack original CUB, hide the original from display (as described above), and integrate the manipulated copy instead. By giving the individual copy a special, descriptive name, all users will have the chance to recognize that the ICE rule set they are currently working with is not the same as the one delivered by RayPack in the first place.

- **Change the actually validated content source**
  Whenever a user works on an RPP file and triggers custom validation, he is able to select which source RayPack will actually use for the validation run:

  o **Current Project**
    The actual Installer database contents of the currently opened RPP file are validated. For most purposes, this quick option should be sufficient, since the structure of the RPP database and a later MSI target package database are almost the same

  o **Built package**
    RayPack generates a temporary MSI from the currently opened RPP on the fly. By validating this temporary MSI, packagers can be sure that the analyzed structures are the same as of an MSI that would be generated from the RPP for QA or deployment reasons. For example, there are authentic values given within the Media table, which is not necessarily the case for validations of of the current RPP.
    Since building a temporary MSI may take a few moments, it is recommended to go for

current project validations first and switch to built package validations later, after the quick validation procedures have been solved.

o **Custom MSI**
Using this option requires to pick an already existing MSI file from the local file system. The contents of the MSI installer database tables will be loaded into a temporary session clipboard, but not directly within the RayPack application. Therefore, when the validation has finished with errors or warnings, it is not possible to jump directly to the reason for the issue, since the validated MSI source is not opened in RayPack. However, it might be necessary to validate any given MSI file, e. g. to compare validation results, or verify that a specific file has been built correctly.

Custom ICE Validation

Besides the Standard ICE validation options outlined above, RayPack offers an interface for totally custom ICE validation procedures. To call the dialog for custom validation:

1. Users open the ICE validation options menu from the `TABLES` editor in PackDesigner's Advanced Mode with a click on the validation icon in the upper left area of the application window.

2. The last item from the now visible option list is **Custom and advanced validation** - clicking it displays the custom validation interface.

Within the custom validation interface, users may load custom CUB files from a local or shared directory resource, or customize the set of active ICE rules for a given standard rule set. Custom settings to the list of applied and ignored ICE rules last as long as the current instance of the RayPack application runs. Once RayPack is closed and relaunched, the definitions made within the active settings profile are applied.

To load a custom CUB file, users click on the browse button [**...**] at the upper right area of the custom validation dialog. Once a CUB is selected from the system explorer, it is automatically added to the list of selectable rule sets prepared in the selector control on the left hand side of the browse button.

When a rule set is selected, the list of ICE rules it contains is loaded into the details pane below. All rules with an active checkbox to their left are used when the **VALIDATE NOW** button is clicked. All rules that do not have an active checkbox are ignored. To make life a bit easier for RayPack users, the interface contains **ALL** and **NONE** buttons, allowing to select or deselect the whole set of rules currently displayed in the details pane.

Clicking on **VALIDATE NOW** executes validation according to the list of currently active ICE rules for the selected rule set. Please refer to *Working with validation results* for information on how to handle validation results.

Working with Validation Results

Once RayPack has executed any standard or custom ICE validation procedure, a validation result panel becomes visible, displaying any issue (*warning or* error) that arose. If all rules reported

successful tests, the result panel is not displayed at the end of the validation procedure.

Hiding / Showing and Resizing the Validation Results

The validation results can be temporarily hidden / shown by pressing the *collapse / expand* button:

The bottom panel can be resized by dragging and dropping its upper separator between the tables and the panel itself.

Since the bottom panel is also used to display the *row reference tracking* results, the following buttons can be used to enable / disable the validation results:

Shows ICE / test results

Hides ICE / test results and shows row tracking results.

Troubleshooting Validation Results

With a double-click on any row of the validation result table, the **TABLES** editor is updated to load the contents of the affected table and focus the specific cell that caused the issue report.

Besides the direct result linking, RayPack supports packagers in their aim to readjust contents that caused validation issues with a special highlighting scheme for tables and cells: Red background for table names or cells indicates that an error was reported, whilst yellow background indicates a warning. Please refer to the *Highlighting & Color Codes* topic for further information on highlighting schemes used within the TABLES editor view.

The Validation Result Table

No matter if the result table is displayed in the *integrated* or *external* panel dialog, it always contains the same set of columns:

- **Type**: Result items may either be a warning or an error. The type column contains an icon that indicates what kind of severeness the result item has.

- **Source**: The ICE rule that reported the issue

- **Table**: The table that was analyzed when the issue was actually found. Since some issues depend on relation integrity, the table named here may differ from the background-information given within the description column.

- **Keys**: The key value(s) of the row that was analyzed when the issue was actually found. Since some issues depend on relation integrity, the key named here may differ from the background-information given within the description column.

- **Column**: The name of the column that was analyzed when the issue was actually found. Since some issues depend on relation integrity, the column named here may differ from the background-information given within the description column.

- **Description**: A textual description of the report reasons. Please read the text carefully to get hints on how to solve the object constellation which caused the report.

Validation Result Action Options

Sort Validation Results

The table that contains the ICE validation results may be sorted by each of the visible columns either ascending or descending. To sort the result according to the values displayed within a specific column, users simply click on the header of the desired column. An arrow icon indicates the current direction. To switch between ascending and descending direction, another click on the very same column header is executed.

The sorting algorithm allows simple ordering, it is not possible to stack sort orders for several columns as multi-level sorting.

Filter Validation Results

The column headers of the validation result table contain cone icons. With a click on that icon users may filter the rows of the result table to display only those with a specific content. The option list displayed in the filter menu shows all varieties of content values plus the option to show only rows with blank cells in this column, or to de-activate the filter and show all columns again.
Filters can be stacked by simply selecting required values for several columns at the same time. To do so, the result pane has to be popped-out for advanced sorting.

> **Note:**
> The features described below can only be triggered when the result pane is popped-out as a separate dialog.

Advanced Sort and Group for Validation Results

To group validation results by one or more specific column types, users have to activate the advanced sort mode to display the group bar. Once the group bar is visible, column headers may be dragged there from. As soon as the column is dropped inside the group bar area, the table contents are automatically and immediately grouped. Each value group is summarized by a header row within the tables data section. These header rows are collapsed by default. To expand them, the arrow icon on the left of the header label has to be clicked. Clicking it again collapses the section again. By dragging further column headers to the group bar, arbitrary combinations of multi-level grouping are available.

Whilst columns are used for grouping, it is possible to sort for each column ascending or descending. Therefore, in the advanced mode, multi-level sorting is available. It is also possible to activate the column based filter to reduce the visible data rows to a closely focused work set.

The advanced group and sort criteria has to be removed manually. To do so, users have to drag the header blocks back into the original table header bar. The column is displayed wherever users drop them, which allows to individually adjust the table structure.

Print Validation Results

Within the activity bar of the popped-out validation result panel, users have access to the print button. Clicking it immediately opens the print dialog with available print options for the current packaging machine. Please configure the print procedure as required.

Export Validation Results

Validation results are not permanently available in RayPack. Once the result pane is closed, former result lists are no longer restorable. Users have to execute the validation procedure again. If any changes have been applied to the packaging project contents since the last validation run, the new results may differ from the former ones. Therefore, exporting validation results is required for later reference.

Within the activity bar of the popped-out validation result panel, users have access to the print

button. Clicking it opens a system explorer dialog. Please select the desired target location, file name and type for the data export. Available target formats are XLS(X), HTM(L), PDF, and CSV / TXT. Each target format has benefits towards specific data handling requirements: Whilst PDF is quite handy for transferring human readable information, other target formats are better fits for analysis and data transfer between applications.

Search Within Validation Results

Within the activity bar of the popped-out validation result panel, users have access to the control for keyword search procedures. At the upper right corner of the result pane, users enter one or more keywords into the input field. The keyword filter is applied immediately, so that entering one letter after another reduces the displayed result set step by step. To display the whole result set again, simply remove the keyword from the input field.

Keyword search and advanced sort & group features may be stacked to create the optimal result set for the current task context.

Display Modes of the Validation Result Pane

The result panel is by default displayed in its **popped-in** mode (see screenshot below). This mode adds a third information area to the **TABLES** editor interface, allowing to display validation issues, the list of packaging project tables, and the content of a specific table at the same time within the same application window.
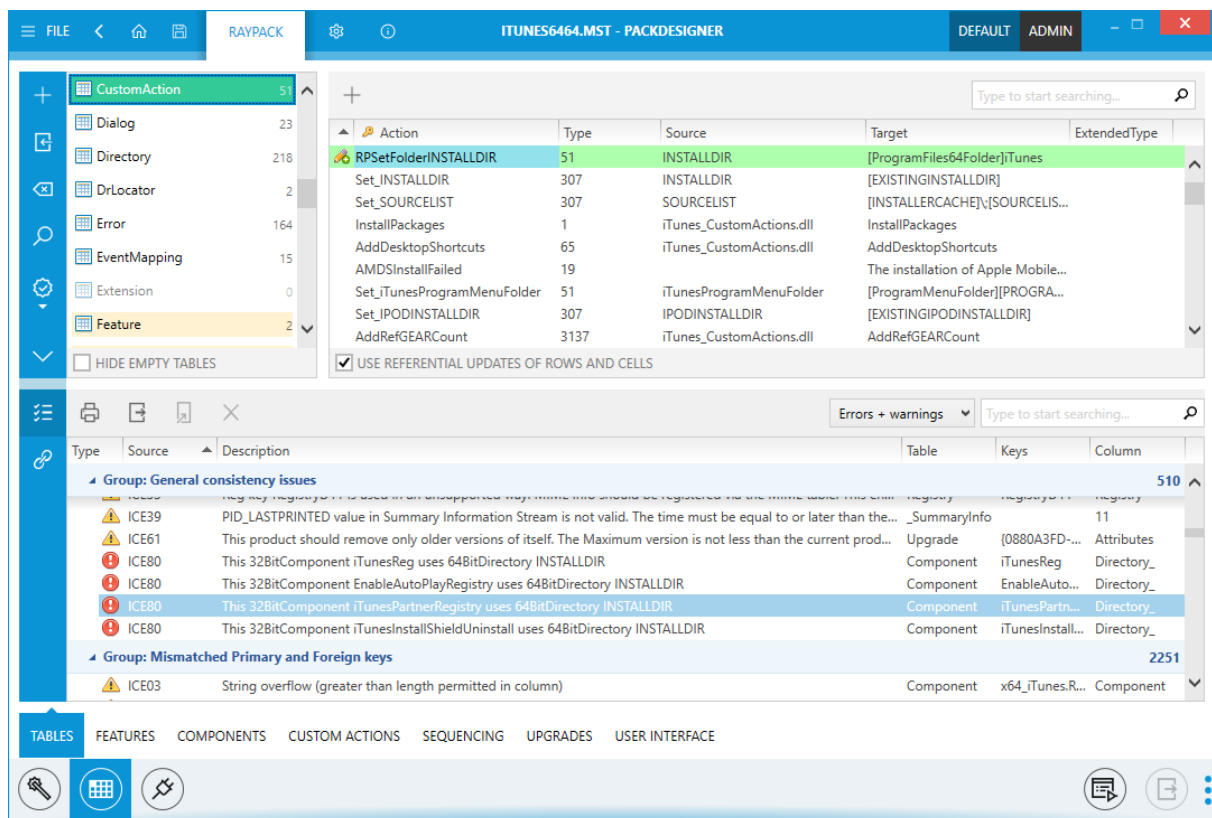
Especially for packagers who work with several monitors, it is quite handy to **pop-out** the ICE results panel. Using this display mode, results may be used as additional reference, but do not block screen space required to gain an overview on the database and table content of the currently analyzed project.
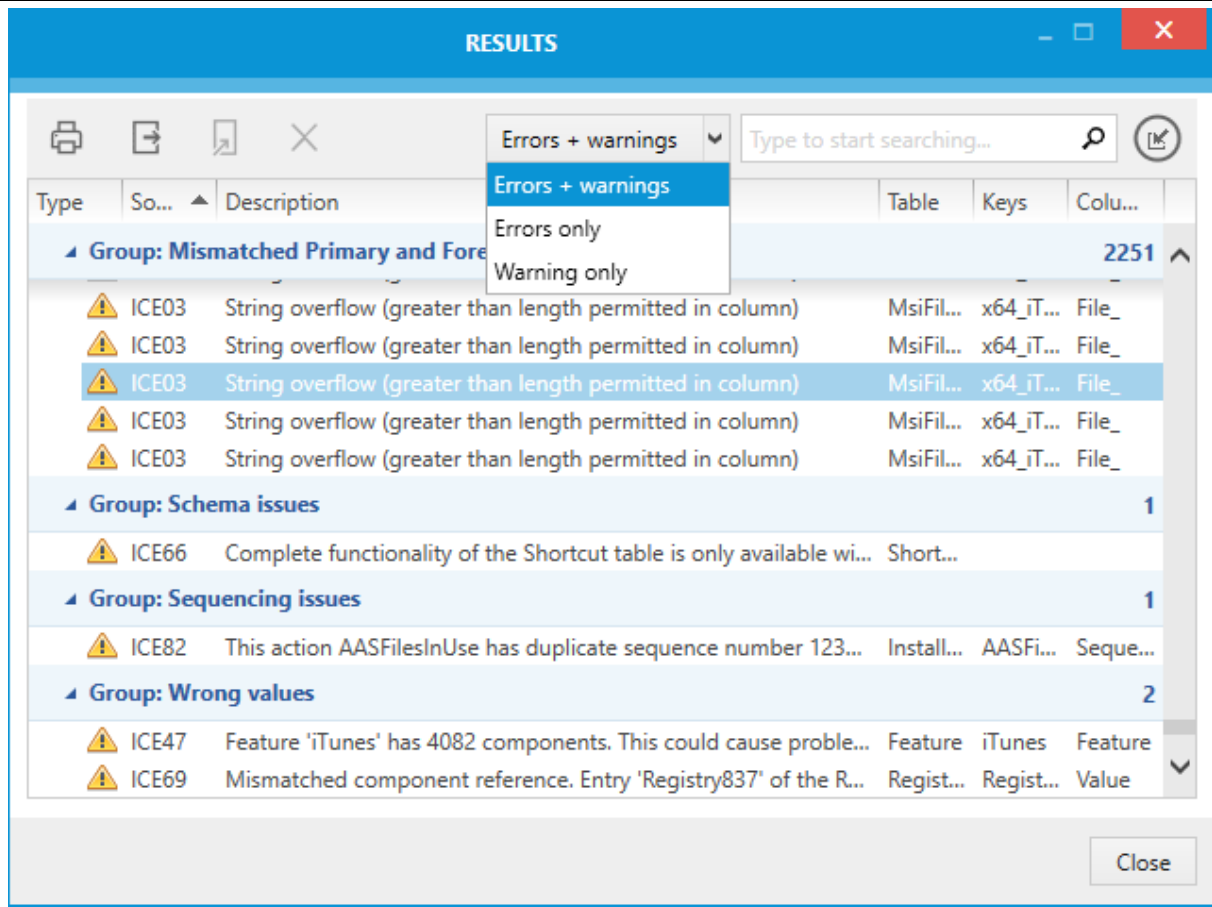
To close the validation result pane, both view modes contain a **CLOSE** button right next to the button for view mode switching.

Exceptional Validation Results

Whenever RayPack encounters issues with the initiation of the validation process in general or the execution of particular ICE rules, the result table contains error messages. This is a difference to the standard defined for ICE validation, since RayPack does not label these issues as failures but errors. However, issues that prevent ICE validation from execution are usually caused by structural deficits of a package or packaging project. Severely damaged database tables may be the result of incomplete file transfers or storage errors.

Further possible reasons for failed ICE validations are invalid CUB files. Especially custom validation rule sets that have been imported to RayPack may cause issues if they do not conform to the CUB file format standard.

Clearing Validation Results

In order to remove the highlighting and ICE validation results, click on the *Clear the results* button:

Search and Replace

Complex application packages and packaging projects may contain substantial sets of tables with potentially voluminous data contents. Manually skimming through tables and rows may surely lead to the desired data object, but it is inconvenient and time-consuming. To support packagers in their requirement for quick and reliable data handling, RayPack's **TABLES** view offers a comprehensive set of search and replace options:

Search Within a Single Table
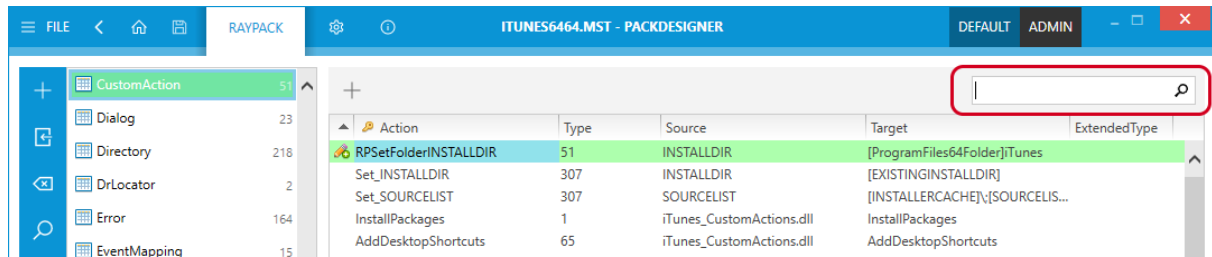
Within the activity bar of the table details pane, users have access to the control for keyword search. At the upper right corner of the details pane, users can enter one or more keywords into the input field. The keyword filter is immediately applied to the currently loaded database table, so that entering one letter after another reduces the displayed row set step by step.

Wherever the keyword is found within the table data, it is highlighted. Therefore it is quite easy to quickly find data objects required for the current task scope. To display the whole table content again, simply remove the keyword from the input field.

This simple search is designed to filter the contents of the currently displayed database table. To search the whole database, RayPack offers an advanced search feature which is outlined below.

Advanced Search Options
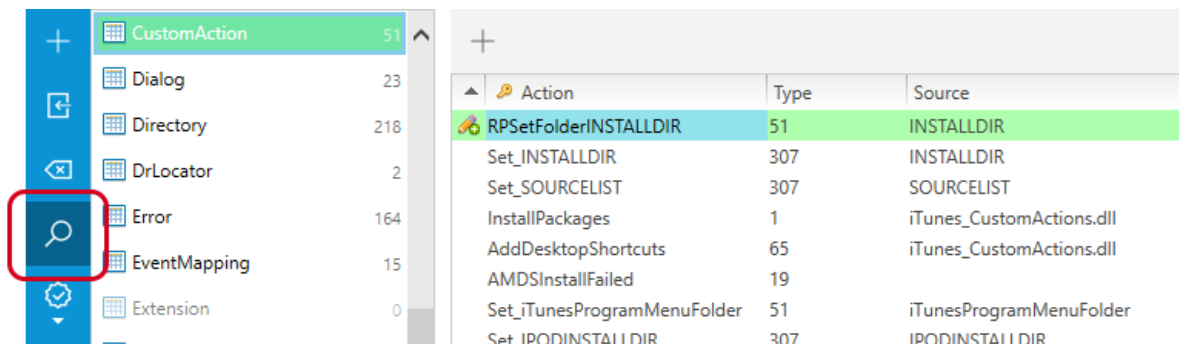
To access the dialog for the advanced database search options, users may either click on the search button available on the left hand side of the **TABLES** view or simply use the keyboard shortcut **Control + F**. The dialog is opened with an activated **FIND** tab containing an input control for the keyword that has to be searched and checkboxes to define whether or not the search procedure should match the case of the keyword and the database value and whether or not only whole words should be included in the result set.

Once these search procedure options are defined, users have to click on the **FIND NEXT** button to initiate the search. Clicking it again navigates to the next matching database data object property. Returning to the full database content display is achieved by closing the advanced search dialog with a click on the **CLOSE** button.

Besides the basic database search options, further controls for fine-tuning the procedure are available: Users have to expand the dialog by clicking the more button at the lower left corner of the dialog.

Regular Expressions

Keywords are by default not handled as regular expressions. To alter this standard method, users have to activate the use regular expressions checkbox, which is available in the expanded search database dialog. Once the checkbox is activated, RayPack resolves keywords as regular expressions, allowing advanced pattern searches for a maximum of search effectiveness.

Narrowing the Scope

Keywords are by default searched within the whole packaging project database. When the advanced search dialog is expanded, users may alter this standard procedure and select a specific table and even narrow the relevant search scope to a specific column. If the controls for table and column selection are used at the same time, the search algorithm is applied to the values stored within the selected column of the desired table. If the column selector is used without a specific table selection, the keyword is searched in all columns with the selected name - no matter what table contains the column. This option is quite handy to get an overview of key value occurrence throughout the whole database.

Replace Database Contents

Sometimes it is required to globally modify specific phrases and terms used within a packaging project database. With the database-wide replace procedure, RayPack contains a mighty little helper to accomplish such tasks. To define the criteria required for replacement tasks, users have to open the advanced database search dialog, and switch from the **FIND** to the **REPLACE** tab. The following controls have to be adjusted before the replacement procedure may run successfully:

- **Find**
- **Replace with**
- **Match case**
- **Match whole words**
- **Use regular expressions**
- **Table**
- **Column**

With the exception of the **Replace with** control, all required settings are already described in the *Advanced Search* section above. The text given as replacement will totally replace the original phrase that matched the search criteria. It is not possible to define options such as prepend or append, the replacement always includes the whole matching phrase.

Once the search criteria are defined, users click the **FIND NEXT** button to start the initial search algorithm. When a matching cell is found, it is possible to either use the **REPLACE** button to explicitly replace the current match or to use **REPLACE ALL** to automatically replace all matches without additional visual confirmation and manual replacement triggering. Each click on **FIND NEXT** allows to navigate to the next matching database cell, no matter if the previous one was actually replaced or not.

> **Be aware:**
> Skipping the time-consuming but safe procedure of **FIND NEXT** and **REPLACE** repetitions has to be handled with care - especially for those replacement procedures that base on complex regular expression searches. Depending on the accuracy of the expression phrase, the replacement might affect database contents that were not targeted intentionally and therefore cause serious damage to the correctness and validity of a packaging project database.

Advanced Editor Interface Features

As outlined above, RayPack includes the most modern and intuitive direct table editor interface packagers might think of. The following topics provide detailed descriptions of the most important advanced table editor features:

- *Hide empty tables*
- *Highlighting & Color Codes*
- *Attributes Editor*
- *Conditions Editor*
- *Reference Integrity Tracking*
- *Input validation*

Raynet strives to establish the best packaging framework on the market. Therefore, providing a best-of-breed direct table manipulation interface is one of our essential ambitions. If you would like us to add a specific feature to the **TABLES** view - please let us know!

Row Tracking

The row tracking mechanism displays and tracks the references between selected row and other cells / rows / tables in the edited MSI / RPP / MST project.

In order to enable the Row Tracking view, activate the bottom panel in the Tables View by using the following button:
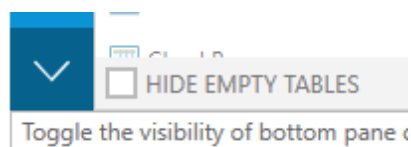
Since the bottom panel is also used to display the ICE / test results, the following buttons can be used to enable / disable the row reference tracking results:

Hides ICE / test results and shows the ICE / test validation results

Shows row reference tracking results

The row tracking is a list of rows that rely or that are referenced by the currently selected row. For example, with the following row in the `Registry` table selected selected:

Registry	Root	Key	Name	Value	Component_
ProductVersion	2	Software\Raynet\RayPack	ProductVersion	[ProductVersion]	Registry_HKLM
InstallDir	2	Software\Raynet\RayPack	InstallDir	[INSTALLDIR]	Registry_HKLM
PackPointDir	2	Software\Raynet\RayPack	PackPointDir	[PACKPOINTDIR]	Registry_HKLM
ProductName	2	Software\Raynet\RayPack	DisplayName	[ProductName]	Registry_HKLM

the following references are shown in the Row Tracking panel:

	Table	Row	Column	Value
Type: Direct references (PK/FK relation)				
Component	Registry_HKLM		Component	Registry_HKLM
Type: Indirect references (formatted string)				
Property	ProductVersion		Property	ProductVersion

Which should be interpreted as:

- The row `ProductVersion` in the Registry table has a reference to external table. The referenced table is `Component`, the identifier of referenced row is `Registry_HKLM`. and the value references the `Component` column.
The value that references the row is `Registry_HKLM`.

    In fact, the column `Component_` in the `Registry` table is a Foreign Key to the `Component` table, and therefore link is shown by RayPack.

- The row `ProductVersion` in the `Registry` table relies on indirectly referenced property `ProductVersion` from the `Property` table, in the `Property` column.

    In fact, the formatted string [ProductVersion] contains a valid reference to Windows Installer property, and therefore the link is shown by RayPack.

RayPack shows the following references:

- Primary Key / Foreign Key relations based on the Validation table
- Primary Key / Foreign Key relations based on internal database of popular references
- Formatted strings using file, property, folder or component syntax

The information is shown for both row that is a source of reference and the target of reference as well.

Jumping to the Reference Source / Target

In order to jump to a reference source / target, double click the entry in the Row Tracking panel. For example, double clicking the first entry in the example above would jump to the `Component` table, to the `Registry_HKLM` row and highlight the `Component` column.

References and Cascade Updates

When a value in a cell is changed, RayPack tracks down all target references and updates the value accordingly. For example, changing the identifier of the `ProductVersion` property to `ABC` would rename the registry value from `[ProductVersion]` to `[ABC]` (see the example above).

The values in tables are replaced recursively, which means that all changed cells are scanned for any references that may require update and so on. This ensures that the package is internally consistent regardless of how deep and scattered the references are.

Similarly, removing a row would remove or change the other values of connected cells. For example, when an entry is removed from the `Component` table, connected entries in the `FeatureComponents` will be removed as well.

Example:

If the referenced column is marked as `nullable` and the referenced row is deleted, RayPack will set the value of the column to `null` instead of removing the column. For example, when a file `ABC` is removed, its `MsiFileHash` and `RPSourcePath` rows will be removed as well. However, the component that references that file via `KeyPath` column will not be deleted, because the `KeyPath` column in the `Component` table is `nullable`. This means that the row reference tracking mechanism will only set the value in the KeyPath column to `null`.

> **Note:**
> While replacing identifiers of formatted strings RayPack applies a special handling to the `INSTALLDIR` references. Any formatted reference (for example `[INSTALLDIR]` in the Control table) is left unchanged. This is by-design to avoid some unexpected behaviors.

Disabling / Enabling Cascade Updates

The automatic cascade updates can be disabled by unticking the checkbox beneath the tables:

The mechanism can be re-enabled at any time in the future.

Hide Empty Tables

The **TABLES** view displays all table entities that are currently available within the opened package or packaging project. Since databases may contain a broad set of tables, the list tends to be quite long and therefore sometimes lacks a bit of clarity. In order to allow packagers to focus on the stock of tables that is actively used and populated with actual content, the **TABLES** view is equipped with a function to hide empty tables. (Analog to the likewise feature for directories, for

example within the Files & Folders view of the Visual Designer Mode.)

Empty tables are marked by a lighter grey font color and a zero in the right column of the table list, indicating that there are no rows stored inside of it.

To Hide Empty Tables

1. Within the **TABLES** view, users find the list of available tables at the left-hand side of the application window.

2. Underneath this list view there is a checkbox called "**Hide empty tables**".

3. **Activate** that checkbox to immediately hide empty tables from view.

To Show Empty Tables

1. Within the **TABLES** view, users find the list of available tables at the left-hand side of the application window.

2. Underneath this list view there is a checkbox called "**Hide empty tables**".

3. **Deactivate** that checkbox to immediately show the empty tables within the list view of database tables.

Highlighting and Color Codes

The **TABLES** view provides a wholistic interface for direct table manipulation. In order to benefit from the advanced options, such as *search and replace algorithms*, *ICE validation*, and the like, packagers have to get used to the highlighting and color code schema applied to database contents in a specific state.

Highlighting is automatically executed whenever a user *manipulates*, *searches*, or *validates* the database contents:

Database Manipulation Highlighting

Whenever the content of any database table is affected by user activity, a specific set of highlighting procedures is applied:

Adding tables

Whenever a table is added to a packaging project, the item that represents the table within the listing of database tables within the **TABLES** view is marked with green background color.

However, if a new table is added manually, it is empty by default. Empty objects are usually displayed in a grayed out font style within RayPack, and the table listing actually follows the same schema. Therefore, the table name itself as well as the green background color are faded out for new, empty tables. Please take a look at the screenshot on the right - the `ServiceInstall` table has been added manually, is still empty, and therefore displayed in the faded style.

RegLocator	12
RemoveFile	35
RemoveRegistry	18
RPTextReplacements	1
SelfReg	1
ServiceControl	2
ServiceInstall	0
Shortcut	2
Signature	2
SxsMsmGenComponents	3

The `RPTextReplacements` table, which is highlighted in the screenshot on the right, has also been added to the packaging project. It has not been added manually, but automatically as the result of the creation of a new TXT Replacement object within the Visual Designer. This table contains a row (as indicated by the 1 in the right column of the table listing), and is therefore displayed in full text and background color intensity.

Adding rows

When a row is added to a table, the table name within the database table listing is marked with green background color (as described above for the `RPTextReplacements` table). Additionally, the new row inside the table itself is marked as well: by green background color and a special icon on the left hand side of the row inside the details pane, visible when the table content is loaded to be displayed.

Changing row content

When the content of a specific cell is edited, the cell background is changed to green. Additionally, the parent row is marked with a pencil icon on the left. Special markup combinations may occur when new rows are edited: In this case the manipulated cells have a darker gray background then the ones that have not been modified since the row was originally created.

Removing rows

Removing a row may either be executed as immediate delete procedure. In this case it is immediately erased from the database, and no traces are left. This is, for example, the standard procedure for data objects such as resources, that have been added to a packaging project during the current work session. If on the other hand an object is removed, that has existed within the packaging project before it was opened for the current work session, RayPack does not display it within the Visual Designer interface any more, but keeps the database contents available for restoring until the project changes are saved. These rows are marked with gray background color, and with a pencil icon on their left which is extended with a minus symbol.

Note:
The described highlighting methods can actually stack, resulting in database table rows that are grayed out due to their removed state, but at the same time contain green cells, since these were manipulated by a separate activity step. Usually it is the last activity that determines the actually displayed row icon.

How to Remove these Manipulation Markers

Manipulation markers may be removed by manually triggering the change history clearing procedure. To do so, users click on the clear history button, available on the upper left area of the TABLES view. The content markup is reset automatically whenever a packaging project is closed.

Database Search Highlighting

Keyword Search for Single Tables

Whenever a user enters text into the input field for table-based keyword search, the currently loaded database table is automatically searched for the current phrase. The displayed set of table rows is immediately reduced to those that contain the search phrase in any column. In order to easily recognize the matching cell, RayPack marks every matching occurrence of the keyword with yellow background color.

Advanced Keyword Search for Whole Databases

Searching for keywords throughout packaging project databases does not create a complete set of search results, but shows each single match in turn. When users click FIND NEXT, the next match is displayed, showed in the full context of the parent table. To distinct advanced search results from results found by table-based simple search procedures, the results of advanced procedures are highlighted differently:
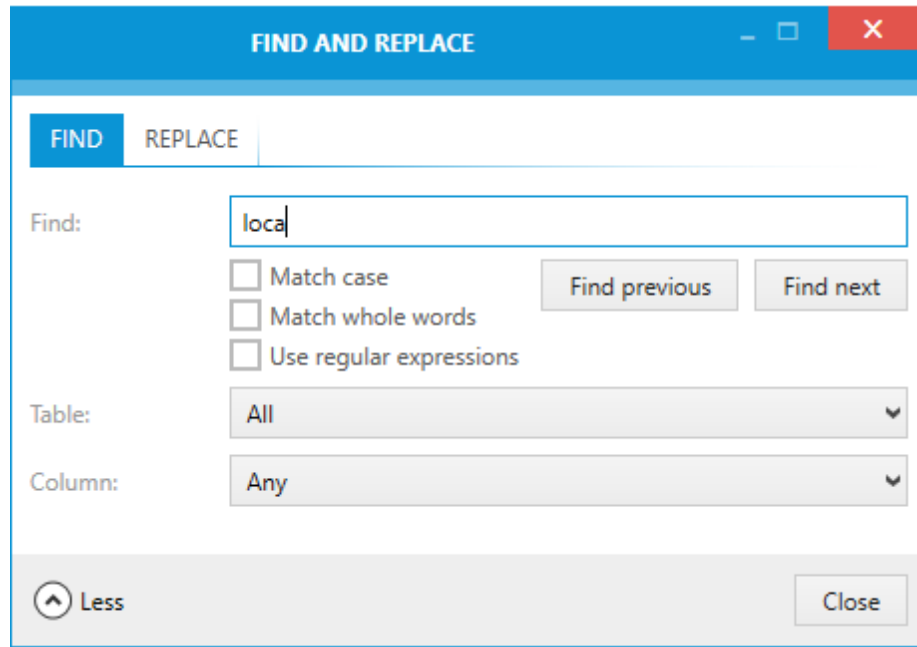
The screenshot above shows a match found in the `UIText` table. The matching row is marked by blue background and white font color, whilst the cell that actually contains the match is marked with dark font and darker blue background color. The name of the parent table within the database table list on the left-hand side of the **TABLES** view is highlighted with a likewise color scheme: white font and blue background color. Since the user needs a clear optical reference to the actual match focused as a result of clicking **FIND NEXT**, only the current advanced search result is highlighted. (Unlike the markup for simple search procedures, which highlights all matches at the same time.)

How to Remove these Search Markers

Search markers are automatically removed when the triggering search is deactivated:

- For the simple, table-based search this is done by removing the keyword from the input field on the upper right corner of the table content details pane.
- To exit the advanced search, users click on the CLOSE button at the lower right corner of the advanced search dialog.

Database Validation Highlighting

Applying ICE validation on packaging projects may lead to the conclusion that a packaging project is valid towards a specific set of ICE rules. In this case RayPack does not apply any validation related highlighting to the database. However, a usual validation result contains at least some warning reports.

The highlight style depends on the type of validation message:

Warning
- Tables containing rows that caused warning results are marked with **yellow** background color in the list of available database tables on the left-hand side of the **TABLES** view.

- Rows containing cells that caused warning results are marked with **yellow** background color when they are selected for display in the details pane on the right-hand side of the **TABLES** view.

Error

- Tables containing rows that caused error results are marked with **red** background color in the list of available database tables on the left-hand side of the **TABLES** view.
- Rows containing cells that caused error results are marked with **red** background color when they are selected for display in the details pane on the right-hand side of the **TABLES** view.

> **Note:**
> Since validation and manipulation highlighting schemes stack, the actually visible effect of the various highlight layers may differ from the description outlined above. For example:
> A combination of red background color for ICE error markup and green background color for cell manipulation markup leads to an actual light shade of brown for the affected cells.

How to Remove these Validation Markers

The validation result highlighting scheme is active for as long as the validation result itself is displayed. As soon as the user closed the validation result pane, the database is set into the standard color coding highlight mode with markup for manipulated contents.
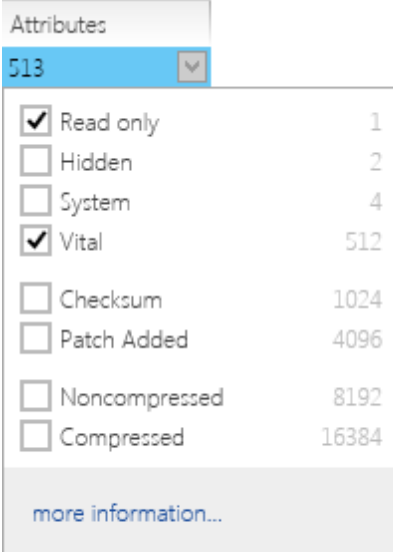
Attributes Editor

Several tables within standard Installer databases contain Attributes columns. The actual value of these columns is created as the integer sum representation of bit wise interpretations regarding set or not set properties. This schema does not only sound complicated, it actually is required to have years of packaging experience to memorize all available options and combinations that may be defined.

In order to allow packagers to quickly set correct values for vital Attributes columns, RayPack contains the so called Attributes editor. It becomes visible whenever a user:

- double-clicks any cell with an Attributes column type.
- clicks on a cell with an Attributes column type to activate the direct inline edit mode, and then expands the editor dialog with a click on the downwards arrow, which is now displayed on the right-hand side of the cell.

The screenshot on the right shows the Attributes Editor dialog for the `File` table. It contains each allowed bit value as a selectable option, along with its numeric value and a short description of the property controlled by each specific bit value. The checkbox in the left column of the editor interface is used to select actively used bit values: The numeric value of the Attributes cell is the sum of the numeric values displayed in the right column of those items, that have an activated checkbox.
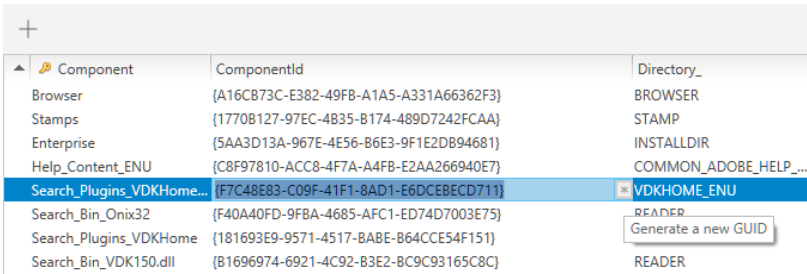
The available set of bit options and their specific meaning change for Attributes columns throughout Installer database tables. Therefore, the Attributes Editor dialog is always adjusted to the set of actually required values for the column it is currently opened from.

If the value labels alone do not provide enough information to clearly decide which options should be selected, the link at the bottom of the Attributes Editor dialog provides a direct deep link to the matching MSDN online documentation.

GUID Generator

Several tables within standard Installer databases have columns containing GUIDs. When a cell that accepts GUID numbers is focused, a button to generate GUID numbers becomes available:

Simply press the button to generate a new random GUID. This functionality is available for all columns declared in _Validation_ as _Guid_ type and certain values in the `Property` table (`UpgradeCode` and `ProductCode`).

Conditions Editor

Several tables within standard Installer databases contain Condition columns. The actual value of these columns is created as a collection of more or less commonly used conditional phrases,

combined via AND and OR operators. These conditions typically control the execution of sequence steps or determine the install level of features.

In order to provide assistance for the definition of these conditions, RayPack provides a condition builder interface. This interface may be called from several views of the user interface, typically by clicking on a link with the label "Edit in Condition Builder". The most recent updates of the **TABLES** editor include the extension of `Condition` table columns with a mechanism to directly call the Condition Builder for edition support.

[...] To open the condition builder for a specific database table cell, users have to double-click a Condition cell and click on the icon that appears at the right-hand side of the cell boundaries. The Condition Builder is invoked, ready for the creation of new or manipulation of existing conditional statements.
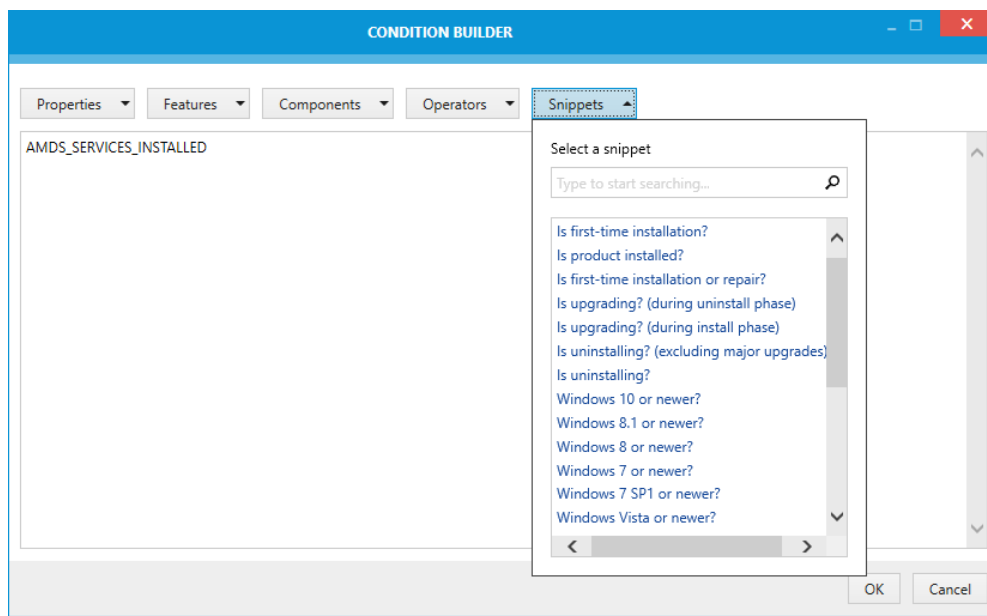
Please refer to the Common Dialogs section about the Condition Builder for further details on how to use this tool.

Reference Integrity Tracking

Since Windows Installer technology is based on the concept of relational database structures, there are tables that do not only contain primary keys, but also foreign keys, which establish object relations to the content of other database tables.

Whenever RayPack recognizes such a relation between table columns, it adjusts the direct inline edit mode of the affected cells: Users may keep on manually typing any value they see fit. But in order to keep an eye on the referential integrity of the Installer database, it is recommended to use one of the values offered by the drop-down selector control that is automatically added to cells that require it.

When users edit cells that offer a list of preselected values, a downwards arrow on the right-hand side of the cell indicates the availability of an additional editor control. Clicking it reveals a list of

possible values for the currently edited column. If the given cell value exists as item within this option list, a blue background color marker is used to indicate its selected state. Picking any of the options from the list replaces the old cell content with this new value.

Making use of the reference integrity features of RayPack has additional benefits: Whenever a referenced value is updated within the original database table, all references are updated automatically. Issues regarding inconsistent key relations are therefore a burden of the past.

The list of cell references (including also formatted string references among others) is shown below the table view. For more information, refer to the section Row tracking.

Input Validation

The guiding theme for functionality and intelligence that is added to RayPack in general, and the TABLES view in particular, is the idea that we assist packagers with hints regarding invalid values and activities where required, but leave them the freedom to be able to turn their back on best practice and do what they see fit. This is why input validation is elaborated an a high level within the Visual Designer mode of PackDesigner, but kept to some essential basics within the TABLES view. The table editor is the place where experience packagers may do as they please. Therefore, the following information about input validation for values entered via the TABLES view is the minimum Raynet considers absolutely vital for packaging projects. Checks that are not mentioned explicitly are most likely not executed.

Validation During Row Creation

Non-nullable Values

Values marked as required due to MSI standard schema definitions are checked for existence during the add row execution check routine.

Type Restrictions

The add row dialog checks whether an entered value is correct towards the expected type. Therefore, error messages will be triggered when strings are entered for values that are expected to be integers.

Domain Boundaries

RayPack checks whether values entered into cells with restricted domain boundaries are valid. For example, if a cell belongs to a column of type integer, the maximal lower and upper limit of the integer definition is checked. Additionally, when string cells are limited to a specific length regarding the number of entered characters, the actually entered string length is checked.

Error Handling Within the Add Row Dialog

When the dialog is opened, the limitations regarding non-nullable cells are automatically marked by red background color for the affected input controls. Hovering over the input field reveals a tooltip including the matching error message.

If invalid values are entered, the same schema of red background color and tooltip info is applied to the affected input control. Additionally, when users try to save a new row, invalid values are referred to in error messages displayed in a red error info box at the bottom of the add row dialog.

As long as there are error messages left, users can only either adjust the reported value issues, or exit the dialog by clicking on CANCEL.

Validation During Row Manipulation

Non-nullable Values

Values marked as required due to MSI standard schema definitions are checked for existence during the save cell update check routine.

Type Restrictions

The save cell update check routine checks whether an entered value is correct towards the expected type. Therefore, error messages will be triggered when strings are entered for values that are expected to be integers to name just one occasion.

Domain Boundaries

RayPack checks whether values entered into cells with restricted domain boundaries are valid. For example, if a cell belongs to a column of type integer, the maximal lower and upper limit of the integer definition is checked. Additionally, when string cells are limited to a specific length regarding the number of entered characters, the actually entered string length is checked.

Error Handling Within the Table Details Pane

If invalid values are entered and the user tries to exit the direct inline edit mode for a cell, RayPack checks the value and displays an exclamation mark icon if the value is not valid. Hovering over the icon reveals a tool tip with the description of the issue that needs to be solved before the cell update can be accepted.

As long as the marker for invalid values is still displayed, users may either abort the cell update by hitting Escape on the keyboard, or keep on trying to adjust the entered value to match the given restrictions in order to be able to finally save the change.

Importing and Exporting Binary Resources

Several tables within the standard Installer databases contain stream columns. Two tables containing binary streams that are frequently used are the `Binary` table (for binary data and bitmaps) and the `Icon` table (for icons).
Streams can be imported and exported directly from within the Tables view.

A binary stream can be imported to or exported from a focused cell only. To access this functionality, focus the required binary cell using your keyboard and/or mouse. Press F2 or Enter or click the focused cell again to activate the edit mode.

The active cell provides two buttons for exporting and importing its binary content.

> **Note:**
> Due to Windows Installer limitations, the ability to discover the original file type is limited. RayPack uses binary stream names to determine default extensions of exported files. However, if the stream name contains no extension (like in the picture above) it is up to user to select the correct extension so that the exported file can be opened by its associated application.

Importing and Exporting Tables

The tables can be imported to and exported from RayPack using a widely supported `.idt` format. The exported tables are stored using open text, together with necessary metadata, schema, and values.

In Order to Export a Table...

1. Select the table in the table list

2. Right click to show a context menu
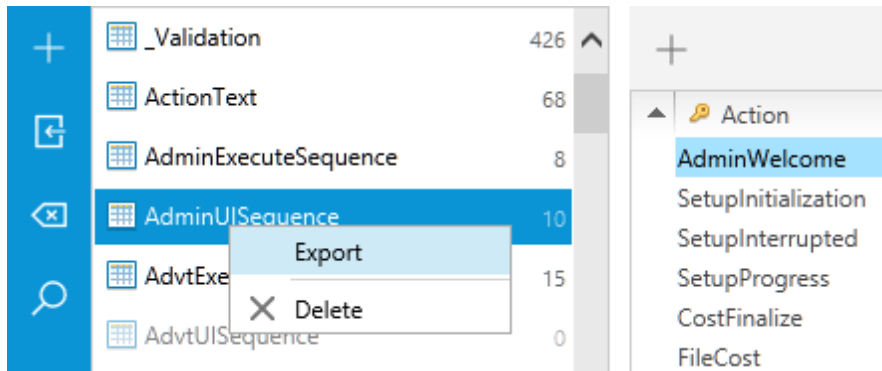
3. Click **Export**.

4. Select the target location of an `.idt` file.

5. Press **Save** to finish exporting.

In Order to Import Table(s)...

1. Press **IMPORT** icon from the left sidebar
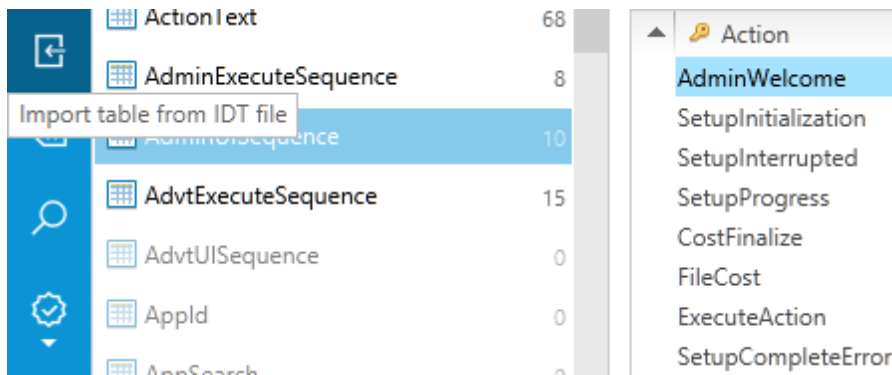
2. Press **Add a file to be imported...** and select any existing `.idt` file.

3. The extended grid shows the list of imported tables and the option to decide what to do in case of conflicts. For example, as pictured below, the table `Error` which is already present in the current MSI can be replaced or merged on import.
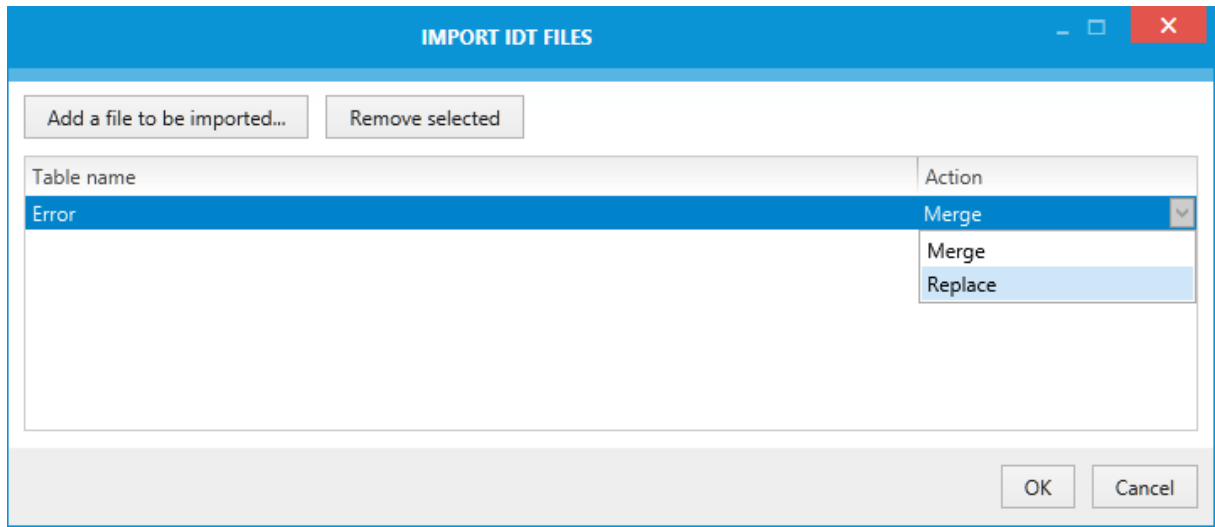
4. Repeat step 3 to add all required tables.

5. Press **OK** to import the selected tables.

Features

The **FEATURES** view allows to manage packaging project feature objects. Compared to the Feature view available within the Visual Designer mode, the **FEATURE** view of the Advanced mode is designed to provide in depth access to advanced feature properties and settings, including access to the components assigned to a specific feature.
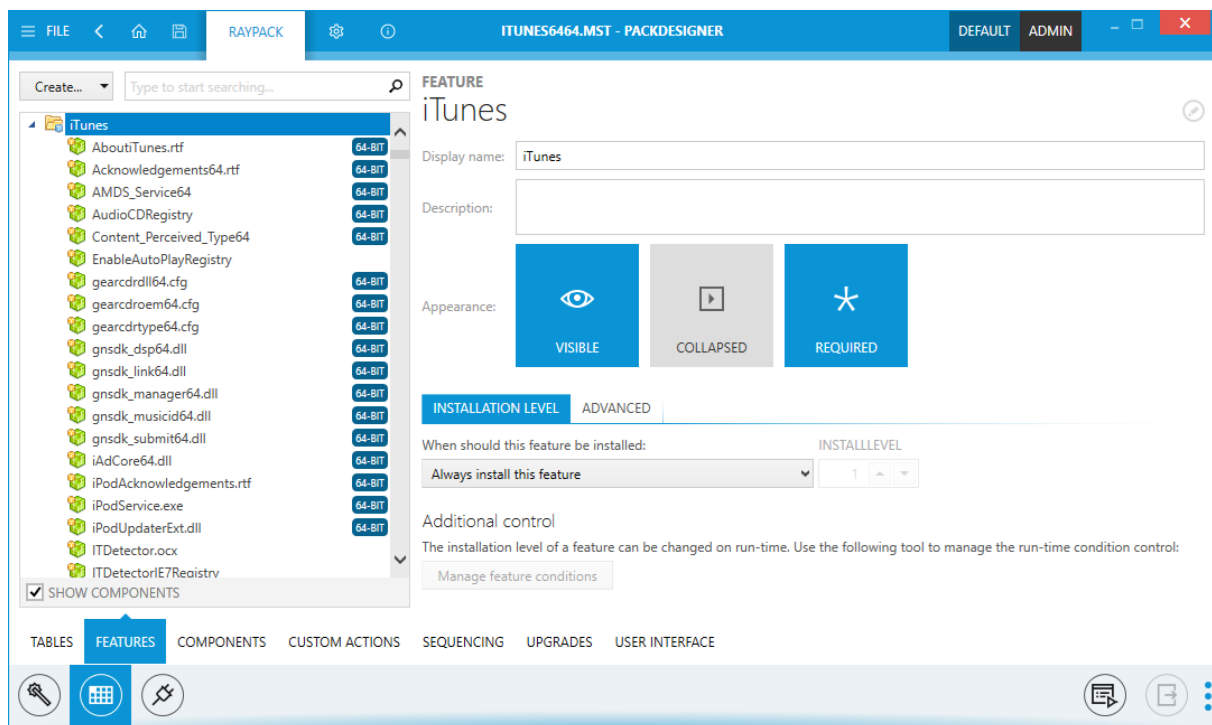
The **FEATURES** view is organized to display the tree structure of already existing features (and

the components which are assigned to them) on the left-hand side, and a details pane on the right for reviewing and manipulating properties and settings for the currently selected feature.

Since the tree structure of features and their components may become quite extensive within complex packaging projects, RayPack offers some supportive interface options for swift orientation:

- At the bottom of the feature tree view, there is an option Show components. It is activated by default. Disabling the checkbox immediately reduced the displayed objects of the tree view to features - components are hidden.
- Additionally, the tree structure may be totally expanded or collapsed by the options provided by the context menu that is revealed when a user right-clicks anywhere within the feature tree display area.

> **Note:**
> Since the **FEATURE** tree view usually displays features and components, selecting an object from that tree may either display the details pane for feature or component properties. This help section is about features. Please refer to the **COMPONENT** view help section for details regarding the functionality provided for component management.

Feature Object Structures

Features are organized in a tree like structure: based on one or more root features (= nodes), packagers nest child features as further node levels and add component assignments as leaves for the tree structure. Since features need to have at least one component assignment, and features cannot be nested below a component, features should never be the leaves of this specific feature tree construct.

Hidden features (the features that are not visible in the installer user interface) are displayed in a grayed out mode and always shown before any other visible features.

Features represent the lowest level of user interaction regarding decisions about what parts of a deployment package (which may be a software application, simply a group of resources transferred to the target machine, or even a set of scripts designed for execution on the target machine) should be actively executed during the package sequences. Since users do not see components but, if configured to be visible, features, it is highly recommended to keep the user perspective in mind when the feature tree is built.

The FEATURE view allows to define settings that prevent users from seeing or optionally selecting features. It is even possible to decide whether a feature may be advertised or installed remotely. Please refer to the following help sections for details about available, recommended and default feature property settings.

Core Feature Management Procedures

- *Add a new feature*
- *Rename a feature*
- *Remove a feature*
- *Edit a feature*
- *Assign components to a feature*

When a feature is right-clicked from the tree view area, the context menu reveals options to rename and remove features, as well as items that allow to *add a new child feature*, and to call the already focused object row in the Features table within the TABLES view.

Selecting **Go to Row** from the context menu loads the TABLES view with the `Feature` table displayed, and the data row of the selected feature highlighted for swift recognition.

Add a Feature

Adding a feature to a packaging project from within the **FEATURE** view can be accomplished by two different methods:

- *Adding a root feature*
  When a new packaging project (RPP) is created, it does not contain any features, hence the first feature that is added has to be a root feature to initiate the tree structure
- *Adding a child feature*
  Child features may be nested to any other feature. However, the depth of the feature tree may not exceed 16 (node) levels, as this is the limitation given by the MSI format standard definition.

Adding a Root Feature

To add a new feature on the topmost level of the feature tree structure, users have to click on the **Create** button, available on the upper left area of the FEATURE view, right above the display area for the existing feature tree. From the menu of activity options, **New root feature** has to be selected.

RayPack automatically adds a new root feature, pre-configured with default settings for new features. These defaults have to be adjusted towards the required feature behavior. First of all, the identifier of the feature should be updated, hence the new item added to the feature tree view is set into direct inline edit mode to indicate the requirement for identifier adjustment. Users may simply start to type for immediate replacement of the default feature identifier.

Please refer to the help section on editing features for details on how to adjust feature properties and what naming conventions have to be followed for feature identifiers.

Adding a Child Feature

To add a child feature to the existing tree structure, users have to **right-click** the desired **parent feature**, and select **New feature** from the context menu. As an alternative, it is also possible to left-click a feature and hit **Insert** on the keyboard to add a new child feature.

Child features are generally handled similar to root features, except of the lower level within the tree structure. Therefore, the very same defaults and update requirements mentioned for root nodes are given for child features as well.

Positioning of New Features

The position of any new feature within the tree structure is defined automatically: New elements are always added to the last position of the targeted node level. To adjust the feature order, users may

- Apply drag and drop on a feature: clicking the item within the tree structure, dragging it to the desired target parent feature, and dropping it there immediately applies the new positioning. or
- Manually update the values set in the Display columns of the affected objects in the `Feature` table. RayPack orders features with the same parent ascending towards their Display value.

> **Be aware:**
> The Display value is used for other feature property definitions as well. It is not recommended to adjust the values manually, unless the resulting effects on settings for feature visibility during installation and the control regarding default behavior for collapsed or expanded display are generally kept in mind, and checked carefully once the manipulation is finished.

Rename a Feature

To rename a feature from the **FEATURES** view, users may use different methods:

- **Left-click** a feature in the feature tree structure on the left-hand side of the view and hit **F2** on the keyboard
- **Right-click** a feature in the feature tree structure on the left-hand side of the view and select **Rename** from the context menu
- Left-click a feature in the feature tree structure on the left-hand side of the view and
  o click the existing feature identifier value in the details pane on the right-hand side
  o click on the edit icon right next to the existing feature identifier value in the details pane on the right-hand side

Either way, the edit mode for the feature identifier value is enabled. Users may directly enter the new identifier value and hit **Enter** on the keyboard to save the new state. (Please refer to the help section about common dialogs for more details about the *direct value editor* interface used in the details pane of the **FEATURES** view.)

> **Be aware:**
> When a feature is renamed, this manipulation takes effect on the identifier of the feature data object, not the display name.

Naming Conventions for Features

Internally, the feature name is stored as a value within an identifier column type. Therefore, the usual restrictions for identifier values have to be considered in order to provide a valid feature name value:

- The name of a feature has to be a unique, non-empty alphanumeric string of max. 38 characters length.
- Feature names may contain the ASCII characters A-Z (a-z), digits, underscores (_), or periods (.).
- Every identifier must begin with either a letter or an underscore.

Remove a Feature

From the **FEATURES** view, removing a feature can be triggered by either:
- **right-clicking** the feature and selecting **Remove** from the **context menu**
  or
- **left-clicking** a feature and hitting **Delete** on the **keyboard**

When the remove procedure is triggered, a confirm dialog is displayed.

> **Be aware:**
> Even if the feature that is about to be deleted does not contain child features, there may be other objects deleted along with it: Whenever a component is assigned to a feature, and the feature is deleted, the component and all its contents (e. g. files, registry keys and values, COM class definitions, etc.) will be deleted if the component is not assigned to another feature.

To execute the remove procedure, users click **REMOVE**. Please be aware that removed features cannot be restored.

> **Note:**
> To suppress the display of the confirm dialog for future remove procedures, the checkbox displayed for the confirm dialog control may be activated in advance. If it is activated, future remove procedures will be executed immediately. Since deleting features may cause severe loss of data, it is not recommended to deactivate the confirm dialog.

Users may exit the confirm dialog without removing the feature by clicking on **DO NOT REMOVE** or **CANCEL**.

Edit a Feature

To edit a feature from the **FEATURES** view of PackDesigner's Advanced mode, users have to load the properties of the data object into the details pane. To do so, they have to select it from the tree structure on the left-hand side of the view. Once this is done, the properties listed below are ready for manipulation.

> **Note:**
> Since the **FEATURE** tree view displays features and components, selecting an object from that tree may either display the details pane for feature or component properties. This help section is about features. Please refer to the **COMPONENT** view help section for details regarding the functionality provided for component management.

Basic Properties

Feature Identifier

The feature identifier, also called feature name, is the unique identifier of a feature object within the Installer database. To edit it, users have to double-click on the existing value. The direct value editor interface is activated and the new value can be inserted immediately.
Please refer to the help section *Rename a feature* for details regarding naming conventions that have to be kept.

Display Name

The display name is the title end-users see during the installation process of a target package (if the feature is configured to be visible, see *below*). Therefore, whilst the feature identifier is used for technical organization, the display name is designed for end-user communication.

Description

The description is a localizable string displayed to end-users as additional feature information during the installation run time (if the feature is configured to be visible, see *below*).

Appearance Options

The options listed below control the general visibility and interoperability of a feature during run-time. The combination of all three options is evaluated by the installer to build the target packages **Selection** dialog.

Visible / hidden
If a feature is marked to be visible, end-users see the feature during the installation run-time as part of the feature listing displayed in the **Select** dialog. Simply displaying a feature does not take effect on the further control options for the feature. If it is marked to be not deselectable by the end-user, the feature is shown, but cannot be manipulated.

The visibility of a feature is handled as inheritable property. This means that hiding a parent feature from display will cause the installer to hide all child features as well - no matter how their individual visibility settings are defined. This is part of the MSI standard execution routine and cannot be overridden.

When a new root feature is created in RayPack, it is visible by default. Newly created sub-features inherit this property from their parents.

To switch between the visible and hidden state, users have to click on the tile for visibility settings. Each click inverts the current state from visible to hidden and vice-versa.

Expanded / collapsed
If a feature is marked as expanded, its direct child nodes will be shown without the requirement for user interaction. This default expansion of the feature sub-tree is handy when user interaction is expected for the child features, e. g. because these are optional items such as language packs and the like. The expanded by default option takes effect on the direct children of a feature. To

expand deeper levels as well, the expanded by default option has to be set for each affected feature.

When a new root feature is created in RayPack, it is collapsed by default. Newly created sub-features inherit this property from their parents.

To switch between the collapsed and expanded state, users have to click on the tile for display settings. Each click inverts the current state from expanded to collapsed and vice-versa.

Actually, the display setting may be affected by the visibility for a feature: Hidden features are automatically defined to be collapsed. Clicking on the light-grey tile does not affect the display setting at all. The tile becomes available for clicking (and gets a slightly darker shade of grey again) as soon as the feature is visible again. This unavailability of display settings is also inherited: If a parent feature is invisible, child features may not be set to be visible, and therefore not be displayed expanded, as well.

Required / optional
If a feature is marked as required, end-users cannot choose manually whether it should be installed or not - it will always be installed as long as the internal package logic assigns it. The feature may be visible, in order to indicate that it is installed, and to give information about it, but it cannot be kicked-out from the end-user during a custom installation setup.

When a new root feature is created in RayPack, it is optional by default. Newly created sub-features inherit this property from their parents.

To switch between the required and optional state, users have to click on the tile for requirement settings. Each click inverts the current state from required to optional and vice-versa.

	Note:
📎	The possible combinations of the **expanded by default** and **visible in the feature selection** dialog are internally coded into the values of a **Features Disp**lay column. Therefore, when this property has been manipulated directly via the **TABLES** view, it is recommended to check the status of the checkboxes in order to make sure the feature appearance is set as originally intended.

Tab: INSTALLATION LEVEL

Install Level Options

In RayPack, there are four standard settings for the install level of a feature:

- **Use default settings**
  Sets the value stored within the `Level` column of the feature to 3.

- **Never install this feature**

Sets the value stored within the `Level` column of the feature to 0.

- **Always install this feature**
  Sets the value stored within the `Level` column of the feature to 1.

- **Only if the INSTALLLEVEL property is greater or equal to x**
  Allows to manually define the value stored within the `Level` column of the feature to an arbitrary integer number.

The install level of any feature can be manipulated due to results of the evaluation of conditions. However, if the install level of a feature is higher than the level defined by the package-global install level property, this feature will not be installed.

Feature Conditions

In order to manipulate the conditions evaluated for a specific feature, users are directly directed to the table editor where the `Conditions` table is already loaded. Usually feature conditions are used to change the install level of features according to target system properties, such as architecture, operating system version or system language. Please refer to the *MSI standard documentation* for details regarding condition handling.

Tab: ADVANCED

Destination

This property defines which directory is used as default installation destination on the target machine. When a feature is set to be visible for end-users, and a public property is given as destination, the directory value may be manipulated during package run time.

Advertised Option

Internally, the advertisement options provided by RayPack are translated into bit wise evaluated values of the `Attributes` column within the `Features` table.
Since these options have to be handled with special care regarding cross dependencies and conditions, it is highly recommended to review the *MSI standard documentation* before manipulating the defaults set by RayPack.

Available options are:
- **Default**
  Set this attribute and the state of the feature is the same as the state of the feature's parent.
- **Disallow advertise**
  Set this attribute to prevent the feature from being advertised.
- **Favor advertise**
  Set this attribute and the feature state is Advertise.
- **Disable advertising if not supported by OS**
  Set this attribute and advertising is disabled for the feature if the operating system shell does not support Windows Installer descriptors.

Remote Installation Option

Internally, the remote installation options provided by RayPack are translated into bit wise evaluated values of the `Attributes` column within the `Features` table.
Since these options have to be handled with special care regarding cross dependencies and conditions, it is highly recommended to review the *MSI standard documentation* before manipulating the defaults set by RayPack.

Available options are:
- **Favor local**
  Components of this feature that are not marked for installation from source are installed locally.
- **Favor source**
  Components of this feature not marked for local installation are installed to run from the source CD-ROM or server.

Rearranging Features

The features view shows the simulated preview of the feature structure as it will be shown in the installer user interface. The features that are visible are displayed in full-color. The features that won't be visible in the installer interface are grayed-out.

Note:
If a parent feature is hidden, its children will be also hidden in the installer interface. However, RayPack may show the child features in full-color mode (reserved for visible features) if their attributes say so.

The position and relationship of features can be adjusted by using a drag and drop technique. Two different types of operations are possible:

- **Reordering features**
  To achieve reordering, users have to drop features at the position between the new neighbors, indicated by a vertical bar at the drop position.
- **Nesting features**
  To achieve nesting, users have to drop features at the position of the new parent feature, indicated by a highlighted background-color of the new parent feature at the drop position.

The following limitations to this mechanism exist:

- It is not possible to reorder hidden features within the same level due to the Windows Installer limitations.
- It is not possible to reorder any feature before any hidden feature due to the Windows Installer limitations.
- It is not possible to drop a parent feature into any of its descendants.

Combining multi-selection with this drag and drop technique can be used to quickly adjust the layout of the whole feature tree.

Note:
RayPack shows all features collapsed by default. Collapsing and expanding nodes in this view has no effect on the actual collapse/expand settings users will see when the installer interface is shown during package run time. To determine whether a feature should be collapsed or expanded by default, the settings in the sidebar have to be adjusted.

Add a Component

To add a new component to a selected feature, users have to right click the feature and select **ADD COMPONENT** from the context menu.

RayPack automatically adds a new component, pre-configured with default settings for new components. These defaults have to be adjusted towards the required feature behavior. First of all, the name of the component should be updated, hence the new item added to the feature tree view is set into direct inline edit mode to indicate the requirement for the name adjustment. Users may simply start to type for immediate replacement of the default component name.

Please refer to the content of the *COMPONENTS* section for further information about additional component manipulation options.

Assign Components to a Feature

Every feature needs at least one component assignment to be valid and installable. In order to create a relation between a features and components, users have to **select the component** and establish a relation from that component to the desired feature.

From the **FEATURES** view, this may be triggered by a **right-click** on any of the displayed components. Selecting **Assign to feature...** from the displayed context menu opens the **Select Feature** dialog. In order to quickly copy a component assignment from one feature to another, drag the required component(s) and drop it/them into the target feature. The dragged components will immediately be assigned to the new features.

The drag and drop technique has the following limitations:

• Dropping is possible if at least one of the dragged components does not already belong to the target feature
• Dragging and dropping is possible if the current selection consist of one or more components. If a multi-selection is mixed of both feature(s) and component(s), dragging is disabled.

> **Note:**
> The old assignment of the dragged component is not removed. Dropping components to features creates new assignments only.

Components

The **COMPONENTS** view is designed to manage the properties of the component objects required to populate features. Whilst the **FEATURE** view shows components in their feature relation context, the COMPONENTS view is designed to provide direct access to the components as core elements of MSI based packaging routines.
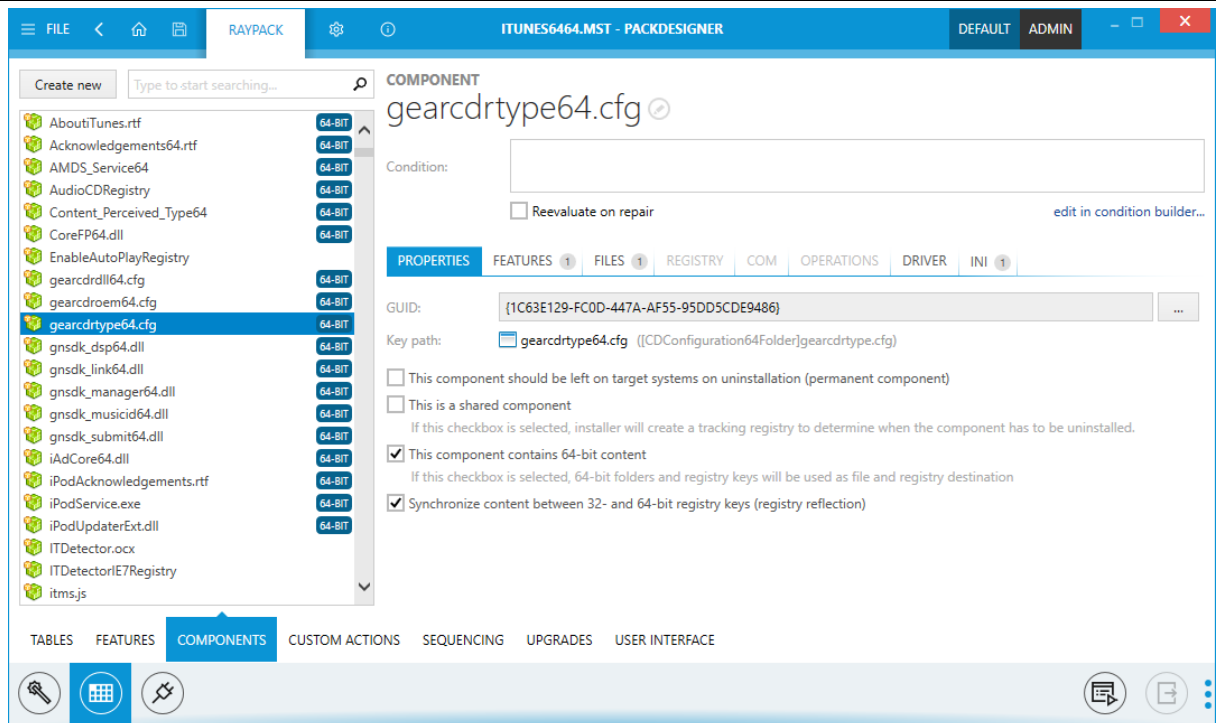
The left area of the **COMPONENTS** view is used to show the flat list of components available within the current packaging project. As soon as a component object is selected from this list, the details pane on the right-hand side of the COMPONENTS view is loaded with the properties defined for that component. Users may view and edit component properties within the same details pane.

Be aware:
Components must be assigned to at least one feature in order to take effect on the package installation procedure. Therefore, unassigned components are marked with a red icon inside the list area. Please make sure to have a list of components with green icons before the packaging project is compiled into an actual package. MSI packages with unassigned components are simply not valid.

Standard Component Object Manipulation Procedures

The following set of procedures may be initiated from the **COMPONENTS** view:

- *Add a new component*
- *Rename a component*
- *Remove a component*
- *Edit a component*
- *Assign features to a component*

When a component is right-clicked within the list area, the context menu reveals options to rename and remove it, as well as items that allow to duplicate a component, and to assign the

component to a specific feature.

Selecting **Go to Row** from the context menu loads the **TABLES** view with the `Component` table displayed, and the data row of the selected component highlighted for swift recognition.

Searching for Components

The pool of components stored within a complex software packaging project can actually be quite voluminous. Therefore we added a search functionality to this view. To filter the list of components to those containing a specific keyword within the component name, users simply use the search field above the component list area. The live search starts to filter the list as soon as a key is entered, narrowing the result list down with every additional key. This intuitive way of searching is used in several RayPack views, since it is considered to be the fastest, most convenient, and intuitive method to filter lists and tables.

To deactivate the keyword filter, users simply remove the search string from the input field. There is no need to hit Enter, the display update is executed immediately.

Add a Component

To add a new component to a packaging project is often executed automatically as the result of a packaging task executed from the Visual Designer mode, e. g. when files are added or the registry is populated with new objects. However, adding a component manually is possible as well.

1. To do so, users simply hit the **Create new** button in the upper left corner of the **COMPONENT** view.
2. From the displayed list of activity options, select **Create a new component**.
3. The new object is immediately created and added to the component list.

Since the component is prepared with a default name and some default settings, adjustments are due. The first thing that should be done is the adjustment of the component name. Therefore, when a new component is added, its item within the component list is automatically set into direct edit mode. This way, any user input made is directly set as new component name.

To save the new component name, users have to either hit Enter on the keyboard or unfocus the component item by clicking on another interface object, such as another component list item or any control object displayed within the **COMPONENTS** view. The help topic *Rename a component* contains details about the *naming conventions for component identifiers* and possible user interface messages regarding valid or invalid component names.

Please refer to the help section *Edit a component* to get detailed information about the manageable component object properties.

Rename a Component

To rename a component from the **COMPONENTS** view, users may use different methods:

- **Left-click** a component in the item list on the left-hand side of the view and hit **F2** on the keyboard

- **Right-click** a component in the item list on the left-hand side of the view and select **Rename** from the context menu
- Left-click a component in the item list on the left-hand side of the view and
  - o click the existing component identifier value in the details pane on the right-hand side
  - o click on the edit icon right next to the existing component identifier value in the details pane on the right-hand side

Either way, the edit mode for the component identifier value is enabled. Users may directly enter the new identifier value and hit **Enter** on the keyboard to save the new state. (Please refer to the help section about common dialogs for more details about the *direct value editor* interface used in the details pane of the **COMPONENTS** view.)

Naming Conventions for Component Identifiers

Renaming a component actually means to manipulate the internal object identifier. Therefore, the usual restrictions on values stored within database columns of the identifier type have to be followed:

- The name of a component has to be a unique, not empty alphanumeric string.
- Component names may contain the ASCII characters A-Z (a-z), digits, underscores (_), or periods (.).
- Every identifier must begin with either a letter or an underscore.

If an invalid value is entered as component name, RayPack reacts differently, depending on the actual area where the renaming was executed:

- **Error handling in the list view**
  Invalid names are marked with a red error icon on the left hand side of the input area. Hovering over the icon reveals the error message as a tool tip. The new value will not be saved until all restrictions are evaluated positively.

- **Error handling in the details pane**
  Invalid names are marked with a red border and background color for the input field. Hovering over the control object reveals a tool tip message, indicating how to solve the issue.

Remove a Component

From the **COMPONENTS** view, removing a component can be triggered by either:
- **right-clicking** the component and selecting **Remove** from the **context menu**
  or
- **left-clicking** a component and hitting **Delete** on the **keyboard**

When the remove procedure is triggered, a confirm dialog is displayed.

> **Be aware:**
> Objects related to the component (e. g. files, registry keys and values, COM class definitions, etc.) will be deleted along with the component object.

To execute the remove procedure, users click **REMOVE**. Please be aware that removed components cannot be restored.

> **Note:**
> To suppress the display of the confirm dialog for future remove procedures, the checkbox displayed for the confirm dialog control may be activated in advance. If it is activated, future remove procedures will be executed immediately. Since deleting components may cause severe loss of data, it is not recommended to deactivate the confirm dialog.

Users may exit the confirm dialog without removing the component by clicking on **DO NOT REMOVE** or **CANCEL**.

Edit a Component

In order to edit a component from the **COMPONENT** view within the **Advanced** mode, its current settings have to be loaded into the details pane on the right. To do so, users select the component from the list view on the left-hand side.

Since the set of available options for component handling is quite broad, there are separate tab-grouped areas of activity:

- Basic component settings
- Tab: **PROPERTIES**
- Tab: **FEATURES**
- Tab: **FILES**
- Tab: **REGISTRY**
- Tab: **COM**
- Tab: **OPERATIONS**
- Tab: **DRIVER**

Please refer to the specific help section for further details on the tab sub-views with manageable component settings.

Basic Component Settings

No matter which tab of the edit component view is active, the upper area of the details pane always displays the following basic setting controls:

Component Identifier

To activate the direct inline edit mode for the component identifier, users may do the following.

- Click the existing component identifier value.

or

- Click on the **edit** icon right next to the existing component identifier value.

Once the editor dialog is visible, starting to type replaces the existing identifier value. Please refer to the help section *Rename a component* for details regarding *Naming conventions for component identifiers*.

Key Path

The read-only information regarding the current component key path displays an icon to determine the key path type (file or registry value), the identifier of the key path object itself, and the path to the key file. Modifying the key path of a component is available from the context menu provided for files and registry values within the tabs **FILES** and **REGISTRY**.

Conditions

The value entered into the conditions text area is evaluated to decide whether or not the component has to be installed on the target machine.

- If there is no condition given, the component will be installed.
- If the conditional statement is evaluated to **TRUE**, the component will be installed.
- If the conditional statement is evaluated to be **FALSE**, the component will not be installed.

The evaluation is executed at least once during the `CostFinalize` sequence action. Later, for example during software maintenance procedures, further references to the evaluation result may, but do not necessarily have to, lead to reevaluation according to current object states. Please read the online documentation for further details regarding *component* conditions and the syntax requirements of *conditional statements*.

In order to make any condition statement definition as easy as possible, RayPack contains a so called **Condition builder** interface. It is reused wherever conditions have to be defined. Please refer to the help section regarding *Common Dialogs* for further details on how to use the **Condition builder**.

Reevaluation

If the **reevaluate on repair** option is active, the conditional statement will be evaluated anew when the MSI Repair procedure is triggered. If this option is not active, the old value, calculated during installation, is used to determine whether or not the component should be installed after the software maintenance is finished.

Tab: PROPERTIES

The PROPERTIES tab of the edit component dialog offers the following manipulation options:

Component Id

The GUID is a unique value for the combination of component, version, and language. Even though the MSI Standard table definition allows empty component id's, it is highly recommend to always provide a fully qualified guid for permanent components, since this property is required for component registration, which in turn is required for MSI features, such as Repair, to operate correctly.

Clicking on the button on the right-hand side of the input field automatically generates a valid guid string, consisting of digits and uppercase letters.

Permanency

Activate this checkbox to keep the component on the target system beyond the package de-installation.

Sharing

If this flag is set, the component is marked to contain shared content. As long as there are other packages requiring the component, the highest available version is kept on the system.

Note:
This setting is not supported for Installer versions prior to 4.5.

Registry Destination

Activate this checkbox to mark a component as containing 64-bit content. If this is a 64-bit component replacing a 32-bit component, activate this checkbox, and assign a new Component Id.

Registry Reflection

Activate this checkbox to enable synchronization between 32 and 64-bit areas of the Registry. Internally, activating the checkbox is reflected as removing the default 512 bit value from the Attributes column.

Note:
This setting is not supported for Installer versions prior to 4.0.

Tab: FEATURES

This tab is designed to manage the feature assignments for components.

The overview of existing feature assignments is displayed as a group of tiles, each representing the relation to one specific feature. Whilst each component may be assigned to a specific feature exactly once or not at all, it is possible to assign a component to several features at the same time. The number of current feature assignments is displayed at the right-hand side of the **FEATURES** tab label.

Each tile that represents a feature assignment is linked to the table entry for that specific feature. Simply click the tile, or right-click it and select **Go to feature** from the context-menu to call the feature row within the TABLES view.

To Add a Feature Assignment

The button to call the Feature selection dialog for establishing new assignment is displayed on the upper left corner of the **FEATURES** tab. Click it to initiate feature assignment.

Browse the tree structure of features and select the one the component has to be assigned to. Multi-selections can be achieved by pressing the Control key whilst clicking additional feature items.

Confirm the assignment by clicking **OK**.

The Select Feature dialog is closed. The **FEATURES** tab of the components details pane is automatically updated to show the new assignment.

To abort the assignment procedure, users either click **CANCEL** or close the dialog by clicking on the close icon in the upper right corner of the dialog window.

If the new assignment is the first for the current component, the prior red icon presented for the component within the list view on the left-hand side of the **COMPONENTS** view is turned into green.

To Delete a Feature Assignment

Right-click the tile of the existing feature assignment that has to be removed. From the context-menu, select **Remove from feature**. The assignment is removed immediately.

Please keep in mind that removing the last feature assignment from a component may lead to an invalid target package, since MSI packages build from a project in such a condition will not pass ICE validation and may lead to improper installation routines.

As soon as the last feature is removed, the icon of the component in the item list on the left-hand side of the **COMPONENTS** view is switched into red in order to remind the user of the assignment requirement. If the component is not required any longer, it should be removed from the packaging project.

Tab: FILES

The **FILES** tab is designed to assist packagers in their task to organize the file related properties of components. Therefore, it beholds a control item to define the components basic directory as well as a file browser to import new files into, or remove already imported files from the components file stock. The number of files that are currently organized within the component is displayed at the right-hand side of the **FILES** tab label.

To Modify the Components Directory Property

When new components are created, their directory property is set to the `INSTALLDIR` defined for the main packaging project. To change this default, users click on the button at the right hand side of the read-only directory path input control, and open the *Select a folder* dialog. Please refer to the *common dialogs* section for more details regarding the handling of this dialog.

To Manage the Components Files

The lower area of the **FILES** tab contains a list view, displaying all files that have been added to the current component. The list is sortable by the name, version, and size column, switching between ascending and descending order by simply clicking once or twice on the desired column header.

With a right-click on any file, the context menu is displayed, revealing a substantial set of manipulation options along with the **Go to Row** option. Clicking this option loads the TABLES view, with the `File` table displayed and the row of the originally selected file marked for swift orientation.

To Import New Files into a Component

1. Somewhere inside the **list view of files** attached to a component:
   a. **Right-click** and select **Import file(s)...** from the context menu.
      or
   b. **Left-click** to focus the file list, and hit **Insert** on the keyboard.

2. A **Select files** system explorer window is displayed. Browse to the file that has to be added.

   Please note that it is possible to select several files by using the control key for multi-selection. However, adding a mixed selection of files and folders, or adding several folders is not available. To add the content of several folders to a component, the content of each folder has to be added in a separate step.

3. As soon as all desired files are selected, click **Open** to add them to the current component.

4. All files are automatically added to the directory that has been defined as root directory for the current component. The Select files dialog is closed and the list view of component files is updated to contain the newly added files.

To Define a File to be the Key Path of a Component

The icon displayed in the outer left column of the file list indicates if one of the files is marked as key path for the current component:

This is a standard file which has been added to the component

This file is marked as key path for the current component

Since only one file or registry value can be the key path for a component, setting this property for one file automatically removes it from the current key path object (file or registry value). To define a file as key path of a component:

- **Right-click** a file from the list, and select **Set as key path** from the context menu
  or
- **Double-click** a file from the list, activate the checkbox "**This file is critical for a given component (key path)**" within the displayed **FILE PROPERTIES** dialog, and click **OK**.

The file icon is automatically switched to indicate the new key path status of the file.

To Remove a File

Users may trigger file removal by

- **right-clicking** a file within the list, and selecting **Delete** from the context menu
  of
- **left-clicking** a file within the list, and hitting **Delete** on the keyboard

Either way, a confirm dialog is displayed, requesting the user to confirm the irrevocable file deletion:

- With a click on the **REMOVE** button, the file is actually deleted
- With a click on the **DO NOT REMOVE** or **CANCEL** button, the deletion procedure is aborted.

> **Note:**
> It is possible to suppress the confirm dialog by activating the "In future do not show this confirmation for delete operations" checkbox. However, deleting objects accidentally may cause serious consequences for the packaging project. Therefore, it is recommended to keep the confirmation as active part of the regular deletion procedure.

> **Be aware:**
> Removing a file from a component actually removes the file from the whole packaging project!

To Edit a File

Editing file properties is triggered by one of the following user activities:

- **Right-click** a file from the list, and select **Properties** from the context menu
  or
- **Double-click** a file from the list

Either way, the **FILE PROPERTIES** dialog is displayed. Please refer to the help section regarding *file manipulation* to get detailed descriptions for this dialog.

> **Be aware:**
> Editing a file via the components file tab changes the file properties globally for the whole packaging project!

To Rename a File

Renaming a file is triggered by one of the following user activities:

- **Right-click** a file from the list, and select **Rename** from the context menu
  or
- **Double-click** a file from the list, and change the file name within the **FILE PROPERTIES** dialog
  or

- **Select** a file from the list, and hit **F2** on the keyboard to set the name into direct inline edit mode.

Please refer to the help section *Rename a file* for details regarding file name conventions and restrictions.

	Be aware: Renaming a file via the components file tab changes the file name globally for the whole packaging project!

To Define a New File Operation

1. Somewhere inside the list view of files attached to a component, right-click any file and select **New operation** from the context menu

2. Select the required operation type:

   a. **Duplicate file:** The file will be copied to another path
   b. **Move file:** The file will be moved or renamed
   c. **Remove file:** The file will be removed

3. The wizard contains additional configuration options which can be used to define the behavior and schedule of file operations.

	Note: After the file operation is created, it will appear in the **OPERATIONS** tab. The component view does not show file operations along with other files.

Tab: REGISTRY

The **REGISTRY** tab is designed to provide full control over the registry objects organized within a specific component.
The tab is separated into a tree view containing registry hives and keys on the left and a list view of values that have been added to the currently selected registry container (selected from the tree view). The number of registry keys and values that are currently organized within the component is displayed at the right-hand side of the **REGISTRY** tab label.

With a right-click on any key or value, the context menu is displayed, revealing a substantial set of manipulation options along with the **Go to Row** option. Clicking this option loads the **TABLES** view, with the `Registry` table displayed and the data row representation of the originally selected object marked for swift orientation.

In order to manage registry objects, users may execute one of the following activities:

- *Add a registry key*
- *Rename a registry key*

- *Remove a registry key*
- *Edit a registry key*
- *Add a registry value*
- *Rename a registry value*
- *Remove a registry value*
- *Edit a registry value*
- *Set a registry value as key path of the component*
- *Export the registry contents*

> 📎 **Note:**
> In order to add a RemoveRegistry operation to a component, the **OPERATIONS** tab has to be used. This function cannot be triggered by the interface options provided via the **REGISTRY** tab.

Add a Registry Key

To add a new registry key, users may:

- **Right-click** a registry key or hive within the tree view on the left-hand side, and select **New key** from the context menu
  or
- **Left-click** a registry key or hive within the tree view on the left-hand side, and hit **Insert** on the keyboard

Either way, a new key is added to the selected parent object, already prepared with a default name (e. g. New key) and default settings. The key name may be modified by simply starting to type, since after creation the name is set into direct inline edit mode.

Rename a Registry Key

To rename a registry key, users have to

- Activate the **direct inline edit mode** for the key name
  - With a **right-click** on the key, selecting **Rename** from the context menu
    or
  - With a **left-click** on the key, hitting **F2** on the keyboard

  For details on renaming restrictions for registry keys, please refer to the specific help section *Rename a registry key.*

- Open the **Registry Key Properties** dialog
  - With a **right-click** on the key name, selecting **Properties** from the context menu
    or
  - With a **left-click** on the key name, and hitting **Alt + Enter** on the keyboard

For details on edit options for registry keys, please refer to the specific help section *Edit a registry key.*

Remove a Registry Key

Users trigger registry key deletion by:

- **Right-clicking** a key, and selecting **Remove** from the context menu.
- **Left-clicking** a key, and hitting **Delete** on the keyboard.

Either way, a confirm dialog is displayed, requesting the user to confirm the irrevocable key deletion:

- With a click on the **REMOVE** button, the key is actually deleted
- With a click on the **DO NOT REMOVE** or **CANCEL** button, the deletion procedure is aborted.

Note:
It is possible to suppress the confirm dialog by activating the "In future do not show this confirmation for delete operations" checkbox. However, deleting objects accidentally may cause serious consequences for the packaging project. Therefore, it is recommended to keep the confirmation as active part of the regular deletion procedure.

Be aware:
Removing a registry key from a component actually removes the key and all values it contains from the whole packaging project!

Edit a Registry Key

Editing registry key properties is triggered by one of the following user activities:

- **Right-click** a key from the list, and select **Properties** from the context menu
  or
- **Left-click** a key from the list, and hit **Alt + Enter** on the keyboard

Either way, the **REGISTRY PROPERTIES** dialog is displayed. Please refer to the help section regarding *registry key manipulation* to get detailed descriptions for this dialog.

Be aware:
Editing a registry key via the components registry tab changes the key properties globally for the whole packaging project!

Add a Registry Value

To add a new registry value, users may:

- **Right-click** a registry key within the tree view on the left-hand side, select **New value** from the context menu, and select the desired **value type** from the sub-menu.
  or
- **Left-click** a registry key within the tree view on the left-hand side, **right-click** somewhere in the **value pane** on the right-hand side, select **New value** from the context menu, and select the desired **value type** from the sub-menu.

Either way, a new value is added to the selected parent key, already prepared with a default name (e. g. New string value) and default settings. The value name may be modified by simply starting to type, since after creation the name is set into direct inline edit mode.

Rename a Registry Value

To rename a registry value, users have to

- Activate the **direct inline edit mode** for the value name
  o With a **right-click** on the value, selecting **Rename** from the context menu
    or
  o With a **left-click** on the value, hitting **F2** on the keyboard

    For details on renaming restrictions for registry values, please refer to the specific help section *Rename a registry value*.

- Open the **Registry Value Properties** dialog
  o With a **right-click** on the value object, selecting **Properties** from the context menu
    or
  o With a **left-click** on the value name, and hitting **Alt + Enter** on the keyboard
    or
  o With a **double-click** on the value name

    For details on edit options for registry values, please refer to the specific help section *Edit a registry value*.

Be aware:
Renaming a registry value via the components registry tab changes the key properties globally for the whole packaging project!

Remove a Registry Value

Users trigger registry value deletion by:

- **Right-clicking** a value, and selecting **Remove** from the context menu.
- **Left-clicking** a value, and hitting **Delete** on the keyboard.

Either way, a confirm dialog is displayed, requesting the user to confirm the irrevocable value deletion:

- With a click on the **REMOVE** button, the value is actually deleted
- With a click on the **DO NOT REMOVE** or **CANCEL** button, the deletion procedure is aborted.

Note:
It is possible to suppress the confirm dialog by activating the "In future do not show this confirmation for delete operations" checkbox. However, deleting objects accidentally may cause serious consequences for the packaging project. Therefore, it is recommended to keep the confirmation as active part of the regular deletion procedure.

Be aware:
Removing a registry value from a component actually removes the value from the whole packaging project!

Edit a Registry Value

Editing a registry value is triggered by opening the **Registry Value Properties** dialog:

● With a **right-click** on the value object, selecting **Properties** from the context menu
  or
● With a **left-click** on the value name, and hitting **Alt + Enter** on the keyboard
  or
● With a **double-click** on the value name

For details on edit options for registry values, please refer to the specific help section *Edit a registry value*.

Be aware:
Editing a registry value via the components registry tab changes the value properties globally for the whole packaging project!

Set a Registry Value as Key Path of the Component

The icon displayed in the outer left column of the registry value list indicates if one of the values is marked as key path for the current component:

This is a regular registry value which has been added to the component

This registry value is marked as key path for the current component

Since only one file or registry value can be the key path for a component, setting this property for one value automatically removes it from the current key path object (file or registry value). To define a registry value as key path of a component:

● **Right-click** a value from the list, and select **Set as key path** from the context menu
  or

- **Double-click** a value from the list, activate the checkbox "**This registry value is critical for a given component (key path)**", and click **OK**.

The registry icon is automatically switched to indicate the new key path status of the value.

Export the Registry Contents

To export the registry contents displayed in the **COMPONENTS** registry tab, users have to **right-click** somewhere within the **registry tree view** area on the left-hand side, and select **Export all entries** from the context menu.

A system browser dialog is displayed, in which **target location** and **file name** for the `.reg` file that will include the exported registry contents have to be defined. Once both settings are defined, click **OK** to start the export.
The system dialog browser is closed and the file automatically populated with the current registry contents of the selected component.

Tab: COM

A COM class is an implementation of a group of interfaces in code executed whenever you interact with a given object.
On each computer, COM maintains a database of all the CLSIDs for the servers installed on the system. This is a mapping between each CLSID and the location of the DLL or EXE that houses the code for that CLSID. COM consults this database whenever a client wants to create an instance of a COM class and use its services.

The COM tab contains a listing of COM classes registered to be organized within the current component, and beholds all controls required to add, edit and remove such objects. Internally, the properties of COM class objects are stored within the `Class` table of the Installer database. The number of COM classes that are currently organized within the component is displayed at the right-hand side of the COM tab label.

With a right-click on any COM class item, the context menu is displayed, revealing a substantial set of manipulation options along with the **Go to Row** option. Clicking this option loads the TABLES view, with the `Class` table displayed and the row of the originally selected file marked for swift orientation.

To Add a COM Class

Within the COM tab of the Advanced mode view COMPONENTS, users click the **NEW CLASS** button to generate a COM class object, by default equipped with an unique CLSID and some other vital properties.
As soon as the class object is created, the default values have to be adjusted to the actually required settings. Please refer to the Edit a COM class section for further details.

To Remove a COM Class

Users trigger COM class deletion by:

- **Right-clicking** a COM class object, and selecting **Remove** from the context menu.
- **Left-clicking** a COM class object, and hitting **Delete** on the keyboard.

Either way, a confirm dialog is displayed, requesting the user to confirm the irrevocable key deletion:

- With a click on the **REMOVE** button, the object is actually deleted
- With a click on the **DO NOT REMOVE** or **CANCEL** button, the deletion procedure is aborted.

To Edit a COM Class

Users have to call the COM class property details to access all relevant object properties for manipulation. To do so, users may:

* **Right-click** a COM class object displayed in the list, and select **Edit** from the context-menu. or
* **Left-click** a COM class object displayed in the list, and hit **Alt + Enter** on the keyboard.

The CLASS PROPERTIES dialog is displayed, revealing a tabbed interface with three groups of properties for manipulation:

* *Basic COM class settings*
* *PROGID settings*
* *CONTEXT settings*

Tab: BASIC

GUID

Each COM class is identified by a CLSID, a unique 128-bit GUID, which the server must register. COM uses this CLSID, at the request of a client, to associate specific data with the DLL or EXE containing the code that implements the class, thus creating an instance of the object.

Description

A textual description of the class.

File Type Mask

The file type mask has to follow the standard format [offset, cb, mask, value]:

* **offset** - If a positive integer value is given, it represents the number of offset bytes until the byte range begins. A negative integer indicates the same offset counted from the end of the file.
* **cb** - Counted from the position defined by the offset, this integer defines the number of bytes which define the logical AND operation on the mask.
* **mask** - Once the start position and length of the mask are defined, here is the actual mask

definition, used to decide which parts of the file have to be considered for the execution of the a logical AND operation.

- **value** - The value must equal the AND operation result in order to recognize the relation between a file and a specific class on the target device.

Icon

If a fitting icon is already available within the stock of icons stored within the current packaging project, it is possible to simply select it from the catalog. To do so, users have to click on the downwards arrow and pick the desired icon from the displayed dialog.
If a new icon object has to be created, users have to click on load from disk and select a `.ico`, `.exe`, or `.dll` file to use as new icon.

AppID

The AppID has to be a GUID that has already been added to the packaging project. It will be registered under HKEY_CLASSES_ROOT\AppID, and is designed to group security and configuration options of COM objects.

Paths

Use relative paths to be able to define flexible COM objects with their bare file name used for COM server definition. Internally, the option is reflected within the Attributes column of the Class table.

Tab: PROGID

Each COM class may be equipped with a default ProgId, defining the version independent identifier for a program. Internally, ProgId objects are stored within the `ProgId` table of the Installer database. RayPack allows users to handle several ProgId objects within this dialog, but only defines a relation to the current COM object for the id that is marked as default.

Add a New PROGID

To add a new ProgId, users simply hit the **Create new ProgId** button, visible in the upper area of the PROGID tab. A new ProgId object is automatically created and added to the ProgId database table. Users have to edit the default settings to define the actually intended ProgId information.

Edit a PROGID

Name
The string defining the name of the ProgId.

Parent
Select one of the other already existing ProgIds stored within the `ProgId` table as parent for the current one. When the ProgId is selected for installation, the corresponding version-independent ProgIds associated through the this parent relation are also selected for registration.

Description
A textual description of the ProgId.

Icon

If a fitting icon is already available within the stock of icons stored within the current packaging project, it is possible to simply select it from the catalog. To do so, users have to click on the downwards arrow and pick the desired icon from the displayed dialog.
If a new icon object has to be created, users have to click on load from disk and select a `.ico`, `.exe`, or `.dll` file to use as new icon.

Remove a PROGID

Users trigger ProgId deletion by:

- **Right-clicking** a ProgId, and selecting **Remove** from the context menu.
- **Left-clicking** a ProgId, and hitting **Delete** on the keyboard.

Either way, a confirm dialog is displayed, requesting the user to confirm the irrevocable object deletion:

- With a click on the **REMOVE** button, the ProgId is actually deleted
- With a click on the **DO NOT REMOVE** or **CANCEL** button, the deletion procedure is aborted.

	Note: It is possible to suppress the confirm dialog by activating the "In future do not show this confirmation for delete operations" checkbox. However, deleting objects accidentally may cause serious consequences for the packaging project. Therefore, it is recommended to keep the confirmation as active part of the regular deletion procedure.

	Be aware: Removing a ProgId from a component actually removes it from the whole packaging project!

Tab: CONTEXTS

Please note the following restrictions and conditions for components context management.

- Each class has at least 1 context
- Each class has maximum 1 instance of each type:
  - InprocServer32
  - InprocServer
  - LocalServer
  - LocalServer32
- The custom InprocServer names may not be the same as one of the four above
- Inproc and InprocServer32 may not have editable default handler or arguments

Add a New CONTEXT

Users may add a new custom context at any time, whilst adding any other type may be

unavailable due to already existing contexts of the same type. To add a new context, users hit the NEW CONTEXT button above the list of already created context objects. The options menu contains all context types, but offers only those for active selection, which have not already been added to the COM object.

Please note that new contexts are always added with default settings, which usually may only be manipulated partially.

Edit a CONTEXT

Name
The context name may only be edited for custom type contexts. Internally, the name is stored as string value.

Default Handler
This property may be edited for local and custom context types. If editable, users may select one of the following four options:

- None - representing an DefInprocHandler value of NULL
- 16 bit - representing an DefInprocHandler value of 1
- 32 bit - representing an DefInprocHandler value of 2
- 16/32 bit - representing an DefInprocHandler value of 3

Arguments
If a LocalServer or LocalServer32 CLSID key appears in the Context field, the text in this field is registered as the argument against the server and is used by COM to invoke the server. The DefInprocHandler and Argument fields can both be Null if LocalServer or LocalServer32 types are selected.

Remove a CONTEXT

Users trigger context deletion by:

- **Right-clicking** a context, and selecting **Remove** from the context menu.
- **Left-clicking** a context, and hitting **Delete** on the keyboard.

Either way, a confirm dialog is displayed, requesting the user to confirm the irrevocable object deletion:

- With a click on the **REMOVE** button, the context is actually deleted
- With a click on the **DO NOT REMOVE** or **CANCEL** button, the deletion procedure is aborted.

Since each COM class object requires at least one context, it is not allowed to delete all context objects via the dialog based user interface. To manually modify the context values, users have to call the `Class` table in the **TABLES** editor.

> **Note:**
> It is possible to suppress the confirm dialog by activating the "In future do not show this confirmation for delete operations" checkbox. However, deleting objects accidentally

may cause serious consequences for the packaging project. Therefore, it is recommended to keep the confirmation as active part of the regular deletion procedure.

Be aware:
Removing a context from COM object and component actually removes it from the whole packaging project!

Tab: OPERATIONS

The **OPERATIONS** tab is designed to assist packagers in their task to organize the file or folder operation related properties of components. The view contains a list of all operations that have already been added to component items. These operations may be performed at run-time by the installer engine:

- Duplicate file
- Move file
- Remove file(s)
- Remove folder
- Remove registry
- Create folder

The view shows a full list of operations assigned to the selected component, regardless of their source and destination folders.

To View Operation Details

To see extended description of the file operation (including partially resolved paths) users have to hover the mouse cursor over the operation icon.
In order to preview the MSI row responsible for a given operations, user has to right click the required entry and choose **Go to row** option from the context menu.

To Manage Operations

Users may trigger the properties dialog of each file operation by right-clicking the required entry and choosing **Properties** from the context menu. A new dialog will appear.

To Remove Operations

Users may remove any operation by right-clicking the required entry and choosing **Remove** from the context menu.

To Add File Operations

Users may add new file operations by right-clicking anywhere within the operations view and choosing **New operation** from the context menu. From the Operations view, the following types are supported:

- Move file
- Remove file(s)

To add a new Duplicate file operation, go to the **FILES** tab and choose the option New file operation > Duplicate from the context menu. A wizard starts, and guides through the process.

To Add Registry Operations

Users may add a new remove registry operation by right-clicking anywhere within the operations view, choosing **New operation** from the context menu, and selecting **Remove registry** from the details menu. The *Registry operation wizard* is invoked. Please refer to the section about this wizard available from the Visual Designer mode's *Registry* view help content.

Tab: DRIVER

A component can have a driver assigned to it. In this case, the associations is shown as the content of this tab.

Note:
The view is functionally equal to the Drivers screen inside the Visual Designer and serves as a shortcut to the same functionality.
For more information how to use drivers, refer to section Drivers.

Tab: INI

The **INI** tab is designed to assist packagers in their task to organize the INI entries on a component basis. The view contains a list of all INI entries that have already been added to component items.

Assign Features to a Component

Assigning features to components can be triggered by two basic user actions:

- **Right-click** a component item within the list of components, and select **Assign to Feature...** from the **context menu**.
- **Left-click** a component item within the list of components, call the **FEATURES tab**, and click on the **Assign to features** button.

Either way, the **Select Feature** dialog is displayed.

Browse the tree structure of features and select the one that the component has to be assigned to. Multi-selections can be achieved by pressing the Control key whilst clicking additional feature items.

Confirm the assignment by clicking **OK**.

The Select Feature dialog is closed. The FEATURES tab of the components details pane is

automatically updated to show the new assignment.

To abort the assignment procedure, users either click **CANCEL** or close the dialog by clicking on the close icon in the upper right corner of the dialog window.

Custom Actions

The **CUSTOM ACTIONS** view is designed to provide an easy to handle interface for packagers who need to implement advanced routines and procedures for the fulfillment of complex packaging requirements. Custom action logic itself is hard enough to handle, therefore the interface has to support the packagers as well as possible.
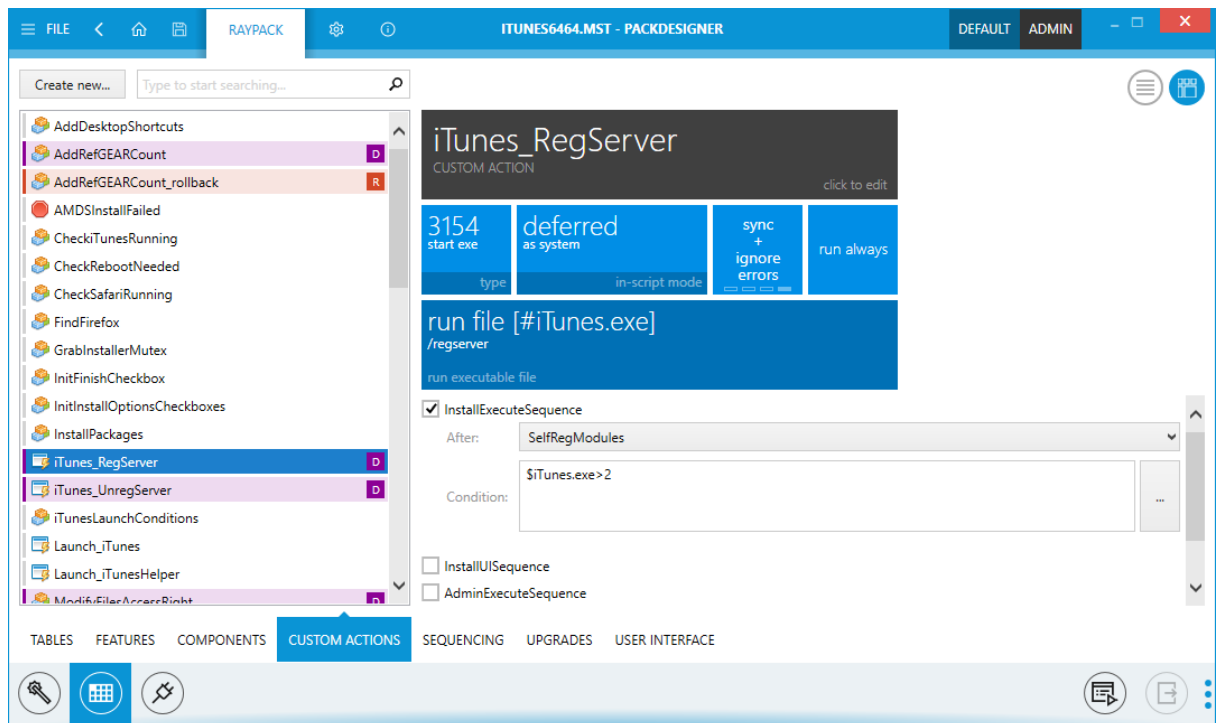
The **CUSTOM ACTIONS** view is separated into a navigation area on the left-hand side and a details pane for action settings on the right. This details pane is equipped with two basic view designs. RayPacks innovative tile based view, and a classic view - based on a table structure with label and value columns. Users may switch between these views on the fly by clicking the controller buttons in the upper right corner of the view.

The default setting: View custom actions in a tile based details pane (as shown in the screenshot above).
Switching to this view is done by clicking on the **tile** icon on the right.

The classic view: View custom actions in a table based details pane.
Switching to this view is done by clicking on the **row** icon on the left.

The list area of already existing custom actions displays the name and the type icon for the custom action. Custom action objects are by default ordered by their name. The type icon

indicates purpose and procedure of the specific custom action.

- VBS script

- Set folder

- JS script

- Executable

- Error message

- DLL

- Set property

Additional icons are displayed on the right of a custom action name if the run-mode is not set to the default (immediate).

D Deferred
R Rollback
C Commit

Please refer to the specific help topics to read details about standard manipulation options for custom actions.

- *Add a new custom action*
- *Remove a custom action*
- *Edit a custom action*

Searching for Custom Actions

The pool of custom actions stored within a complex software packaging project can actually be quite voluminous. Therefore we added a search functionality to this view. To filter the list of custom actions to those containing a specific keyword within the custom action name, users simply use the search field above the list area. The live search starts to filter the list as soon as a key is entered, narrowing the result list down with every additional key. This intuitive way of searching is used in several RayPack views, since it is considered to be the fastest, most convenient, and intuitive method to filter lists and tables.

To deactivate the keyword filter, users simply remove the search string from the input field. There is no need to hit **Enter**, the display update is executed immediately.

Sequencing of Actions

Position, condition and sequence of each **Custom Action** can be controlled directly from this

view. A set of checkboxes is responsible for 5 available sequences:

- `InstallUISequence`
- `InstallExecuteSequence`
- `AdminUISequence`
- `AdminExeucteSequence`
- `AdvtExecuteSequence`

After checking a checkbox, two additional fields are shown for each of affected sequences:

- **After:** This option determines the relative element after which the action is executed.
- **Condition**: This option determines the condition of execution

Add a new Custom Action

Adding a new custom action to a packaging project has to be triggered by a click on the **Create new ...** button, displayed at the upper left corner of the **CUSTOM ACTIONS** dialog. The Custom Action Wizard is invoked, starting to escort packagers throughout the process of custom action definition.

Wizard Step: Action Type

The actual step sequence of the wizard depends on the action type selection made on the first wizard screen:

Standard MSI custom actions:

- **Executable**
  - *Executable source*
  - *Execution parameters*
- **DLL**
  - *DLL Source*
- **VBS Script**
  - *Script Source*
  - *Script Content*
- **JS Script**
  - *Script Source*
  - *Script Content*
- The types **Set folder**, **Set property**, and **Show error** message do not contain any additional steps, but extend the **Details** step with specific settings.

Wrappers over standard MSI custom actions:

- **PowerShell**

To actually select one of the presented custom action types, the specific icon tile has to be clicked. The current selection is indicated by a colored box with a checkmark in the upper right

corner displayed as action type tile.

As soon as one type is selected, the following step has to be called with a click on the **NEXT** button. At each time during the wizard execution, users may abort the custom action creation by clicking the CANCEL button.

Optional Wizard Steps

The following wizard steps are displayed for specific custom action type creation procedures:

Executable Source

- Only for custom action type Executable -

Select one of the following source types with a click on the type icon:

- **Executable from a binary source**
  Requires an executable resource object within the `Binary` table.
  Please refer to the help section about *Resources* to get details on how to manage binary resources via the Visual Designer interface.

- **Executable deployed within this package**
  Requires an executable file object within the `File` table.
  Please refer to the help section about *Files and Folders* to get details on how to manage files via the Visual Designer interface.

- **Executable pointed by a property**
  Requires a property from the `Properties` table, containing a path that points to a specific file, which may either be stored within the current packaging project or on the target machine.
  Please refer to the help section about *Properties* to get details on how to manage properties via the Visual Designer interface.

- **Executable on destination**
  Allows to define a path to an executable file.

Clicking **NEXT** calls the **Executable parameters** step.

Executable Parameters

- Only for custom action type Executable -

When **Executable from a binary source** had been selected in the previous step, users have to select between the available options:
- **New file from disk** - select a file from a local or shared system drive.
  or
- **Existing binary stream** - select one of the binary resource objects that have already been added to the packaging project.

When **Executable deployed within this package** had been selected in the previous step, users have to select one of the file objects that have already been added to the packaging project.

When **Executable pointed by a property** had been selected in the previous step, users have to
- **Select the source property** - Pick one of the properties that have already been added to the packaging project
  and
- **Define the path to the executable** - The displayed value is automatically updated to match the actual content stored within the source property. Manually adjust it to define the actual path to the executable file.

When **Executable on destination** had been selected in the previous step, users have to select one of the directory objects that have already been added to the packaging project.

Command line arguments may optionally be given for every executable source type. Define the arguments that have to be appended to the command line that is used to actually call the executable.

Clicking **NEXT** in this step calls the **Details** step.

DLL Source

- Only for custom action type DLL -

When a DLL from a binary source has to be used:

Define the **Source DLL file** by activating one of the available options:

DLL from a binary source may be selected by the definition of either
- **New file from disk** - select a file from a local or shared system drive.
  or
- **Existing binary stream** - select one of the binary resource objects that have already been added to the packaging project.

DLL deployed within this package offers a drop-down menu to select one of the file objects that have already been added to the packaging project.

The DLL has to be called through the **Entry point**, passing the handle to the current install session as argument. The specified entry point must match that exported from the DLL.

Clicking **NEXT** calls the **Details** step.

Script Source

- Only for custom action types VBS and JS script -

Scripts may be added to custom actions via several methods:

- as a binary stream resource
- as inline script source code
- as a property value
- as a file deployed within the package

The method selected on the upper area of the wizard interface decides about the controls required to actually add the script, which are displayed in the lower part of the wizard dialog screen:

When **source stream** is selected, users may either trigger the creation of a new stream resource, or select one of the resources already available within the packaging project.

When **inline script** is selected, users will have to enter the code during the next wizard step.

When **property value** is selected, users may either trigger the creation of a new property, or select one of the properties already available within the packaging project.

When **deployed file** is selected, users may pick one of the files that have already been added to the packaging project.

Clicking **NEXT** calls the **Script content** step.

Script Content

- Only for custom action types VBS and JS script -

The control interface that is shown within this step directly depends on the selections made during the prior one:

When **source stream, inline script,** or **property value** have been selected, a text area is displayed, allowing to either manually type the script code, manipulate the already available code, or load source code from an external file. The text area may not be left empty, since a script based custom action must contain the code for execution.

Additionally, when **source stream, property value,** or **deployed file** have been selected, there is an input field, allowing users to define the actual function that should be called in the scope of the custom action. The function name is mandatory for deployed files, and optional for the other options.

Clicking **NEXT** calls the **Details** step.

Whilst the type depending steps may vary, the following steps are common and required for all types:

Wizard Step: Details

The details step demands the definition of the **execution mode**:

- **Immediate** - Custom actions that set properties, feature states, component states, or target directories, or that schedule system operations by inserting rows into sequence tables, can in many cases use immediate execution safely.
- **Deferred** - Custom actions that change the system directly, or call another system service, must be deferred to the time when the installation script is executed.
- **Rollback** - The rollback procedure runs as a clean-up whenever an installation fails to be executed. Custom actions that are marked to run during rollback will not be executed during the primary installation.
- **Commit** - The commit procedure runs to finalize successful installations. A commit custom action is the complement to a rollback custom action and can be used with rollback custom actions to reverse custom actions that make changes directly to the system.

Additionally, packagers have to determine the user that is defined to be executor of the custom action:

- **Run as system** - When custom actions are defined to run as system user, the properties of some user related properties and paths are affected, such as the user related AppData directories and current user registry targets.
- **Run as current user** - This standard setting impersonates the custom action to the user that actually installs the package.
- **Terminal server aware** - Required only in terminal server based target environments.

The following optional behavior settings may be activated:

- Run as **64-bit** script
  On 64-bit operating systems, Windows Installer may call custom actions that have been compiled for 32-bit or 64-bit systems. A 64-bit custom action based on Scripts must be explicitly marked as a 64-bit

- Check **exit code** upon finishing
  If this option is activated, the custom action will be considered as finished successfully if the return code 0 is given. All other return values express erroneous custom action execution. Deactivate this option to ignore the return code and always proceed with the procedure the custom action is a part of.

When **Set folder** has been selected as action type, this step is used to select the folder that has to be renamed, and the new value it has to be given. The folder identifier has to be an already known object from the Directory table. Users may use the syntax suggestion feature to set the new value based upon dynamic values, such as properties.

When **Set property** has been selected as action type, this step is used to select the property that has to be changed, and the new value it has to be given. The property identifier has to be an already known object from the Properties table. Users may use the syntax suggestion feature to set the new value based upon dynamic values, such as paths.

When **Show error** has been selected as action type, this step is used to determine the actual

error message that has to be displayed. The message may not be empty.

Clicking **NEXT** calls the **Sequence** step.

Wizard Step: Sequence

This step allows to optionally define in which of the installation sequences the custom action should run.

To add the custom action to one of the sequences, users have to expand the toggle for the required sequence type first. Once expanded, the toggle area contains the actual sequence(s) available for the type.
Activate the checkbox left of the sequence description in order to be able to define the position of the custom action within the specific sequence.
Custom actions may be added to no sequence, all sequences, or any combination of sequences in between.

This setting may be changed via the **SEQUENCING** view of the Advanced mode at any later time.

Clicking **NEXT** calls the **Additional Settings** step.

Wizard Step: Additional Settings

RayPack automatically generates a suggestion for the **name** of the custom action, based on the action type. The name of the custom action may not be left empty, has to be unique throughout the packaging project, and should not contain special characters.

The optional **conditional statement** input field may be used to enter a string for evaluation at run time. The custom action will only be executed if the condition is left empty (resembling a NULL value), or is evaluated to be TRUE.

Packagers may adjust the condition later as part of the *sequencing* management activities.

Clicking **NEXT** calls the **Summary** step.

Wizard Step: Summary

This overview provides a chance to check the settings made during the prior steps. Please verify that all properties are noted correctly towards the desired custom action behavior.
Use the BACK button to navigate to former steps in case of adjustment requirements. Please be aware that going back may affect the already given values in steps that are depending on each other. If, for example, the resource type for a script based custom action is changed, the actual script code has to be re-defined.

Clicking **PROCESS** creates the custom action according to the settings displayed, and closes the wizard dialog.

Adding PowerShell Custom Actions

PowerShell custom actions are exclusive feature that RayPack builds on top of the Windows Installer technology. The usage is similar to native scripting languages (`.vbs` or `.js`), with certain limitations.

- Three source types are allowed: binary table, inline script and a Windows Installer property. It is not possible to start a script deployed with the package.
- The whole content of a script will be started. It is not possible to start a particular function.
- Only Immediate and Deferred execution is supported. Rollback and Cleanup custom actions are not supported.

PowerShell script editor has a standard set of functions.
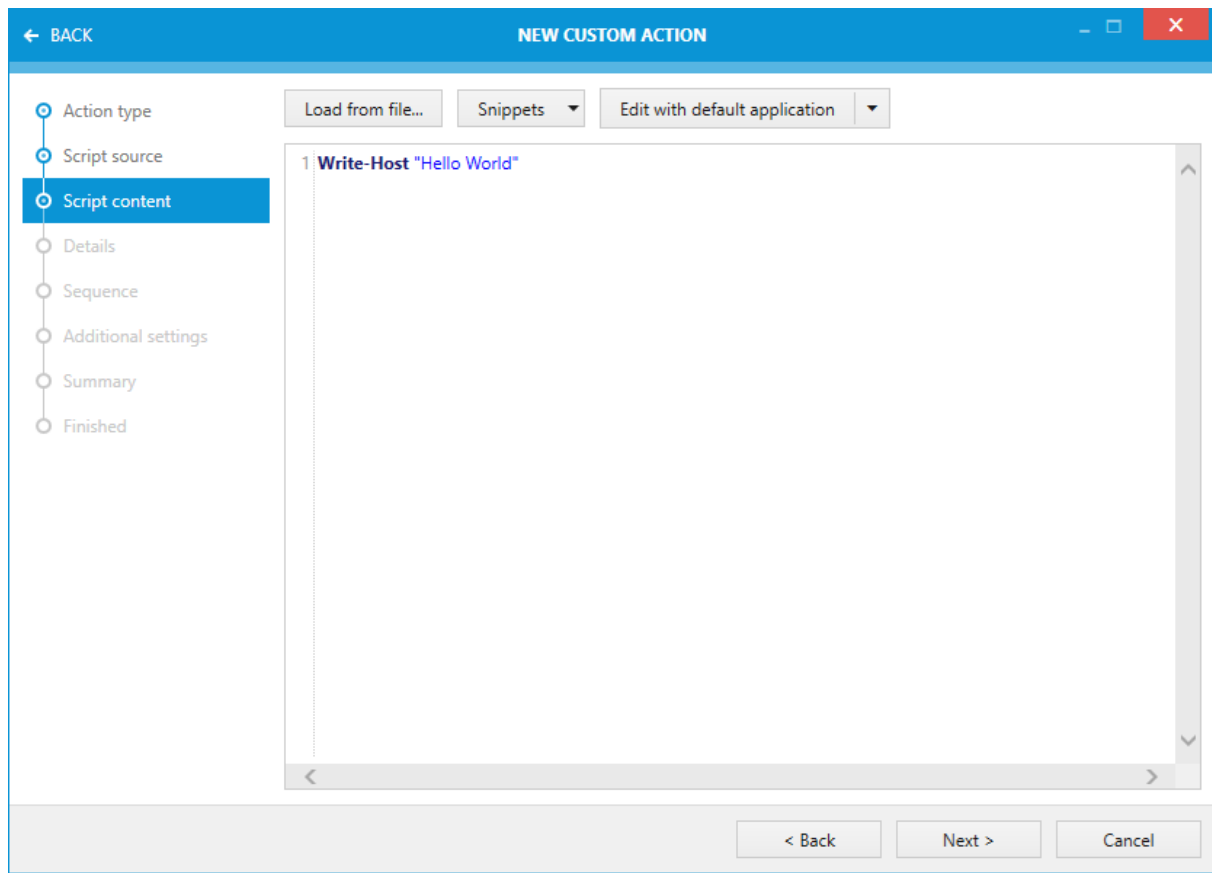
- **Load from file...** - Loads a content of an existing `.ps1` file.
- **Snippets** - A dropdown of reusable content, scripts, and snippets.
- **Edit with default application** - A split button opening the current script in external editor (default) or one of editors that is associated to open `.ps1` files.

The text editor offers syntax highlighting for PowerShell and line numbers feature.

Working with MSI Session and Properties

PowerShell script executed from RayPack environment as a Custom Action has special capabilities to access (read and modify) session properties and its methods. In order to access the session, use predefined object `$msiSession`. The methods provided by the session are described in the MSDN documentation of the automation interface: https://msdn.microsoft.com/en-us/library/windows/desktop/aa367810(v=vs.85).aspx.

Reading MSI Properties:

For a quick access to a value of an MSI property, use the following code sample:

```
$productCode = Get-MSIProperty "ProductCode"
```

This will read a value of `ProductCode` and save it in `productCode` variable. Similar functionality can be also achieved by a direct access to the MSI session.

Writing to MSI Properties

In order to save the value back to the MSI:

```
Set-MSIProperty "MyCustomProperty" "Value Of My Property"
```

Similar functionality can be also achieved by a direct access to the MSI session.

Expanding Strings

In order to expand an MSI string (for example to get the path to a file `myFile.exe`):

```
$expandedValue = $msiSession.Format("[#myFile.exe]")
```

Working with Snippets

Default installation of RayPack has a set of snippets preinstalled.
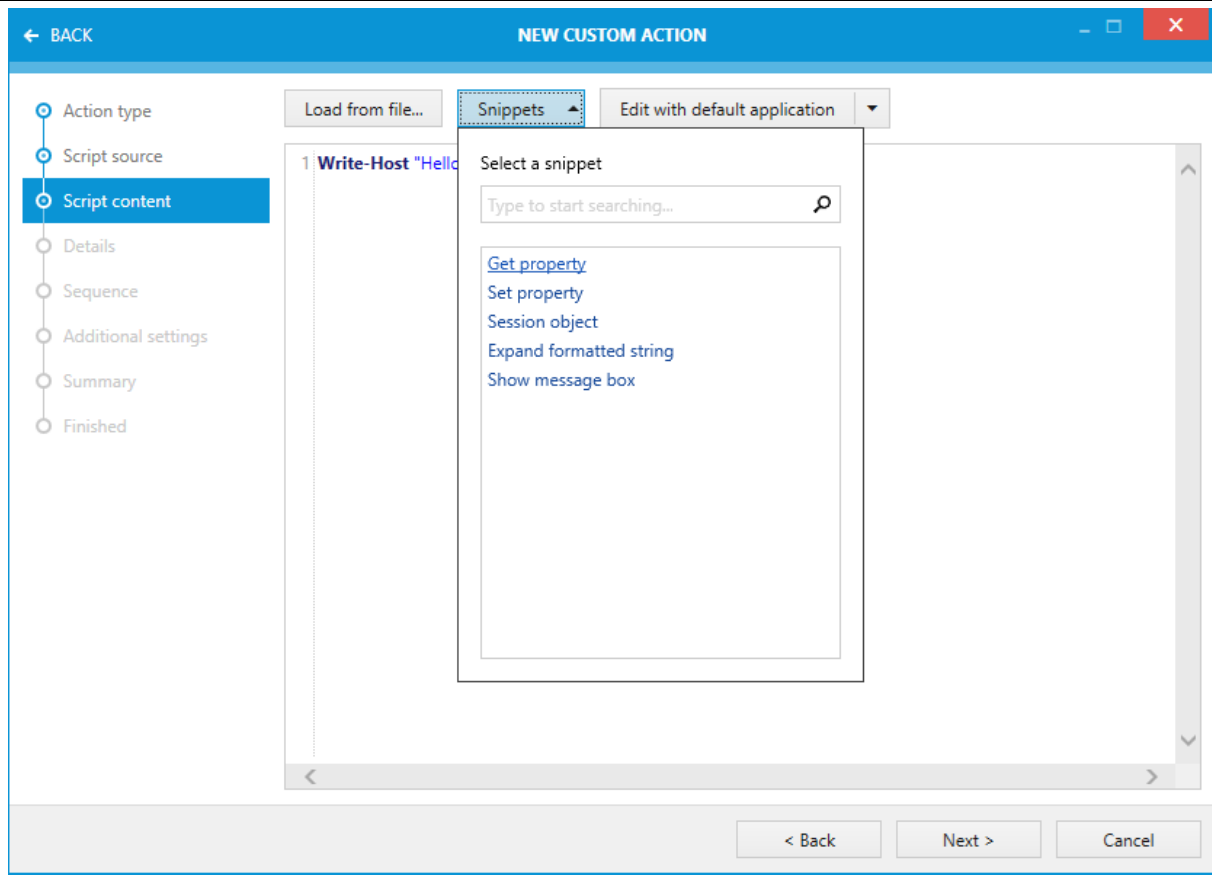
Pressing the name of the snippets inserts its content to the editor. Hovering the mouse cursor over the name reveals the content to be inserted.

Snippets are stored in the following location:

```
<PackPoint>\Snippets\PowerShellSnippets.xml
```

Each snippet has a name and a content. Use any XML or text editor to add or customize the default set of snipp

Remove a Custom Action

From the **CUSTOM ACTIONS** view, removing a custom action can be triggered by **right-clicking** its list item on the left-hand side, and selecting **Remove** from the **context menu**

When the remove procedure is triggered, a confirm dialog is displayed.

> **Be aware:**
> Information that is stored only locally within the custom action, such as error message contents and inline script source codes, will be deleted along with the custom action object. Objects that have been referenced, such as properties or files, will remain part of the packaging project.

To execute the remove procedure, users click **REMOVE**. Please be aware that removed custom

actions cannot be restored. When a custom action is removed, it is automatically deleted from any installer sequences it had been part of.

Users may exit the confirm dialog without removing the custom action by clicking on **DO NOT REMOVE** or **CANCEL**.

Editing a Custom Action

In order to edit a custom action from the **CUSTOM ACTIONS** view within the Advanced mode, its current settings have to be loaded into the details pane on the right. To do so, users select the custom action from the list view on the left-hand side.
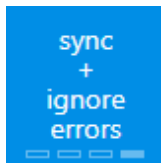
If the **tile based** details view is active, users may edit some of the custom action settings directly by clicking on the tile that displays the current setting. For all tiles that display a set of state indicating rectangles at the bottom, e. g. execution mode and behavior as shown on the left, the different options may be set by simply clicking the tile until the desired value is displayed.

To access the full set of custom action properties for manipulation, users have to click on the black title tile. The **CUSTOM ACTION PROPERTIES** dialog will be displayed.

If the **table based** details view is active, users have to use the button Edit this custom action to display the **CUSTOM ACTION PROPERTIES** dialog.

The CUSTOM ACTION PROPERTIES Dialog

The custom action properties dialog allows packagers to manipulate the full scope of settings related to the custom action object. The dialog is divided into tabs:

- **General** - changing settings for name, run mode and activity related behavior
- **PROCESSING AND EXECUTION** - changing settings for execution mode, patch options and execution related behavior

The dialog shows a set of buttons at the lower right corner, allowing users to:
- Save changes and close the dialog by clicking on the **OK** button
- **APPLY** changes whilst the dialog is kept open
- **CLOSE** the dialog without trying to save the changes

Tab: GENERAL

Name

The name of the custom action may not be left empty, has to be unique throughout the packaging project, and should not contain special characters. Please make sure to use a name value that allows you to quickly identify the custom action intention and purpose. Keep in mind that abbreviations may be hard to remember later, and that other packagers that might have to edit your packaging project in the future may not know them.

Run Mode

- **Immediate** - Custom actions that set properties, feature states, component states, or target directories, or that schedule system operations by inserting rows into sequence tables, can in many cases use immediate execution safely.
- **Deferred** - Custom actions that change the system directly, or call another system service, must be deferred to the time when the installation script is executed.
- **Rollback** - The rollback procedure runs as a clean-up whenever an installation fails to be executed. Custom actions that are marked to run during rollback will not be executed during the primary installation.
- **Commit** - The commit procedure runs to finalize successful installations. A commit custom action is the complement to a rollback custom action and can be used with rollback custom actions to reverse custom actions that make changes directly to the system.

The run mode may not be changed for custom actions of type set folder, set property, and error message.

User Context

- **Run as system** - When custom actions are defined to run as system user, the properties of some user related properties and paths are affected, such as the user related AppData directories and current user registry targets.
- **Run as current user** - This standard setting impersonates the custom action to the user that actually installs the package.
- **Terminal server aware** - Required only in terminal server based target environments.

The user context may not be changed for custom actions of type set folder, set property, and error message.

The additional controls shown depend on the actual custom action type, and therefore vary:

Type: Executable

Type
The following four basic types are available for executable custom actions:

- **Executable from a binary source**
  Requires an executable resource object within the Binary table.
  Please refer to the help section about *Resources* to get details on how to manage binary resources via the Visual Designer interface.

- **Executable deployed within this package**
  Requires an executable file object within the File table.
  Please refer to the help section about *Files and Folders* to get details on how to manage files via the Visual Designer interface.

- **Executable pointed by a property**

Requires a property containing a path that points to a specific file, which may either be stored within the current packaging project or on the target machine.
Please refer to the help section about *Properties* to get details on how to manage properties via the Visual Designer interface.

- **Executable on destination**
Allows to define a path to an executable file.

Stream, File, Property, or Directory
The type selected above controls, which setting actually has to be defined. Changing the custom action type immediately switches the controls to enable stream file or property selection, etc..

Command or Arguments
Once again, the type selected above decides whether arguments or a specific command is required.

Type: DLL

Type
The type selection controls whether the DLL will be provided as resource stream or deployed file. Switching the type automatically changes the control for the source definition.

Stream or File
Select the binary resource or file deployed along with the package. Please make sure to provide a valid dll reference.

DLL Entry Point
The DLL has to be called through the entry point, passing the handle to the current install session as argument. The specified entry point must match that exported from the DLL.

Type: VBS or JS Script

Type
The four available script types are
- Binary stream resource
- Inline script source code
- Property value
- File deployed within the package

Base for the actual script content
According to the selected type, users are requested to select or define the actual script content or reference.
Using the **Edit the script** link calls an additional **SCRIPT** tab, only available when script based custom actions are edited. This tab offers the existing source code as changeable script code, and contains an additional input control for the definition of the initiating function.

Type: Set Property and Set Folder

Property or Folder name

Select the property or folder that has to be manipulated.

New value

Define the new value for the property or folder that has to be set at custom action run-time

Type: Error Message

The message actually displayed as error information may be changed. Please keep in mind that the message text cannot be empty.

Tab: PROCESSING AND EXECUTION

The collection of actually available controls depends on the type and configuration settings for the custom action. The tab may contain any combination of the following property controls:

The following optional behavior settings may be activated:

Execution Mode

- Always execute
- Execute only if running on client after UI sequence
- Execute only once
- Execute only once per process

Run During Patch Uninstall

This option is available for Installer versions later than 4.0. It may be used to exclusively execute custom actions on patch uninstall procedures. To read details about the scenario for patch uninstall custom actions, refer to *MSDN*.

Run as 64-bit Script

On 64-bit operating systems, Windows Installer may call custom actions that have been compiled for 32-bit or 64-bit systems. A 64-bit custom action based on Scripts must be explicitly marked as a 64-bit

Check Exit Code Upon Finishing

If this option is activated, the custom action will be considered as finished successfully if the return code 0 is given. All other return values express erroneous custom action execution. Deactivate this option to ignore the return code and always proceed with the procedure the custom action is a part of.

Run this Action Asynchronously (in the Background)

This option may be set for executable and DLL based custom actions. If it is set, the execution of the parent sequence is continued whilst the custom action logic is still executed.

Sequencing

The **SEQUENCING** view is designed to provide full access to the procedures called during the execution of the five MSI standard sequences:

- **Installation**
  - o Execution sequence - internally represented by the rows stored within the `InstallExecuteSequence` table
  - o UI sequence - internally represented by the rows stored within the `InstallUISequence` table

- **Administration**
  - o Execution sequence - internally represented by the rows stored within the `AdminExecuteSequence` table
  - o UI sequence - internally represented by the rows stored within the `AdminUISequence` table

- **Advertisement**
  - o Execution sequence - internally represented by the rows stored within the `AdvExecuteSequence` table

Each sequence is built from a broad set of so called standard and / or custom actions. Standard actions are generally defined by the MSI standard, whilst custom actions may be defined by packagers in order to achieve highly complex package behavior requirements. Each action added to a sequence may be regarded as one step on the path that is drawn by the whole sequence run.
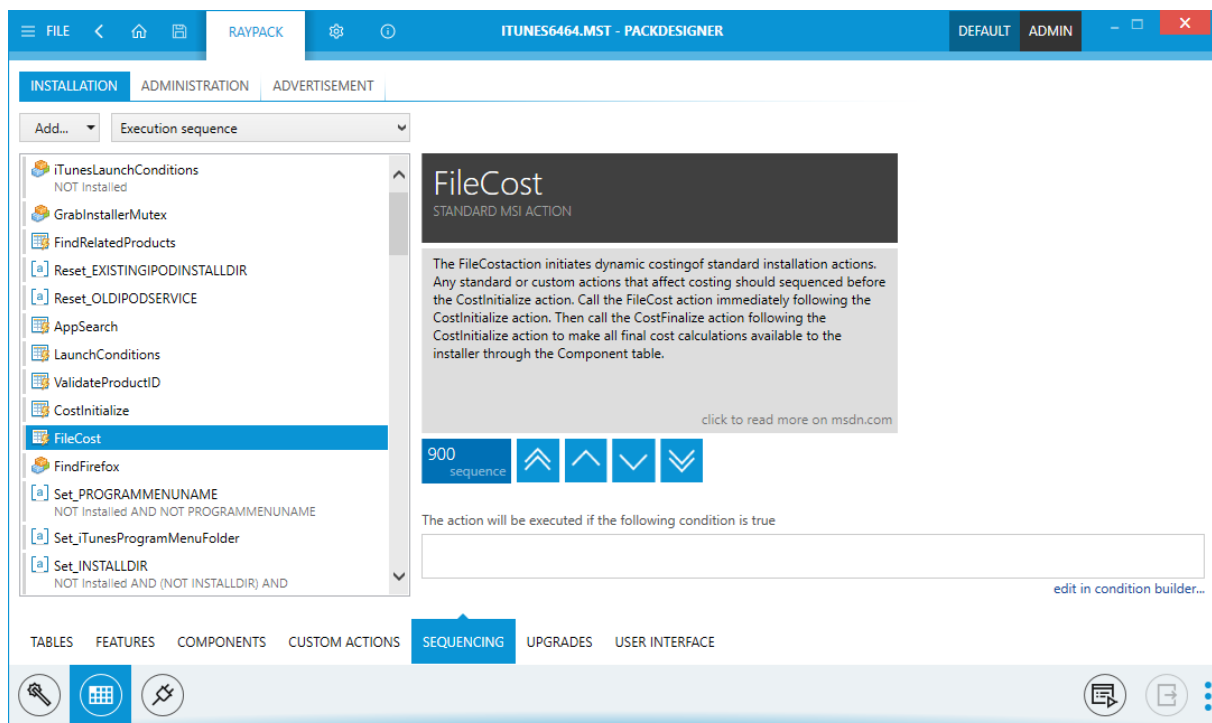
The **SEQUENCING** view is separated into a sequence navigation area on the left-hand side and a details pane for action settings on the right. From the tab menu on the upper left area of the **SEQUENCING** view, users may choose the actual set of sequences they want to work on: Installation, Administration of Advertisement. When either the Installation or Administration tab is active, an additional selector is presented to switch between Execution and UI sequence.

The combination of tab and sequence selector defines the data source for the actual sequence actions displayed in the list view below the navigation elements.
Each action is displayed with its name and an icon, indicating the type of action:

Standard action

Dialog

Custom actions are marked with a broad variety of icons, since each custom action type has got it's own icon.
For example:

Custom action - type call `.dll`

Custom action - type set property

Whilst experienced packagers know how to re-design the sequence choreography in order to get a working and valid result, unexperienced packagers should stay to the MSI standard as close as possible in order to be able to maintan target package validity. Issues caused by re-ordered and manipulated sequences are hard to track and even harder to repair.

However, with RayPack, users have full access to the sequence steps and may execute one of the following manipulation options on sequences and their single steps:

- *Add a sequence action*
- *Reorder a sequence action*
- *Remove a sequence action*
- *Edit a sequence action*

Add a Sequence Action

In order to add an action to a specific sequence, users have to select the required target sequence first. This is done by clicking on one of the sequence navigation tabs (Installation, Administration, Advertisement) on the upper left corner of the SEQUENCING view, followed by the selection of the actual sequence type (execution or ui).

Once the right target sequence is selected, the list view on the left displays all actions that have already been added to the sequence.

The **ADD** button on the upper left corner of the **SEQUENCING** view reveals an options menu when it is clicked, offering different types of actions that may be added to the sequence:

- **Predefined standard action**
  Clicking this option opens a dialog for standard action selection. Simply pick one of the listed actions and click **OK** to add it to the sequence. Each standard action has a pre-defined default position within the sequences, therefore RayPack adds them automatically at that default-position.

  Standard actions are pre-defined action sets which are delivered along with the RayPack resources. These actions contain the logic to execute repeatedly required activities during MSI sequences. Some of them are basically MSI standard items (e. g. `ResolveSource`), whilst others are added as internal RayPack system logic (e. g. `RPTextReplacementData`)

- **Existing custom action**
  Clicking this option from the menu displays a list of already created custom actions for selection. Simply pick one of the listed actions and click **OK** to add it to the sequence. Custom actions do not have pre-defined default positions within the sequences, therefore RayPack adds them automatically at the first position of the currently selected sequence. Users have to *reorder the action* to make sure it is executed during the desired phase of the sequence.

  Custom actions provide procedures that have been defined as specific logic prepared for execution within the current packaging project.
  Please refer to the help section about the **CUSTOM ACTION** view to get details regarding custom action management within RayPack.

- **New custom action**
  Clicking this option from the menu calls the **Custom Action Wizard**, and therefore allows users to add a new custom action.
  Please refer to the help section about *adding new custom actions* for details.

  One of the custom action wizard steps contains an interface for the definition of sequence positions for the new custom action. If these settings are defined during the wizard run, RayPack will add the new custom action at the specified position within the activated sequences. If no sequencing was pre-defined during the wizard execution, RayPack does not add the newly created custom action to any sequence at all, not even the currently displayed sequence!

- **Dialog**
  Clicking this option from the menu displays a list of already prepared standard dialogs that may be added to sequences. To add one of them, users simply select the desired one from the list on the left, and click **OK**. The new dialog is automatically set to the position right before the first non-dialog action is executed. Please note that predefined dialogs may be added only once per sequence. If further dialogs (e. g. error message dialogs) are required, they have to be defined as **custom actions**.

  From time to time the user interface that is shown during a sequence execution has to be extended by an additional standard dialog as required by the results of conditional statements evaluation or other interactive target machine or user specific circumstances.

Once an action has been added to a specific sequence, users may *edit* the *conditions* it is bound to, or the *position* of the action within the sequence step order.

Reorder a Sequence Action

In order to reorder the actions organized within a specific sequence, users have to select the affected sequence first. This is done by clicking on one of the sequence navigation tabs (Installation, Administration, Advertisement) on the upper left corner of the SEQUENCING view, followed by the selection of the actual sequence type (execution or ui).

The actions that build a specific sequence will be executed exactly in the order in which they are listed in the sequence action listing on the left-hand side of the SEQUENCING view. Therefore, adjusting the order of actions here automatically adjust the actual sequence execution.

There are several ways to change the position of an action within a sequence:

- By free positioning via **drag & drop**

    To freely define the new action position, users have to:

    1. **Left-click** on the item that represents the action within the sequence step list view on the left.
    2. Keep the mouse key pressed, and **drag** the action object to the desired position within the sequence. A **bar marker** indicates where the action would be positioned if the user would stop pressing the mouse key.
    3. Stop pressing the mouse key when the desired target position is reached to **drop** it there.

- By relative positioning **one step earlier or later**

    To move an action one position up or down compared to their current position, users have to:

    1. **Right-click** on the item that represents the action within the sequence step list view on the left.
    2. Select
    Execute earlier to move the action one position up
    or
    Execute later to move the action one position down

    Please note that using this relative re-positioning is only available for actually possible movements - executing an action earlier than first or later than last is simply not possible, and therefore not offered for the action at that specific position.

- By direct positioning to the **first or last position**

    To move an action one position up or down compared to their current position, users have to:

    1. **Right-click** on the item that represents the action within the sequence step list view on the left.
    2. Select
    Execute first to move the action to the highest position
    or

Execute last to move the action to the lowest position

Please note that using this absolute re-positioning is only available for actually possible movements - moving an action to the first (last) position simply does not make sense when the action is already at the first (last) position, and therefore not offered for the action at that specific position.

Moving an action up, down, to the first or last position is also possible by clicking the icons displayed within the details pane when an action has been selected. Please note that the icons are displayed slightly faded and are not clickable if a specific option is not possible for the selected action (e. g. move the first action within a sequence one position up or to the first position)

Execute first

Execute earlier

Execute later

Execute last
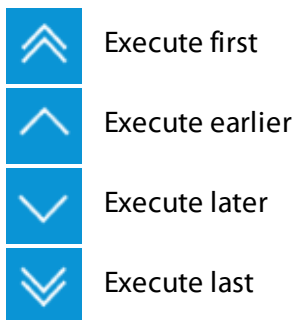
Remove a Sequence Action

In order to remove one of the actions organized within a specific sequence, users have to select the affected sequence first. This is done by clicking on one of the sequence navigation tabs (Installation, Administration, Advertisement) on the upper left corner of the SEQUENCING view, followed by the selection of the actual sequence type (execution or ui).

From the SEQUENCING view, removing an action from a specific sequence can be triggered by **right-clicking** the action and selecting **Remove** from the **context menu.**

	WARNING!
⚠	RayPack does not show a confirm dialog when actions are removed from a sequence, since the action object is not finally deleted from the packaging project itself, but only from this one specific sequence. It is always possible to add the same standard action, custom action, or dialog to the packaging project again.
	However, removing actions from any sequence may cause unpredictable target package behavior at run-time. It is highly recommended to double check any sequence manipulation, in order to make sure no irrevocable harm is done. Please make sure to keep a copy of any original resource material, indicating the original sequence order that might have to be restored at a later time.

Edit a Sequence Action

In order to edit one of the actions organized within a specific sequence, users have to select the affected sequence first. This is done by clicking on one of the sequence navigation tabs (Installation, Administration, Advertisement) on the upper left corner of the SEQUENCING view, followed by the selection of the actual sequence type (execution or ui).

Editing the condition for an action

All types of actions (standard, custom, dialog) may be edited towards the condition statement that is evaluated to determine whether or not the specific action should be executed during the sequence.

To edit the condition of a sequence action, users have to load the action properties into the details pane on the right-hand side of the SEQUENCING view. Below the tiled information about the action, there is a large input field. This is exactly the place to define the conditions that have to be fulfilled, which actually means have to be evaluated to TRUE, in order to execute the action itself during the sequence run-time.

Since conditional statements for evaluation during run-time tend to turn into complex structures, RayPack has its own condition builder, assisting users with pre-defined phrases, snippets and operators. Please refer to the common dialog section about the condition builder for more details.

Manipulating custom actions

Further manipulations are only available for custom actions that have been added to a sequence. To edit one of the custom actions, users have to:

- **Right-click** the custom action item from the list view, and select **Edit** from the context menu.
- **Left-click** the custom action item from the list view, and click on the **black title tile** of the custom action within the details-pane on the right.

The Action Properties dialog is opened, revealing all custom action settings that may be manipulated. Please refer to the specific help topic regarding custom action handling to get detailed information on available settings and options.

Upgrades

The UPGRADES view is designed to manage the upgrade information required by the Windows Installer technology when the product is to be upgraded.
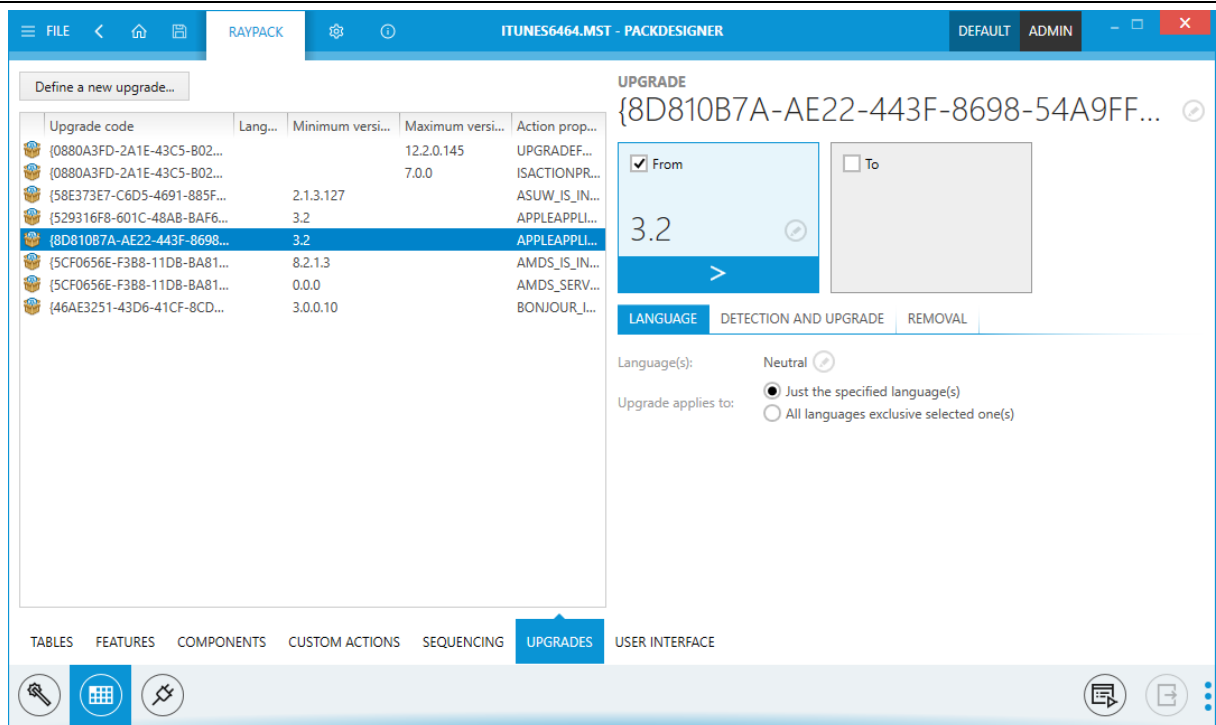
The left area of the UPGRADES view is used to show the list of upgrade entries available within the current packaging project. As soon as an upgrade entry is selected from this list, the details pane on the right-hand side of the UPGRADES view is loaded with the properties defined for the upgrade entry. Users may view and edit upgrade information within the same details pane.

Standard upgrade entry manipulation procedures

The following set of procedures may be initiated from the UPGRADES view:

- Add new upgrade information
- Edit properties of existing upgrade information
- Remove upgrade information

Add New Upgrade Information

In order to add a new upgrade information to a packaging project, go to the Upgrade view of the Advanced mode.

Click the **Define a new upgrade...** button, which is available in the upper left corner of the views content area.

> **Note:**
> If no previous upgrades have been defined so far, the list and sidebar won't be shown. In that the case please click the **Define a new upgrade...** in the center of the screen.

The New upgrade wizard is displayed.

Work your way through the steps of the wizard to define all required properties for the new upgrade information.
At any time, using the NEXT or BACK buttons, which are displayed at the bottom of the wizard dialog, allows to navigate within the already processed steps.
To exit the wizard without creating a new object, use the CANCEL button, also located at the bottom of the wizard dialog.

Step 1: Previous version

The wizard can work in semi-automated or manual mode. In the semi-automated mode, RayPack will read the necessary data from the MSI package containing the old version of the upgraded product. If you have access to the MSI, it is recommended to choose the first option "I have the package for the previous version of this application". If you don't have access to the sources, please select the second option "I don't have the package for previous version of this application". If the second option is selected, you have to manually provide a few details (UpgradeCode, language, versions etc.) in the next page.

Step 2: Details

In semi-automated mode it is necessary to specify the path of the previous package. Click the ellipsis button to open the file browser dialog, select the previous version of the package and press OK. RayPack will automatically load and display the properties of the previous product. You can also change and refine the results manually.

In manual mode, all necessary information (the UpgradeCode, language, version etc.) have to be provided manually.

	Note: The NEXT button will be disabled until all necessary data are entered and are valid. For example, the value entered in the Upgrade code field must be a valid GUID identifier, surrounded by curly braces.

Step 3: Upgrade type

This page is only visible if the "I have the package for the previous version of this application" has been selected on the first screen. There are three types of MSI upgrades available:

- Major upgrade
- Minor upgrade
- Small upgrade

The options available on this screen may differ, depending on the previous package details:

- Small upgrade is available only if the following is true (both must be true):
  - ProductCodes of the old and new application are the same
  - ProductVersions of the old and new application are the same

- Minor and major upgrade is available if any of the following is true:
  - ProductCodes of old and new package are not the same
  - ProductVersions of the old and new package are not the same

Depending on the selection here, RayPack may adjust the ProductCode and UpgradeCode accordingly to make sure that the currently edited project meets the Windows Installer criteria of being major, minor or small upgrade.

Step 4: Synchronization

This page is only visible if the "I have the package for the previous version of this application" has been selected on the first screen.
This page allows performing optional synchronization process. The synchronization reads the data from the new and old MSI package and determines which components are unchanged (the same) in both packages.

Two components are considered the same if (all must be true):

- They have the same resources (files, registries, shortcuts, etc.)
- The versions of files (if applicable) are the same
- The installation folders are the same

If any pair of such components is found, RayPack will try to synchronize the GUIDs of the component by updating the GUID of the component in the current package. By doing so the process of upgrade will be more smooth and reliable, because components having the same content will be also considered the same by the upgrade mechanism (which distinguish components by their GUIDs).

To start the synchronization, click the **SYNCHRONIZE NOW** button.
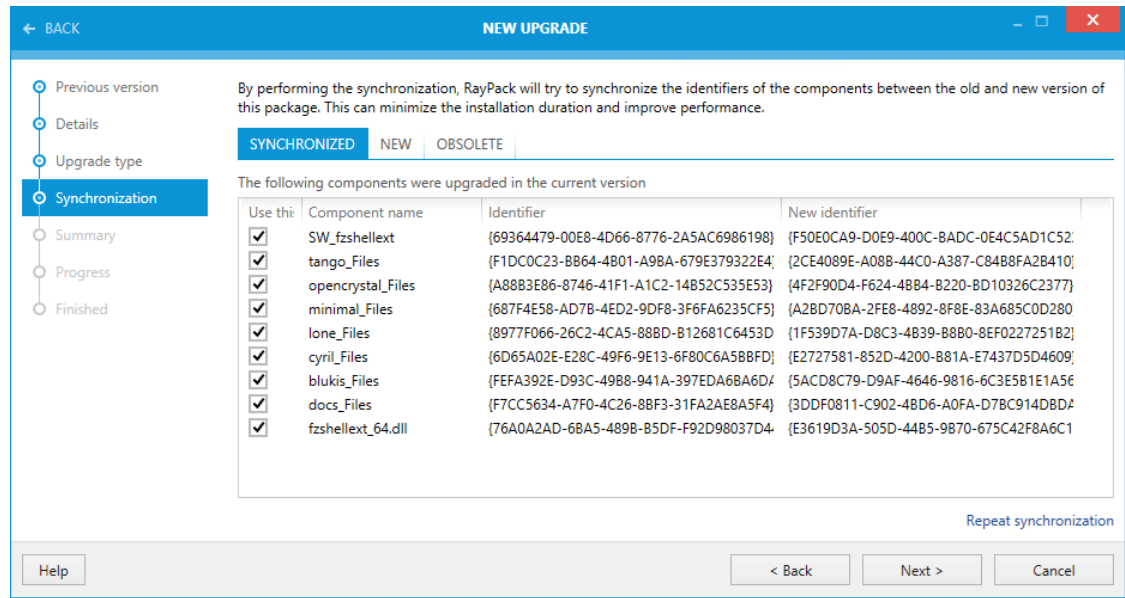
There are three tabs:

- **SYNCHRONIZED**: These components were evaluated and found to be the same in old and new package. The Identifier column shows the current GUID of the component, the New identifier column shows the GUID to which the component will be renamed.

Be aware:
If two same components were found but their identifiers are already the same, they will be not displayed in the *SYNCHRONIZED* tab.

By ticking and unticking the checkboxes you can also control which changes will be actually applied to the current project.

- **NEW**: These components are present in the new package but absent in the old one. Also components that were changed are listed here.

- **OLD**: These component are present in the old package but absent in the new one. Also component that were changes are listed here.

Step 5: Summary

Use the summary page to check the correctness of the upgrade properties that were defined during the previous wizard steps.

- If all properties are set as required, click **PROCESS** to finally create the shortcut
- If changes are due, click BACK until the wizard step with the incorrect property definition(s) is displayed and make modifications as required.

Please note that changes in an early step may lead to different defaults or options in any later

step. Therefore, please verify that all steps contain the desired settings whilst NEXTing to the summary page again.

Step 6: Progress

The upgrade information will be processed and applied to the current packaging project.

Step 7: Finished

Once the new upgrade has been created, the wizard can be closed by using the FINISH button at its lower right corner. The UPGRADES view is updated, and the list of existing upgrades contains the newly created entry.

Edit Existing Upgrade Information

There are two ways of editing upgrade entries in the UPGRADES view of PackDesigner's Advanced mode.

- The basic properties like UpgradeCode, language, minimum and maximum version and the name of the action property can be viewed and edited directly in the table. To enable the edit mode, press **F2** or double click the required value.
- All properties (including these visible in the table) can be also edited directly in the details pane. To do so, users have to **select** it from the list on the left-hand side of the view. Once this is done, the properties listed below are ready for manipulation.

UpgradeCode

The UpgradeCode property is a GUID representing a related set of products. The UpgradeCode is used in the `Upgrade` Table to search for related versions of the product that are already installed.

Language

The set of languages detected affected by the upgrade. This has to be a list of numeric language identifiers (LANGID) separated by commas. If this field is left empty, all languages will be detected by the upgrade mechanism.

> **Tip:**
> This value can also specify the excluded languages (see the LANGUAGE tab).

When the upgrade is edited by the interface provided within the details pane, editing the language becomes fairly easy, as RayPack offers a pre-defined list of available language settings for one-click activation.

Minimum version

Lower boundary of the range of upgradeable product version. If this value is null, all previous

versions will be detected by the upgrade mechanism.

> **Note:**
> Both minimum and maximum versions must be valid product versions. Windows
> Installer uses only the first three fields of the product version. The fourth one is always
> ignored.

Maximum version

Upper boundary of the range of upgradeable product versions. If this value is null, all previous
version greater than (or greater than or equal to) the lower boundary will be detected.
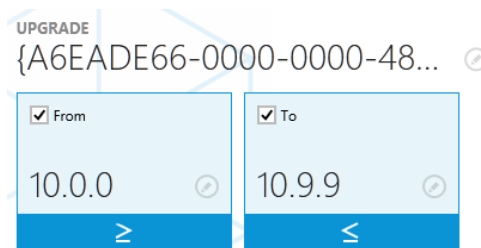
> **WARNING**
> At least one of the versions has to be specified. Both fields cannot be empty at the same
> time.

Action property

The name of the property to which product code of the found upgradable package will be
appended (semicolon is used as a separator when necessary). The property specified in this
column must be a public MSI property and has to be referred by the SecureCustomProperties
property.

The sidebar

Upgrade code

Double-click the name, or click on the edit icon right next to it to use the Direct Value Editor for
changes on the upgrade code.

> **Note:**
> A valid GUID format is required by this field.

From / To

The two tiles can be used to control the required version range. To enable the lower or upper

range, tick the appropriate checkbox.
When the checkbox is checked, the tile will receive a blue focus and provide two additional editing functionalities:

- The product version can be changed by clicking on the version or the pencil icon next to it.
- The blue toggle below controls whether the specified version is in or out of range.

Tab: LANGUAGE

The language tab contains options controlling which languages are included/excluded by the currently edited upgrade entry.
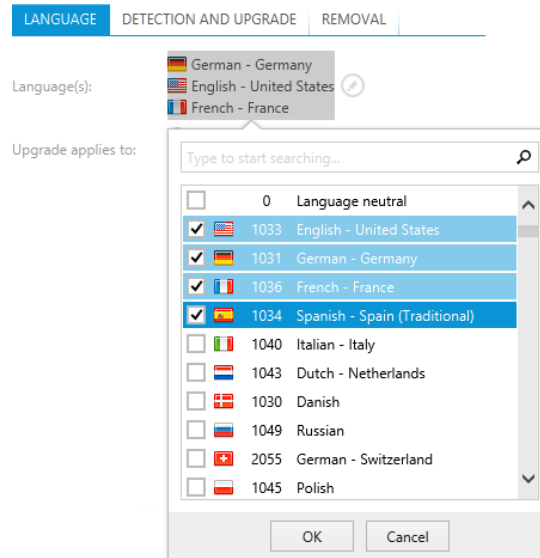
There may be more than one language selected. Use checkboxes to indicate that more languages should be detected by the Windows Installer engine during the upgrade.

Language

Select one of the presented languages. Combined with the following option, this selection determines, which language(s) the upgrade actually targets:

Upgrade applies to

The radio button controls whether the upgrade applies to the selected language version, or excludes all but the selected language version.

Tab: DETECTION AND UPGRADE
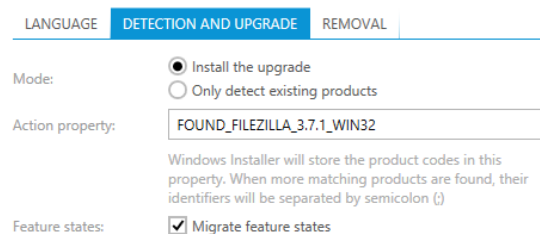
Mode

The radio button controls whether the upgrade entry defines the actual upgrade (Install the upgrade option) or simply the detection of the product (Only detect existing products).

Action property

Specifies the name of the property to which Window Installer appends ProductCodes of found products. Installer uses semicolon as a separator when necessary. The property specified in this column must be a public MSI property and has to be referred by the SecureCustomProperties property.
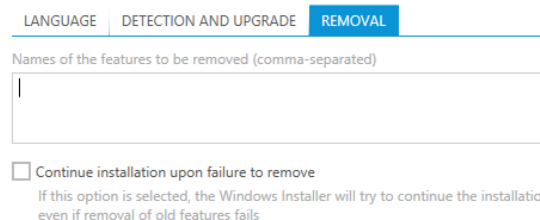
Migrate feature states

The checkbox controls whether the states of the features will be migrated. If the option is enabled, Windows Installer reads the feature states in the existing application and then sets these feature states in the pending installation.

> **Note:**
> The method is only useful when the new feature tree has not greatly changed from the original.

Tab: REMOVAL

Names of the feature to be removed

The textbox controls feature names that will be removed during upgrade installation. To separate feature names, a semicolon (;) has to be used. If the value of this field is left empty, all features from the previous package will be removed.

Continue installation upon failure to remove

If this checkbox is active, Windows Installer will continue the installation upon failure to remove a product or application.

> **Note:**
> The MSI technology imposes several restrictions on the values entered in this view. The upgrade code, version range, language and attributes form together the primary key of

the Upgrade table. RayPack verifies the entered values before saving them and if necessary shows the warning when the required change is not allowed due to resulting not-unique Primary Key.

Remove Upgrade Information

From the UPGRADES view, removing an upgrade entry can be triggered by **right-clicking** the upgrade and selecting **Delete** from the **context menu**.
When the removal procedure is triggered, a confirm dialog is displayed.

To execute the removal procedure, users click REMOVE. Please be aware that removed upgrades cannot be restored.

> **Note:**
> To suppress the display of the confirm dialog for upgrade remove procedures, the checkbox displayed for the confirm dialog control may be activated in advance. If it is activated, upgrade removal procedures will be executed immediately.

Users may exit the confirm dialog without removing the upgrade by clicking on **DO NOT REMOVE** or **CANCEL**.

User Interface

The USER INTERFACE view is designed to offer manipulative access to the dialogs that are part of the user interface belonging to the current packaging project standard installation routine.

Once the USER INTERFACE view is opened, a tile based overview of dialogs is displayed. All dialogs that are contained in the current RPP project or MSI package are displayed. Users may edit and remove dialogs from this collection.

The user interface editor is divided into the following sections:

- Installation Mode selector
- Dialog list / Dialog tree
- Designer canvas
- Current selection properties

Installation Mode selector

Dialog list /
Dialog tree

Current selection
properties

Designer canvas
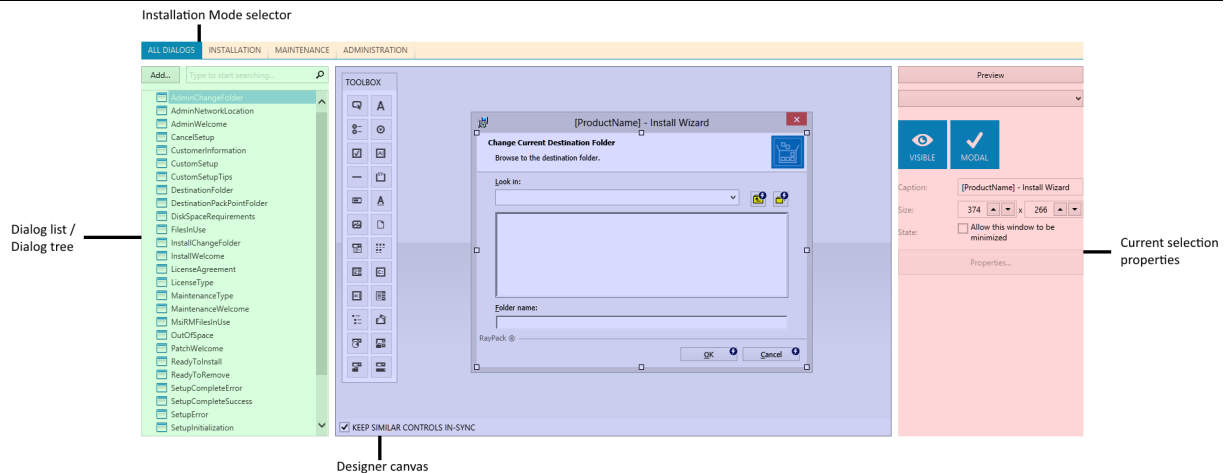
Installation Mode selector

The tabs on the top can be used to change the installation mode to be displayed in the other sections.

- **ALL DIALOGS**

  All dialogs present in the MSI package will be shown.

- **INSTALLATION**

  Only the dialogs visible during installation and uninstallation of the product will be shown

- **MAINTENANCE**

  Only the dialogs shown during maintenance (repair, modify) will be shown.

- **ADMINISTRATION**

  Only the dialogs shown during maintenance (repair, modify) will be shown.

Dialog list

The list of the dialogs is only shown when the option **ALL DIALOGS** is selected. The control contains the alphabetical list of all dialogs present in the MSI package, irregardless of the connections and conditions that were defined for them. This view is especially useful to locate a dialog by its name.

☞	**Be aware:** Adding a new dialog to the project is only possible within the ALL DIALOGS tab.

To filter the list by a name or its part, type the required filter string into the search box.

In order to delete a dialog from the package, right click its name and select **Delete....** RayPack will show the list of all items that depend on the dialog being deleted:

- Controls (from the `Control` table)
- Events (from the ControlEvent table)

In order to confirm the deletion, press **YES**. Otherwise, click on the **NO** button to abort the deletion.

> **Note:**
> Deleting a dialog will remove it from all sequences. All dependent items will be also removed. This operation is irreversible. After the dialog is deleted it may be necessary to manually adjust the events to make sure the navigation flow is working as expected.

Dialog tree

The logical tree of dialogs is only shown when any option **but ALL DIALOGS** is selected. The control contains a tree, which simulates in which order the dialogs are shown to the end user. This view is especially useful to locate a dialog without knowing its name, but knowing when it gets displayed.

> **Be aware:**
> Due to the Windows Installer simulation, RayPack only simulates the logical tree by examining the dialog conditions, events and actions. For very complicated expressions, the connections between dialogs may be misrepresented or not shown.

The tree displays the dialogs on three levels:

- Root items are representing the actual order of actions, as present in the `UIExecuteSequence` table.
- The children of root items show a simulated flow and order of dialogs as presented to the end user.
- The third level of dialogs show modal dialogs or popups shown by the installation

If a dialog links to a dialog which has been already displayed above, it is now shown anymore in the tree to avoid duplication. If a dialog links to a dialog which has not occurred before, it will be shown in the tree as a next item in the tree.

If a dialog is present in the `UIExecuteSequence` and references another dialogs, it will be shown in a separate group node named after the first entry dialog that is present in the sequence.

Designer canvas

This is the central place where dialogs are edited and previewed. The canvas contains a 1:1 representation of the dialog being currently selected in the Dialog tree / Dialog list and the toolbox allowing to add new controls to it.
The existing controls can be edited by using either the sidebar or the properties dialog, available from the context menu.

Current selection properties

The sidebar contains a quick overview of basic properties of the current selection. If a control (for example a button or a bitmap) is selected, then its properties are shown in the sidebar. In any other case, when nothing is selected, the properties of the current dialog are shown.

Depending on the selection, the layout and amount of information present in the sidebar may vary. For most of the controls, standard properties that are available are:

- Visibility control
- Enabled/disabled state control
- 3D/Flat look control
- Size (X and Y coordinates)

Adding New Dialogs

In order to show a new dialog during installation:

1. The dialog has to be added to the Installer database (into the `Dialog` table)
2. There must be a link between any control in the database which will call and navigate to that dialog

To add a new dialog to the project

1. In the ALL DIALOGS view, click on *Add...* button
2. A list of predefined dialogs will be shown. Select a dialog from the list that will be a base for a new one.
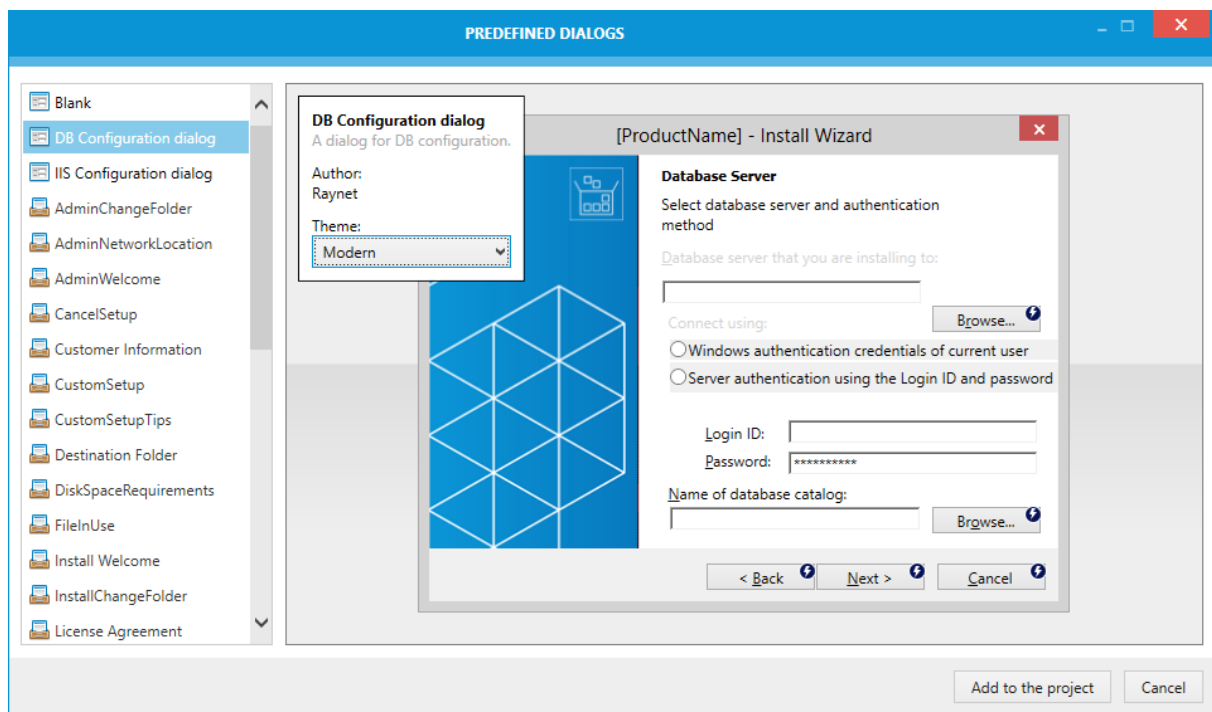
The dialog list is sorted:

1. On the top there is always a template for a blank dialog. The blank dialog comes in two flavors - one with navigation buttons and basic graphics, and one without anything for custom layouts
2. Custom dialogs sorted alphabetically
3. Standard MSI dialogs sorted alphabetically (to distinguish them, RayPack shows a different icon for these entries).

In order to add the selected dialog to the project, press the *Add to the project* button.

Themes

For some dialogs, it is possible to change the Theme. The theme can be changed by selecting a required one from the combobox available in the toolbox.

Defining Custom Dialog Templates

> **Note:**
> This is a topic for advanced users.

The dialogs available in Predefined Dialogs browser can be customized and extended. RayPack looks for the dialog definitions in the following folder:

```
<PackPointDir>\Dialogs
```

By default the `C:\RayPack\PackPoint\Dialogs` directory is being used.

Each dialog is defined using at least two files:

- A small `*.xml` file, defining the basic dialog properties: name, description, manufacturer and the source file. Optionally, themes can be defined for each dialog as well.
- An `*.rpd` file containing the necessary dialog data, images, actions, events etc.

For example, the SetupCompleteError dialog is defined in a following way:

```
<?xml version="1.0" encoding="utf-8" ?>
<Dialog IsStandard="false" Name="SetupCompleteError" Author="Raynet"
Source="SetupCompleteError\SetupCompleteError.rpd">
  <Description><![CDATA[A dialog for SetupCompleteError configuration.]]>
  </Description>
</Dialog>
```

- The `Source` attribute defines a relative location of a file, containing the dialog and controls definition.
- The `IsStandard` attribute can be either `true` or `false` and affects whether the dialog is displayed as a custom one or as a standard one in the predefined dialog browser.

The `*.rpd` files are simple MSI databases with just extension changed. They do not contain the full structure of MSI schema, usually only the following tables are present inside:

- `Binary`
- `Control`
- `Dialog`
- `ControlCondition`
- `ControlEvent`
- `TextStyle`

When the dialog is added to the project, the content of all tables are merged with the content of the current project. If any conflict appears (for example trying to add a binary resource that already exist in the package) it is silently ignored. This is to make sure that any kind of customization is preserved when importing a dialog. There is a special handling for the `SecureCustomProperties` property. Its value is merged with the base one using semicolon as separator. For any other MSI Property, the value present in the current project wins.

Preparing a custom template

In order to prepare a custom template:

1. Create a subfolder in the `<PackPointDir>\Dialogs` folder (by default `C:\RayPack \PackPoint\Dialogs`), for example `MyDialog`.
2. Create a new project in RayPack, use the User interface editor to create a custom dialog
3. Build the project as MSI into the subfolder created in the first point. By convention, the name of the file should be the same as the name of its folder, but you can enforce your own naming conventions.
4. Remove any unnecessary leftovers from the project.
   a. Anything that has nothing to do with the dialog should be removed. Ideally, only the tables mentioned above should be left inside, and only rows describing the new dialog should be left.
   b. Remove the entries from the `Binary` and Icon table that are not required by the dialog
5. Save the MSI file, and rename its extension to `.rpd`
6. Create an XML file in `<PackPointDir>\Dialogs`. By convention its name should be the same as the name of the `.rpd` file, but you can enforce your own naming conventions.
7. Copy the sample from this section, update the necessary attributes (name, description and author)
8. Update the `Source` attribute to correctly reference the path you saved in point 5.
9. The changes are applied as soon as the *Predefined Dialogs* window is shown.

Creating themes

A theme is a simple MST transform with `.rpdt` extension. In order to create a theme for a dialog:

1. Pick up a dialog or create a new one using the previous steps.
2. Create an MST transform to the original `.rpd` file
3. The transform should change the necessary parameters, add required images etc.
4. Save the transform in the subfolder where the original `.rpd` file is
5. Adjust the XML file containing the dialog definition. For example, the sample below defines two themes:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<Dialog IsStandard="false" Name="DB Configuration dialog"
Author="Raynet" Source="db\db.rpd">
    <Description><![CDATA[A dialog for DB configuration.]]></
Description>
    <Themes>
      <Transform Name="Wide" Source="db\Wide.rpdt" />
      <Transform Name="Modern" Source="db\Modern.rpdt" />
    </Themes>
</Dialog>
```

6. The changes are applied as soon as the *Predefined Dialogs* window is shown.

Defining Transitions Between Dialogs

The transitions between dialogs can be defined in two different ways:

1. In Advanced View > Tables, by editing `ControlEvent` and `UIExecuteSequence` tables
2. In Advanced View > User interface, by editing the Control Events for buttons

To define a transition between two dialogs

1. Locate a button that will trigger the transition
2. Right click the button and go to the *Properties > Events* tab.
3. Press `Add...` to add a new event
4. Change the value in the *Event* column to:
   a. `SpawnDialog` to show a modal window
   b. `NewDialog` to simply go to the new dialog
5. In the *Arguments* column, type or select from the list the name of the dialog to go to
6. You can also optionally redefine the *Condition*. By default its value is 1, meaning the link will always work, regardless of any condition. By changing this value you can define different transition paths, depending on installation conditions, properties, states etc.

Editing the Dialog Canvas

The dialog canvas provides several ways to change the layout of controls / dialogs:

- Resizing the dialog
- Moving the controls
- Resizing the controls
- Aligning the controls
- Deleting the controls
- Adding new controls.

To resize a dialog

Make sure no control is selected. This can be performed in two ways:

- Hold the `Ctrl` button and left click the currently selected control, or
- Left click on any place in the canvas belonging to no control

When no control is selected, thumbnails are displayed around the current dialog. Left click any of them, and move your mouse around without releasing the left mouse button.
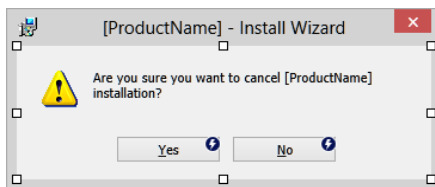
To accept the new size, release the left mouse button.

To move a control

Select the control by clicking on it with a left mouse button. Small thumb buttons will appear around the control to indicate the selection is active:
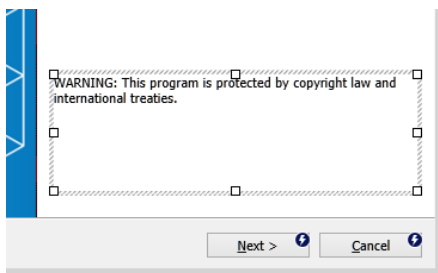
Drag and drop the control around to change its position. When the control is moved around, sometimes alignment guides are visible. They help you to align left/top/right or bottom edge with other controls. For example, when moving the NEXT button, the following may be shown:
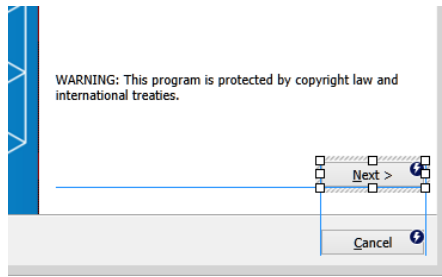
The blue indicate that the new position is aligned with the left and right edge of the CANCEL button, and the bottom edge of the WARNING text.

To adjust the position, release the left mouse button while dragging.

> **Note:**
> The control cannot be positioned outside of the current dialog.

To resize a control

Select the control by clicking on it with a left mouse button. Small thumb buttons will appear around the control to indicate the selection is active. Left click any of them and drag the mouse while the left button is still pressed. The control will be resized accordingly. The aligning guides may be also shown, indicating that alignment to an edge of existing control is possible:
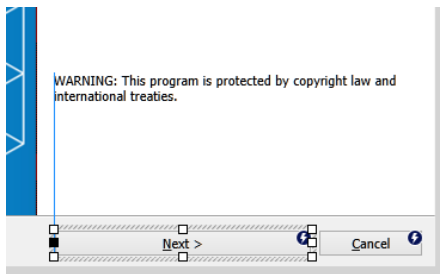
Release the left mouse button to accept the new size.

> **Note:**
> The control can be only resized so that it doesn't exceed the dialog canvas,

Special Indicators

RayPack shows additional indicators which are not visible when the actual Installer session is started. They help to navigate around and make it easier to edit the advanced connections between packages.

Events indicator

If a control is enclosed with a "bolt" icon, then there are some event attached to it. For example, the following picture suggests that three navigational buttons (*Back*, *Install*, *Change* and *Cancel*) have a special functionality inside and are publishing events to control the installation.
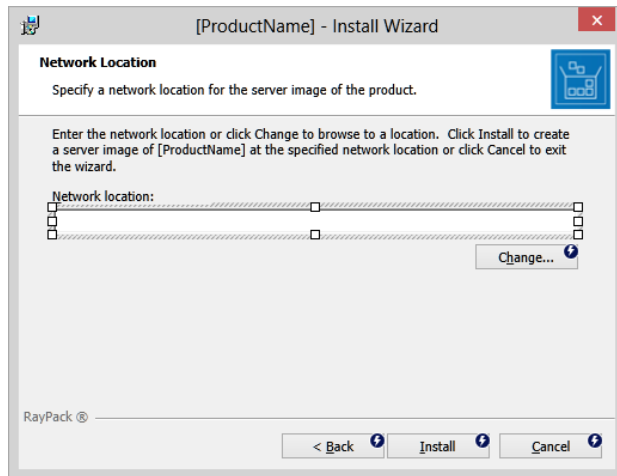
In order to see and edit the events, right click a control and click on *Properties...* item. More information about editing the events can be found in this chapter.

Visibility indicator

Controls that are not visible by default are displayed using a partially transparent layer.
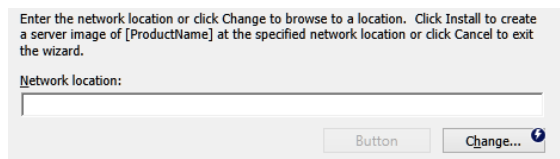
For example, the *Button* control is hidden in the picture above, while the *Change...* button is visible.

MSI Property placeholders

When an MSI Property is used, its value is shown unresolved in the dialog designer. For example, the picture below shows `[ProductName]` placeholder to indicate the value in this field is dynamic.
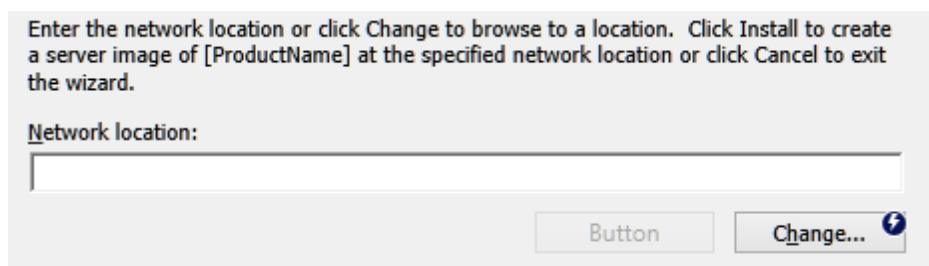
Aligning Items

The items can be aligned in two ways:

1. By dragging and dropping / resizing the controls

Refer to the Editing the dialog canvas section for more information about guidelines and aligning.

2. By selecting more than one element and choosing the alignment options

When more than one control is selected, the sidebar shows the alignment options:
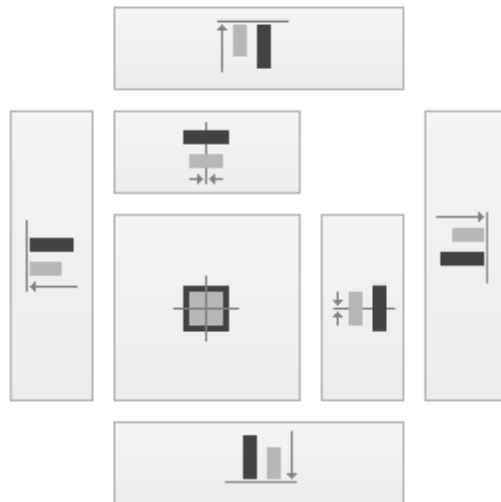
Use the buttons to align all selected controls to left / right / top / bottom, or center them vertically / horizontally or both vertically and horizontally.

Keeping Controls in Sync

Typically, several controls should share similar characteristics across more than one control. Some examples include:

- The same position of navigation buttons (NEXT, BACK, CANCEL)
- The same position of background bitmaps
- The same bitmap for branding images
- Etc.

RayPack contains a mechanism that allows to synchronize the changes made to one dialog with other dialogs. The synchronization will work if all the conditions below are true:

1. The controls are having the same initial size and position
2. The controls have the same identifier
3. The controls are of the same type

For example, if a NEXT button has been defined on 5 different dialogs and the following is true:

1. On each dialog, the button is called `NextButton`
2. On each dialog, the button is located in the same place and has the same size

Then by resizing a button in one dialog all other NEXT buttons will be resized as well. This behavior is automatic and requires no settings.

Disabling the synchronization of controls

In order to disable the synchronization, one of the two following actions can be performed:

1. Rename the button and use a unique name for it. This way, the changes made to that button will not be synchronized anymore. The name of the button can be changes in the tables view, in the Control table. You can also right click a button and press "Go to row..." to highlight the necessary row automatically.

2. You can also disable the synchronization for all controls. The checkbox *KEEP SIMILAR CONTROLS IN SYNC* located under the designer canvas can be used to control whether the synchronization takes place at all. It can be re-enabled at any time.
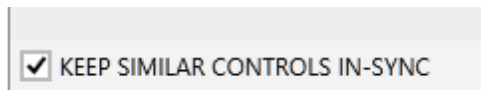
Editing the Basic Properties

The sidebar contains basic information about the currently selected dialog / control.

The content presented in the sidebar may vary, depending on current selection. Some properties are commonly found for a number of controls:

- **VISIBLE / HIDDEN**

  Defines whether the control is visible or hidden during installation. A condition can be used to dynamically change this value on runtime. The controls that are hidden are still shown in the designer, but to distinguish them from visible controls a transparent layer is used for them.

- **ENABLED / DISABLED**

  Defines whether the control is enabled or disabled during installation. A condition can be used to dynamically change this value on runtime. The actual impact of this setting may vary depending on the type of the control. For example, disabled textboxes do not receive focus and no value can be typed in. A disabled button cannot be clicked. Also, the appearance of a disabled control may be different, for example a button is grayed-out, but a bitmap does not have any visual differences between enabled and disabled state.

- **FLAT / SUNKEN**

  Defines whether the control is displayed using a flat style or sunken style (also known as 3-D). The actual appearance of the control may vary, for example a sunken text control receives additional border, but there is no visual difference between sunken and flat bitmap.

More properties may be available depending on the type of the current selection. For example a button (pictured above) contains additional fields for a style and the label (text).

To see / edit more properties, click on the *Properties...* button to open the Properties dialog.

Editing access keys

For certain controls (for example buttons, checkboxes etc.) it is possible to define access keys.

Editing the Properties of a Control

The *Properties* dialog contains advanced settings and provides a full control over the events and conditions for selected controls.
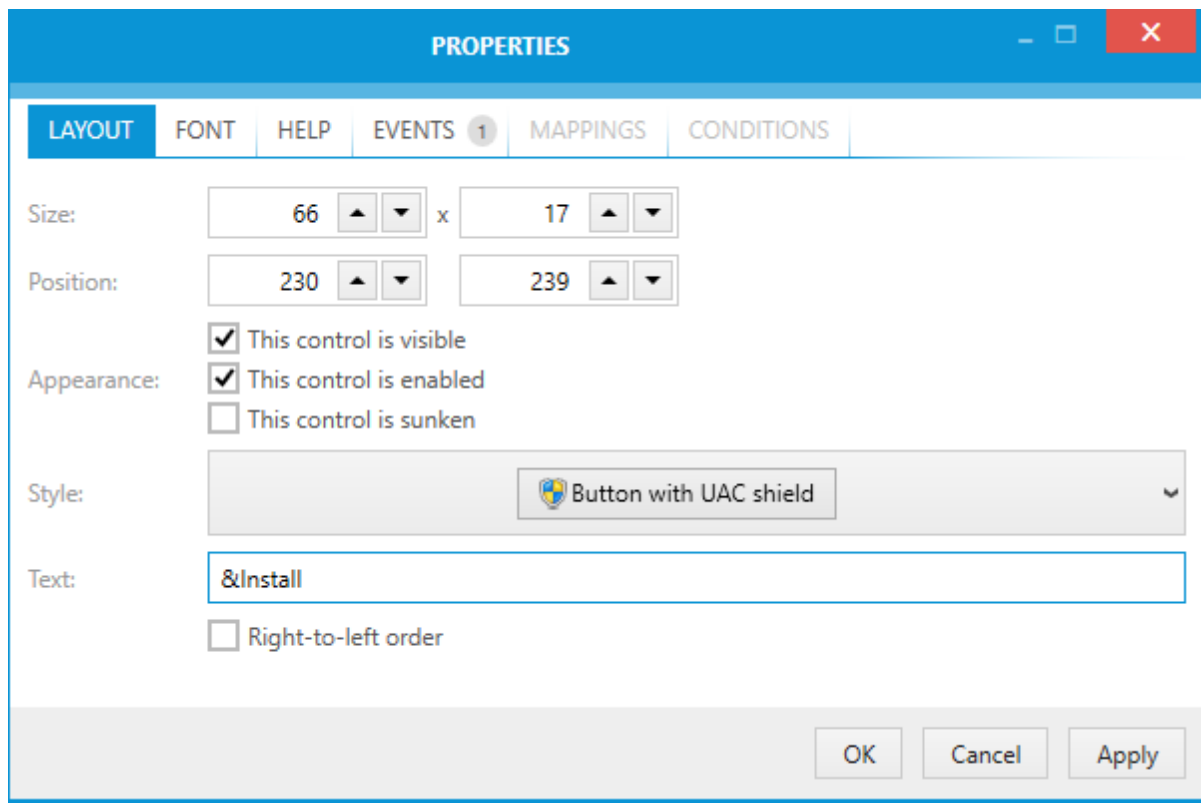
The following tabs are displayed for each control:

- **LAYOUT**
  Basic properties defining the look and feel of a control, including position, size and appearance. For certain controls more options can be changed here. For example, when editing a button (pictured above) the text, right-to-left settings and the button style can be changed.

- **HELP**
  Allows to change a tooltip and an alternate text used by screen readers.

- **MAPPINGS**
  Allows to define the dynamic bindings between various installer events and control properties. For example, a mapping for `SetProgress` event for a ProgressBar control can be used to define that the progress bar should display the current progress value.

- **CONDITIONS**
  Allows to enable, disable, hide, show or set a control as default when a certain condition is met.

Some tabs are only available for specific control types or scenarios:

- **IMAGE** (available for controls displaying an image: bitmap controls, icon controls, buttons etc.)
  Allows to change the source of the image to be displayed by a selected control.

- **FONT** (available for controls displaying text: buttons, checkboxes, texts etc.)
  Allows to change the font to be used by a selected control.

- **EVENTS** (available for buttons, checkboxes and selection trees)
  Allows to change events occurring when a control is clicked.

- **ITEMS** (available for child controls, for example list boxes, combo boxes etc.)
  Allows to add / edit / remove / order the child elements to be displayed in the child control element.

Events

Events are used to perform additional functionality when for example a button is pressed or a selection is changed.
To define an event for a control, select one that supports event publishing, that is:

- A button
- A checkbox
- A selection tree

The *Properties* dialog lets you define the combination of an event, its arguments and condition.

- **Event**
  One of the predefined names recognized by Windows Installer, or a name of the Windows Installer property surrounded by square brackets.

  The following predefined events are available:

  - `ActionData`
  - `ActionText`
  - `AddLocal`
  - `AddSource`

- CheckExistingTargetPath
- CheckTargetPath
- DirectoryListNew
- DirectoryListOpen
- DirectoryListUp
- DoAction
- EnableRollback
- EndDialog
- IgnoreChange
- MsiLaunchApp
- MsiPrint
- NewDialog
- Reinstall
- ReinstallMode
- Remove
- Reset
- RmShutdownAndRestart
- ScriptInProgress
- SelectionAction
- SelectionBrowse
- SelectionDescription
- SelectionNoItems
- SelectionPath
- SelectionPathOn
- SelectionSize
- SetInstallLevel
- SetProgress
- SetProperty
- SetTargetPath
- SpawnDialog
- SpawnWaitDialog
- TimeRemaining
- ValidateProductID

A detailed description of each of these can be found on MSDN website:
https://msdn.microsoft.com/en-us/library/aa368043(v=vs.85).aspx

- **Argument**
  This is a string-based argument that is passed along with the event. Most of the events require a specific value to be present. RayPack correctly recognizes events like `NewDialog`, `EndDialog` etc. and will offer a choice of possible values when editing the value of the *Argument* column.

  If the value of the *Event* is an MSI property enclosed in square brackets, then the value of *Argument* specifies the value that will be assigned to the property.

  Note:
  This field is formatted. This means MSI properties can be used here, for example `[ProductName]` etc.

- **Condition**
  The condition can be specified to fine-tune when the event will be actually raised. For example, it is possible to disable a specific event to be published in case of uninstallation, based on value of MSI property, component state and more. Typing `1` into this column means that the event is always fired.

Adding a new event

To add a new event, click on the *Add...* button. A new row will be added automatically.

Editing an event

To edit an existing event, focus the cell to be edited and press `F2` or click it for a second time using the left mouse button. RayPack will display a combo box editor if the set of allowed values could be determined. The value can be also typed manually. Some fields are additionally validated by RayPack. For example, RayPack prevents entering an empty string into the *Event* column to preserve internal MSI consistency. Any validation errors are shown inline. If you want to cancel editing and restore the previous value, press `ESC` while the validation error is being shown in the cell.

> **Note:**
> Due to the Windows Installer limitations, some combinations of values may produce same MSI Primary Keys. When the changes are saved (by pressing either *OK* or *APPLY* button) RayPack will merge the events definition and ensure that they do not break the internal consistency of the package.

Deleting an event

To delete an existing event, focus the row belonging to the event, and press the button *Remove selected*.

Reordering events

Events can be reordered using drag and drop technique. The higher the event is present on the list, the sooner it gets executed in actual Installer session. To reorder the items, left click an event to be moved, and drag it with the left button still pressed to a desired place.

Event Mappings

Event mapping are used to bind a certain Installer event to a property / behavior of an MSI control.
The *Properties* dialog lets you define the combination of an event and its arguments for all MSI controls.

- **Event**
  One of the predefined names recognized by Windows Installer, or a name of the Windows

Installer property surrounded by square brackets.

The list of accepted mappings is available on MSDN website:
https://msdn.microsoft.com/pl-pl/library/aa368036(v=vs.85).aspx

- **Argument**
  This is a string-based argument that is passed along with the event. Most of the events require a specific value to be present.

Adding a new mapping

To add a new event mapping, click on the *Add...* button. A new row will be added automatically.

Editing an event event

To edit an existing event mappping, focus the cell to be edited and press `F2` or click it for a second time using the left mouse button. Some fields are validated by RayPack. For example, RayPack prevents entering an empty string into the *Event* column to preserve internal MSI consistency. Any validation errors are shown inline. If you want to cancel editing and restore the previous value, press `ESC` while the validation error is being shown in the cell.

> **Note:**
> Due to the Windows Installer limitations, some combinations of values may produce same MSI Primary Keys. When the changes are saved (by pressing either *OK* or *APPLY* button) RayPack will merge the event mappings definition and ensure that they do not break the internal consistency of the package.

Deleting an event mapping

To delete an existing event mapping, focus the row belonging to the mapping, and press the button *Remove selected*.

Conditions

Conditions are used to change the visibility or enabled state of a control on runtime. The *Properties* dialog lets you define the combination of an condition and action.

- **Condition**
  A condition using the Installer syntax.

- **Action**
  One of the five values:

    - Show - makes the control visible when the condition is *true*

    - Hide- makes the control invisible when the condition is *true*

    - Enable- makes the control enabled when the condition is *true*

    - Disable- makes the control disabled when the condition is *true*

    - Default- makes the control default on the current dialog when the condition is *true*

Adding a new condition

To add a new condition, click on the *Add...* button. A new row will be added automatically.

Editing a condition

To edit an existing condition, focus the cell to be edited and press `F2` or click it for a second time using the left mouse button. RayPack will display a combo box editor if the set of allowed values could be determined. The value can be also typed manually. Some fields are additionally validated by RayPack. For example, RayPack prevents entering an empty string into the *Condition* column to preserve internal MSI consistency. Any validation errors are shown inline. If you want to cancel editing and restore the previous value, press `ESC` while the validation error is being shown in the cell.

> **Note:**
> Due to the Windows Installer limitations, some combinations of values may produce same MSI Primary Keys. When the changes are saved (by pressing either *OK* or *APPLY* button) RayPack will merge the condition definition and ensure that they do not break the internal consistency of the package.

Deleting a condition

To delete an existing condition, focus the row belonging to the condition, and press the button *Remove selected.*

Fonts

For certain controls supporting custom fonts it is possible to define which text style should be used.

The text styles are shown as small previews using a tile-based grid:
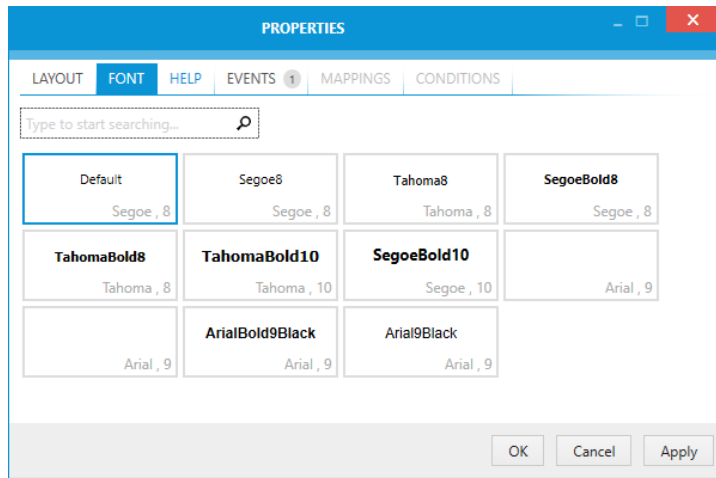
The current font is highlighted using a blue border. Each tile contains a preview showing the appearance of the text style, and a font name and size underneath.

To add a new custom style, use the Tables view and add an appropriate entry into the `TextStyle` table.

Changing the default font

The font and color of the default font is defined using the `DefaultUIFont` MSI Property. To change the default style use the Tables view and edit the value of the property in the `Property` table, or use the Property view in Visual Designer.

Previewing Dialogs

While RayPack tries to represent the edited controls to match the Windows 8 look and feel, some values and run-time settings can be only seen when the actual session is running.

In order to see a preview of the Window, rendered and running by the actual Installer session, press the *Preview* button from the properties sidebar.

For example, the two pictures below show the same dialog, once (the left one) as seen in RayPack while editing, and once (the right one) as seen when the MSI is actually running.
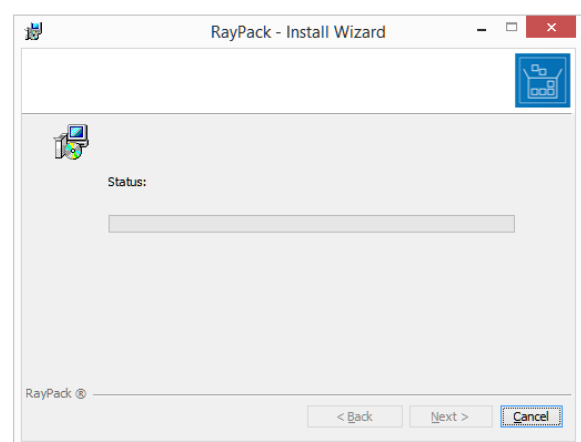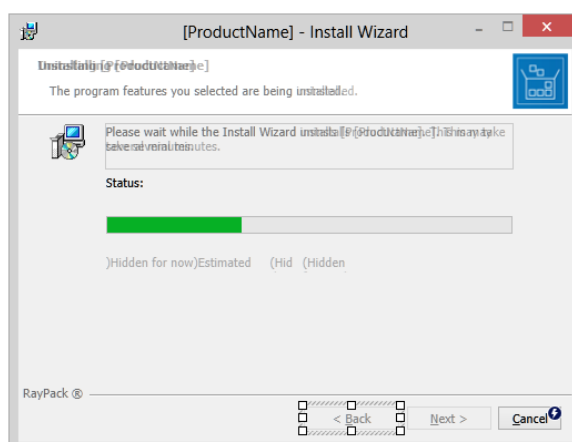
Some things that may require previewing:

- The usage of MSI Properties: left picture - properties are unresolved (`[ProductName]`), right one - properties are resolved (`RayPack`)
- Hidden controls, condition: left picture - some controls are defined as hidden and have dynamic conditions; right one - conditions are resolved and only visible controls are shown

To end preview mode

To end the preview mode, focus the main window of RayPack and press the *CLOSE PREVIEW* button.

> **Note:**
> Pressing the X button in the previewed dialog will not have any result.

Plugins View

The **Plugins** view contains a list of plugins, scripts, and macros that are available to use and / or for configuration. Depending on your installation, you may already have a few plugins preinstalled with your RayPack edition. In order to select the details of a plugin, select its name in the left sidebar. Right panel shows the details of the current selection, for example:
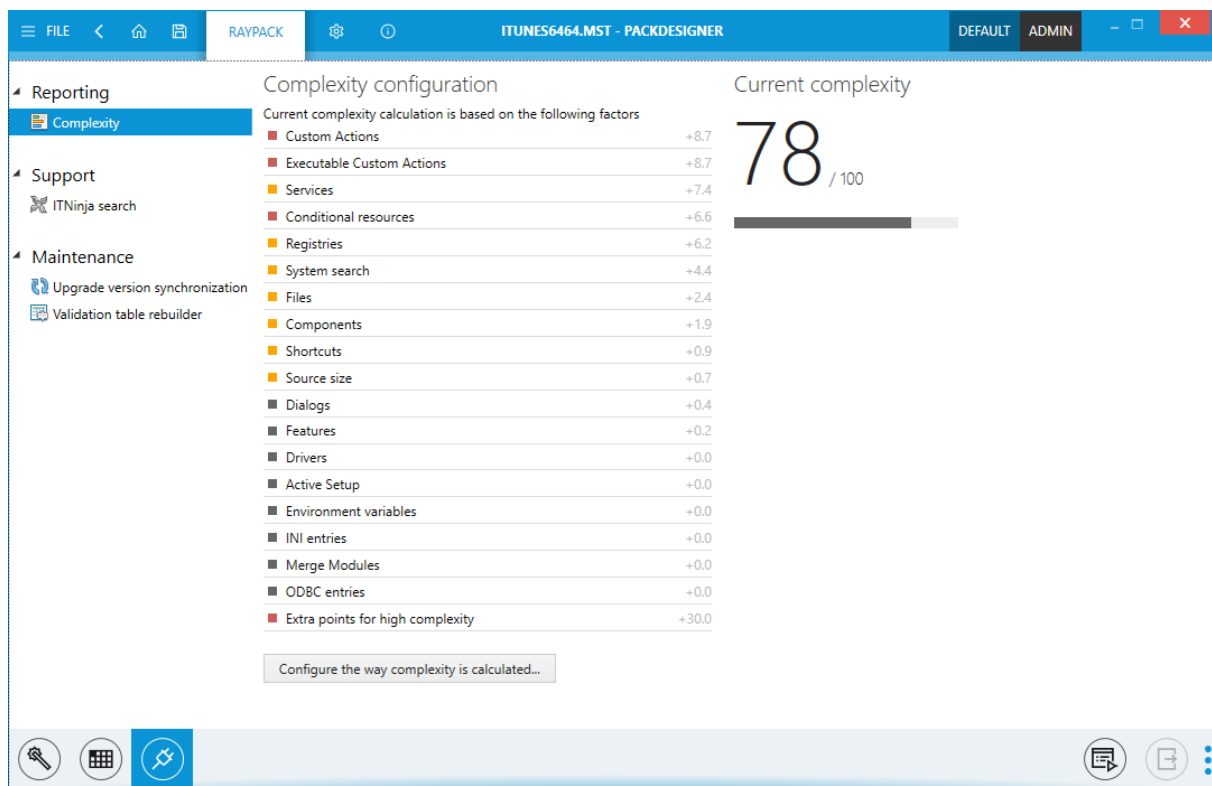
Scripts

RayPack 6.5 provides an easy way to start scripts (written in VBS, PowerShell) or any arbitrary executable on the current project. Scripts and tools are read from the following location.

```
<PackPoint>\Scripts
```

`<PackPoint>` is the location where PackPoint resources are installed. The default installation of RayPack already contains two sample scripts for GUID manipulations (both equal from a functional point of view; one written in VBS, the other one in PowerShell). In order to run a script, select it from the list and press the **Execute** button.
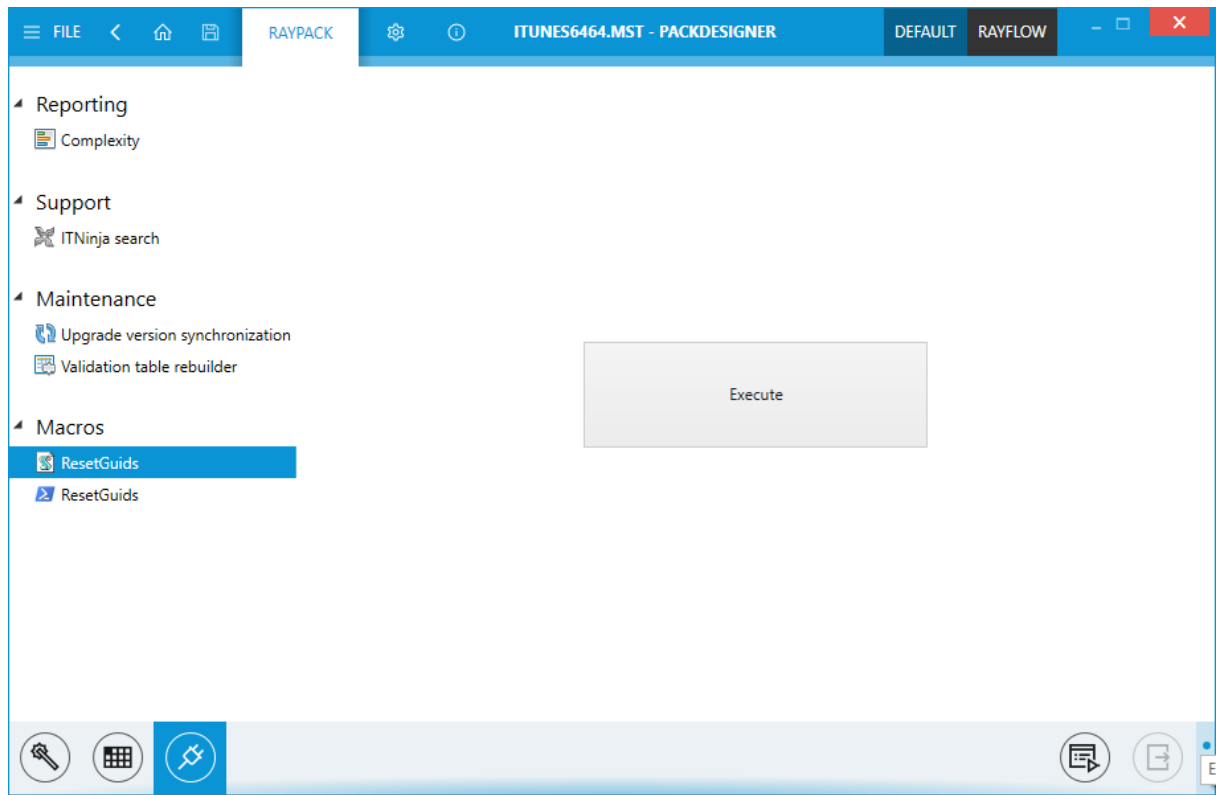
Below are the results of running the **ResetGuid** script.

To create own scripts...

The advanced topic *Creating PackDesigner scripts* discusses how to create own scripts that automate the work in PackDesigner.

Building Packages

As soon as a packaging project (RPP), an Installer database (MSI), or a transform file (MST) has been opened for edition in PackDesigner, it is always possible to build a target package from it.

Always means, that from a technical perspective, the Build option is permanently available. Users may simply use the hot key **F7** to call the Build dialog at any time during a PackDesigner working session. From a logical point of view, packages should not be build from invalid or incomplete resource bundles. Therefore, it is highly recommended to validate the project / the package contents first and only build target packages from them when the validation has been successful.

However, **from an RPP project** opened within PackDesigner packagers may build the following target formats:

- MSI
- App-V 4.6*
- App-V 5.X*
- MSIX
- MSIX app attach (VHD)
- ThinApp**
- SWV***
- Citrix AppLayering (LAYPKG)
- Intune (win32 package)

Packagers may build the following target formats **from an MSI package** opened within PackDesigner:

- RPP
- MSI
- App-V 4.6*
- App-V 5.X*
- MSIX
- MSIX app attach (VHD)
- ThinApp**
- SWV***
- Citrix AppLayering (LAYPKG)
- Intune (win32 package)

> **Be aware:**
> Building is a different process than simply saving. When a target package is build, all resources are checked for validity and existence, all target format restrictions are evaluated, and finally, the target package files (e. g. `*.msi` and `*.cab` files) are generated anew.
>
> Saving changes made to a file does not include all these steps but simply saves the updated state of the current project or package. Saving should never be the last step for a target package. The last step for a target package should always be build.

The following table shows the differences between building and saving of packages:

Target format Source format	MSI	MST	RPP	Virtual formats (App-V, ThinApp, SWV), MSIX packages, AppLayering files
MSI	**BUILD:** Recreates Media layout by reading and recompressing all files. Because of that, building is usually slower than saving. Optionally, it builds linked folders and setup wrappers. **SAVE:** Compiles only additional files, existing sources stay intact. Faster than building.	**BUILD:** Not available **SAVE:** Compiles only additional files, existing sources stay intact. RayPack changes to Transform mode after saving as an MST file.	**BUILD:** Extracts the content of an MSI package, creates a project file (XML-based), does not recompress resources. **SAVE:** Not available	**BUILD:** Exports the current package to a virtual format. **SAVE:** Not available
MST	**BUILD:** Not available **SAVE:** Compiles only additional files, existing	**BUILD:** Not available **SAVE:** Compiles only additional files, existing sources stay	**BUILD:** Not available **SAVE:** Compiles only additional files, existing sources stay	**BUILD:** Exports the current package to a virtual format. **SAVE:** Not available

Target format / Source format	MSI	MST	RPP	Virtual formats (App-V, ThinApp, SWV), MSIX packages, AppLayering files
	sources stay intact. Faster than building.	intact.	intact.	
RPP	**BUILD:** Builds Media layout by reading and compressing all files. Optionally, builds linked folders and setup wrappers. **SAVE:** Not available	**BUILD:** Not available **SAVE:** Not available	**BUILD:** Not available **SAVE:** Updates the project meta-data and streams. Does not compile anything.	**BUILD:** Exports the current package to a virtual format. **SAVE:** Not available

To Build a Target Package...

1. Click on the **FILE** button within the Main Toolbar at the top of the Visual Designer application screen, or simply hit **F7** on the keyboard.

2. Select **Build** from the options menu column on the left-hand side.

   Make sure the radio button **To disk** is selected.

3. Decide whether the newly created target format object should be displayed in the context of a system explorer instance once the build process has finished. If such an **inspection explorer** is wanted, users have to activate the checkbox **Open the folder after the building is finished**, displayed within the Settings section, at the lower dialog area.

4. Click on the **tile** that represents the desired **target format** (see the list of available target formats above).

5. Define the target package **name** and **location**.

6. Click **save**.

7. Wait for the process to finish.

   Depending on the complexity of the resources that have to be molded into the target package format, the build process may take a while.

   > **Note:**
   > The process information will be displayed within the progress dialog: The currently processed resource is displayed by type and path, indicating the area of activity and possible reasons for longer waiting periods (e. g. because a file that has to be deployed with the package is quite large).

_____

* Only when the virtualization pack has been licensed
** Only when the virtualization pack has been licensed, and the ThinApp SDK is installed on the packaging machine. Please refer to the Release Notes for details regarding supported ThinApp SDK versions
*** Only when the virtualization pack has been licensed, and the SWV agent is installed on the packaging machine. Please refer to the Release Notes for details regarding supported SWV agent versions

To Build a Target Package and Save It to RayFlow...

Chapter Saving Files in RayFlow describes how to build a package and upload it automatically to the current RayFlow instance.

To Make a Quick Build...

Quick build is an option to build the package to a predefined folder and MSI format. You can find more information about quick builds in section Making Quick Builds.

Rebuilding and Consolidating Windows Installer Databases

Any MSI opened in RayPack can be rebuild using custom build settings and Cabinet layout options. Typical scenarios include:

- Extracting compressed images
- Consolidating uncompressed images back to a compressed one
- Consolidating patched administrative images
- Splitting Cabinet files in already compressed files

To rebuild Windows Installer databases

1. Open any valid `*.msi` file
2. In the Build options section set up the build settings according to the requirements
3. Press **FILE** > **BUILD** (or simply hit **F7** on the keyboard) to show the build screen.
4. Select the target **MSI format**
5. Select the **location** to which the MSI will be build
6. The file will be rebuilt using the settings present in the Build options screen

> **Note:**
> The created MSI will be not opened. In order to perform further customization, click **FILE** > **OPEN** (or **CTRL+O**), select the MSI format and point to the location where the MSI has been saved.

Building Microsoft Patches

A Windows Installer patch (`*.msp` file) is a relatively small, self-contained package including the updated resources of an application, and additional metadata, describing which versions and products can receive the patch.

Upgrading applications by delivering an MSP file has certain advantages over the full upgrade procedure: An MSP can contain a minimal set of data necessary to patch, for example a binary difference between files that have been actually changed. This allows users to download an upgrade patch that is much smaller than the full installation package. User customizations of the patched application are usually preserved during the patch update.

RayPack provides an easy way to create a Windows Installer patch for any valid MSI file. These two basic scenarios are supported:

1. Manually preparing two images (old and new) and creating a patch containing the delta
2. Opening any MSI, adjusting required properties and features, and creating a patch against the original MSI contents.

Note:
Patching is only available in RayPack Professional or Enterprise edition.

To create a patch between two MSI images

1. Prepare two images – one for the old application (before the patch) and one for the new one (after the patch)
2. **Open the NEW image** in RayPack
3. (Optionally) Apply adjustments to necessary properties, resources etc.
4. Press **FILE** > **BUILD** (or simply hit **F7** on the keyboard) to open the Build dialog
5. Select the **target format MSP**
6. Select the **location** where the MSP file has to be saved
7. RayPack will ask for a location of the base image. **Select the OLD image** when prompted to do so.
8. Patching may take some time, depending on the number and size of included resources (such as files).
9. When the progress indicator disappears, the patch creation has been finished, and the MSP file is available at the selected target location.

Note:
RayPack does not require images to be extracted administratively before opening them in RayPack. Any valid MSI (regardless of compression and media layout) can be used in this process. In order to compare the sources, necessary files will be extracted to a temporary location. In such case, make sure that the amount of free space on your hard drive is at least thrice as big as the actual size of the source and target image.

To create a patch from a single package

1. **Open any MSI package** that will be a base image for patching
2. Apply any adjustments to necessary properties, resources etc. It is recommended to adjust the ProductVersion, PackageCode at this point.
3. Do not save the changes. Doing so would update the base image!
4. Press **FILE** > **BUILD** (or simply hit **F7** on the keyboard) button to open the Build dialog
5. Select the **target format MSP**
6. Select the **location** where the newly created MSP file will be saved
7. RayPack will ask for the location of the base image. **Select the same MSI** as opened in the first step.
8. Patching may take some time, depending on the number and size of included resources (such as files).
9. As soon as the progress indicator disappears, the patch creation has been finished, and the MSP file is available at the selected target location.

To customize patch options

When the patch is built the default settings will be used. This configuration can be adjusted in the Build options view.

In order to change the patch properties (for example the availability of removal functionality) adjust these settings before starting the actual building procedure.

Windows Installer limitations

When creating a patch, the following restrictions have to be taken into account:

- Do not change the structure of existing features. New features can be added, and when removing an obsolete feature all child features must be also removed.
- Do not change identifiers of any component (note: use *Upgrade* tab to synchronize components between updated packages)
- Do not change the name of the MSI package between versions
- Change the *ProductCode* of the upgraded package if any of the following is true:
  - Coexistent installations of old and new product must be possible
  - The name of the .msi file has been changed
  - Identifier of any existing component has been changed
  - A component has been removed from an existing feature
  - Structure of features has changed (for example, an existing feature is now a child of another existing feature)
  - An existing child feature has been removed from its parent

More comprehensive list of limits and requirements can be found here: https://msdn.microsoft.com/en-us/library/aa367850.aspx

Uninstallable Patches

The Build options view can be used to customize whether a patch should be uninstallable. Enabling this setting will set the informational attribute to the patch metadata, informing that the patch can be removed once it is installed. However, the actual availability of the removal options depends on number of factors due to Windows Installer limitations.

The patch will be uninstallable if all the following conditions are fulfilled:

1. The checkbox **Allow the patch to be removed** in the Build options view has been checked

2. The Windows Installer installing the patch package is in version 3.0 or newer

3. The machine policy DisablePatchUninstall is not set on target computer.

4. Patches have to have the MsiPatchMetadata table (RayPack creates this table automatically in the background)

5. The patch has been applied to an application installed in a context for which the user has insufficient privileges to uninstall patches (see http://msdn.microsoft.com/pl-pl/library/aa372102%28v=vs.85%29.aspx for more information).

6. The patch is not a major upgrade. Major Upgrades of an application should be performed by installing the upgraded application (`*.msi` file) rather than a patch.

7. The patch does not add new rows to the following tables

- AppId
- BindImage
- Class
- ComPlus
- CreateFolder
- DuplicateFile
- Environment
- Extension
- Font
- IniFile
- IsolatedComponent
- LockPermissions
- MsiLockPermissionsEx
- MIME
- MoveFile
- MsiServiceConfig
- MsiServiceConfigFailureActions
- ODBCAttribute
- ODBCDataSource
- ODBCDriver
- ODBCSourceAttribute
- ODBCTranslator
- ProgId
- PublishComponent
- RemoveIniFile
- SelfReg
- ServiceControl
- ServiceInstall
- TypeLib
- Verb

When the option **Allow the patch to be removed** is checked, RayPack will automatically check

whether the patch can be marked as removable by analyzing the content of the tables mentioned above. If new content is detected in any of these tables, RayPack will issue a warning, informing that the patch will not be removable even though the setting is enabled.

Be aware:
Patches applied to an administrative installation are not uninstallable.

Building MSIX Packages
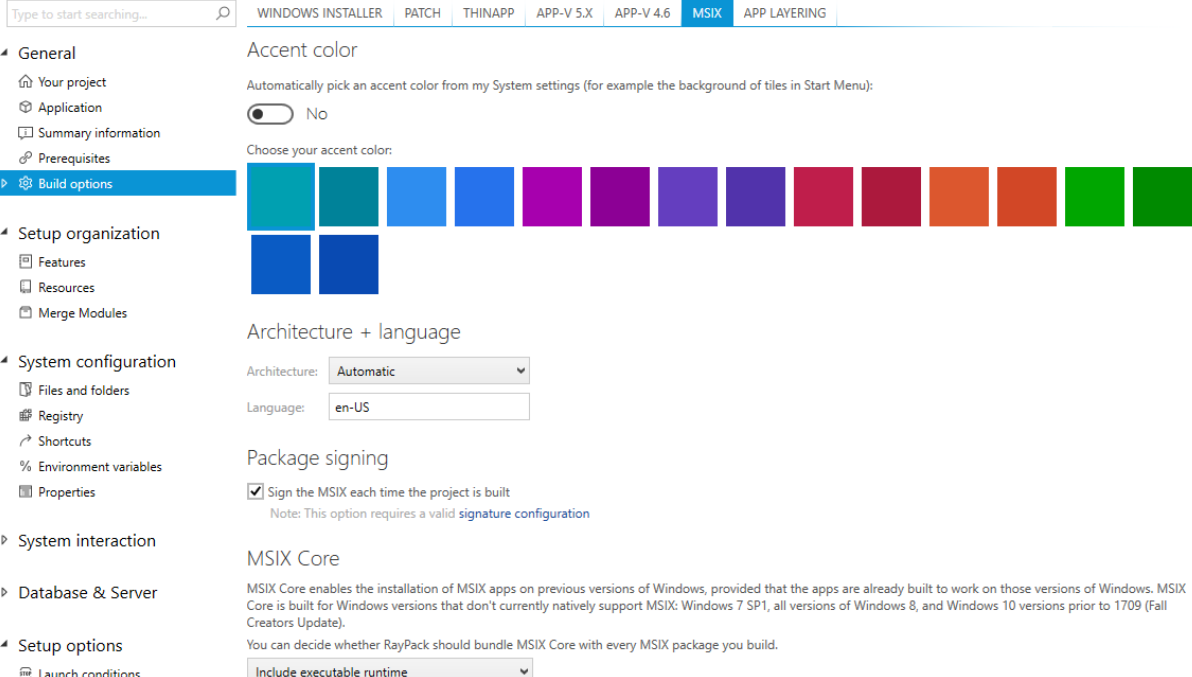
MSIX is the file format used to distribute and install apps on Windows 10 1809 and later and on certain mobile and gaming devices. Whilst the original concept required software developers to rewrite their apps using the new technology and security concepts, later Windows 10 builds offer a native support for classic apps which have been converted to MSIX format and are installed / handled by a system using a technology called Desktop Bridge. RayPack can convert existing MSI / RPP / MST projects to MSIX format without actually expanding of setups on a target machine.

In Order to Convert an RPP / MSI / MST project to a UWP (Universal Windows Platform) Bridge APPX Package...

1. Make sure that the current profile contains a definition of the signature and the certificate.
2. Open a project to be converted.
3. Go to **General** > **Build Options**. This is the place where conversion options can be configured. The default settings are based on the current profile configuration.

4. Press **FILE** > **Build** and select MSIX item.
5. The package will be automatically converted and signed.

Configurable Settings

For a list of configuration settings, refer to the *Conversion* chapter (section MSIX + UWP).

Building MSIX app attach (VHD) Files

With MSIX app attach, the application is completely detached from the OS it runs on, and can be dynamically attached and detached without any noticeable delay. This means a clean deployment, no need to prepare golden images and all benefits of MSIX technology. The technology is currently in preview and can be tested starting from Windows 10 May 2020 Update (2004).

The configuration for MSIX app attach builds is the same as for any MSIX build (see more information about configuring for MSIX build in section Building MSIX Packages). The only difference is the format, which is going to be a VHD with expanded content and a set of permissions required for a proper attaching. Additionally, RayPack writes 4 scripts which can be used to test the deployment: two for staging and registering, and another two for de-staging and de-registering.

More information about MSIX app attach:
https://docs.microsoft.com/en-us/azure/virtual-desktop/app-attach

Building Intune Packages

Any RPP/MSI/MST project can be converted to an `.intunewin` package. This is a fully automated conversion process which does not require any additional input. To convert the currently opened project to Intune format, select **Intune win32 package** from the **FILE** > **Build** menu.

> **Note:**
> Some Intune functions may require extra dependencies to be present on the local machine. You may be prompted to download them, which RayPack does automatically. Without these dependencies however the process will be aborted.

Building Citrix AppLayering Layers (LAYPKG)

LAYPKG is a file extension used for layers, exported from Citrix AppLayering. RayPack can convert existing MSI / RPP / MST projects to LAYPKG format without actually expanding of setups on a target machine, and with no runtimes or third-party tools required.

> **Note:**
> In order to be able to build LAYPKG file, the information about parent package (OS Layer) must be configured in Build options or in the default settings of AppLayering target format. For more information, see the Settings > Conversion chapter.

In Order to Convert an RPP / MSI / MST project to a LAYPKG (Citrix AppLayering) file...

1. Open a project to be converted.
2. Go to **General** > **Build Options**. This is the place where conversion options can be configured. The default settings are based on the current profile configuration.

OS Layer package configuration

You can select OS Layer for newly built package.

Compression

Compression method: None

3. At the minimum, the information about OS Layer is required. Experienced users may enter a GUID of the layer for which the package is meant. The easiest choice to get the identifier is to use the **...** button and pick the right LAYPKG (you may need to export it beforehand from your system). RayPack will then read the identifier and set it for you. Note: since in most cases the parent OS layer is the same for all packages, it is highly recommended to set it up once in the Settings screen, so that no more subsequent changes for each package are required. If you need to support more OS layers, consider setting up several RayPack profiles, each for a specific OS layer.
4. Press **FILE** > **Build** and select **PackLayering** item.
5. The package will be automatically converted and signed.

> **Be aware:**
> The automatic conversion may not be able to correctly convert packages that rely on **Custom Action** usage. In this scenario, it is recommended to repackage the product on a physical machine and then convert the repackaged output to the MSIX format.

Configurable Settings

For a list of configuration settings, refer to the *Conversion* chapter (section APPLAYERING).

Making Quick Builds

By making the Quick build, the current package is rebuilt using exactly the same method as the standard File > Build procedure:

- Compression and build settings from build options are respected.

- Dynamic streams are being refreshed.
- Files and folders are being compressed.

However, the following settings are set automatically and users are unable to override them:

- The target format is always MSI (to build to another format, use standard File > Build procedure).
- The location of the built package is always in a subfolder `_MSI` in the current package directory

Making a quick build saves a few keystrokes and produces a package which can be quickly tested and troubleshooted.

> **Be aware:**
> This option is not available for MSI or MST files. Only RPP files are supported.

In order to make a quick build...

1. Make sure that an RPP project is opened in PackDesigner

2. Press button **Quick build** in the bottom part of the screen

Quick build
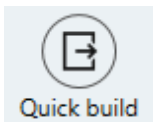
3. Alternatively, use **CTRL**+**F7** keystroke to achieve the same

Testing Packages

Certain aspects of the package (like Custom Actions running in the UI or the Immediate sequence) can be troubleshooted and tested only once an actual MSI installation is started. The problem is that performing a full Installer session and, thus changing the current state of the machine by the Windows Installer engine, is not desired. Especially when not on a Virtual Machine.

The solution to this problem is to trigger testing from PackDesigner. RayPack prepares a build which is similar to an actual package, with the following differences:

1. In order to speed the process up, building of CAB files and medias is skipped

2. The installation sequence is adjusted to prevent any deferred system changes to be written. Effectively, this means that while the installation runs as normal (and even displays the progress bar of the installation) no changes are actually made to the system, and so the package can be tested many times in a row.

Be aware:
Standard Windows Installer functionality and well-designed Custom Actions are only making system changes by writing to a deferred execution in-script. Sometimes however, poorly developed packages may contain certain actions that do not respect this rule, for example by changing the local system directly from the UI Sequence. These changes may affect the local system.

In order to test the package

1. Open any supported package file (RPP, MSI, or MST) in PackDesigner

2. Press button **Test** in the bottom part of the screen
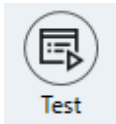
3. Alternatively, use **F8** keystroke to achieve the same

4. A quick build of the package without CAB/Medias will be created

5. After building is finished, the installation routine will be started automatically

6. After closing the Windows Installer session, either by canceling or finishing, a Windows Explorer window will be shown and the log file from the installation will be automatically highlighted.

Tip:
By default, a log is created in `_Logs` subfolder and verbose logging options (`/l*v`) are used. These settings can be configured by manually adjusting the XML file of the selected RayPack profile. Additionally, using the same technique it is possible to disable log generation completely.

Converting MSI Files into RPP Projects

Import MSI packages into RPP project files

To create an RPP from an MSI:

1. Go to the **Home Screen** and use the **open** tile.

As an alternative, is is also possible to

a. Click on the FILE tab available from the Main Toolbar.
b. The **FILE menu** is displayed.
c. Click on the **Open** option from the view menu column at the left-hand side.
d. Select the first option "**Windows Installer project**"

2. Either way, a Windows dialog is displayed, allowing users to **navigate to the MSI** file which has to be opened.

3. Select the MSI and click **Open**.

> **Tip:**
> The same import procedure also works for MSI + MST combinations. To import a combination of those file types into an RPP file, simply navigate to the MST and click Open. RayPack will automatically ask for the base MSI that belongs to the selected MST. Select the MSI and click Open. The following steps of the procedure are identical for both scenarios.

4. RayPack automatically loads the MSI file into the PackDesigner interface.

5. Once the MSI is opened, click on the **FILE** tab available from the Main Toolbar.

6. Click on the **Build** option from the view menu at the left-hand side.

7. Use the tile "**Raynet Package Project *.rpp**"

8. In the displayed Windows file system browser dialog, **navigate to the desired target directory** and adjust the default file name if required.

9. Click **Save**.

10. A progress indicator dialog is displayed whilst RayPack transfers the MSI based package information into the new RPP file.

    When the dialog disappears, the build process has been completed.

11. Please **close the MSI** file. Make sure to discard any changes performed in the interim.

12. Once this is completed, open the RPP file.

    To do so, follow the first three steps of this procedure, and simply navigate to the RPP file instead of the MSI file.

13. As soon as the RPP is opened in PackDesigner, it is ready for manipulation.

    Editing the RayPack package project file does not take any effect on the original MSI file.

Silent conversion of MSI projects

To silently convert an MSI database into RPP project, use the command-line `rpcmd.exe` tool.

```
RpCmd.exe build -format "RPP" -input "<path_to_msi_file>" -output "<path_to_rpp_file>'
```

You can find more information about silent command line switches in this section.

Working with Transforms

This chapter describes various topics regarding creation and management of Windows Installer transform files (`.mst`).

Creating an MST Transform

To create an MST transform

1. Open the base MSI package
2. In either of Visual Designer or Advanced Mode perform the necessary customization. Note that RayPack highlights the edited, added and removed cells (see Highlighting & color codes for more information)
3. Click on **FILE** button and select **Save as** option from the left menu
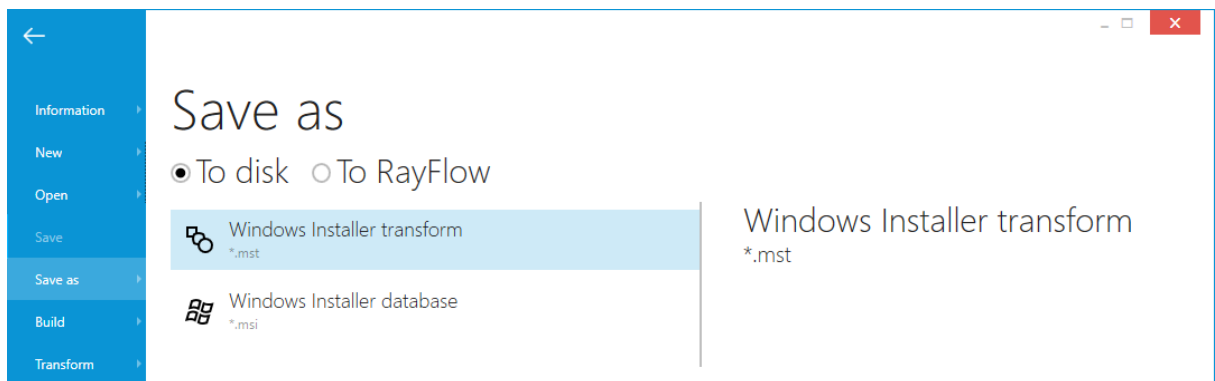
4. Click on the **Windows Installer transform** button and select where to save the transform. By default, RayPack will open the folder where the current MSI package is located and offer its name as a default file name for a new MST file. Select the desired transform file name and location and press **Save** to save the changes as MST transform.
5. From now on, RayPack works in the Transform mode, which means that pressing **CTRL** + **S** or clicking on the floppy icon in the left top corner saves changes to the transform, and not to the underlying MSI file.

> **Tip:**
> To create a response transform (an MST file containing values that would be otherwise entered during the UI installation) use the PackTailor wizard. More information can be found here.

Chained transforms

Transform files created by RayPack are chained. This means the changes are not cumulative if more than one transform is already applied to the package. Consider the following scenario:

- `PackageA.msi` is opened with `TransformA.mst` and `TransformB.mst` applied to it.
- User adds an MSI property and then saves the transform as `TransformC.mst`.
- At this point, RayPack shows changes from all transforms and highlights them using a green background (see Highlighting & color codes), and the information tab Transforms in the Backstage menu indicates that two transforms are currently applied.
- As a result, a new transform `TransformC.mst` is created, containing only the single change (a new MSI property). The changes present in first two transforms are not a part of the result.
- In order to install the package and apply all changes, the transforms have to be applied in the very same order, using command line argument
  `TRANSFORMS=TransformA.mst;TransformB.mst;TransformC.mst`.

> **Be aware:**
> In RayPack 2.1 and older, if any transforms were applied before pressing *File > Save as* button and selecting *Windows Installer transform option*, the resulting transform was always containing all changes merged. This has changed in RayPack 6.5, as the saving creates now a chained transform. In order to consolidate transforms, see Managing current transforms.

Saving to RayFlow

The radio button To disk/To RayFlow can be used to control the destination of an MST file. More information about how to save a transform in RayFlow can be found in chapter Saving files in RayFlow.

Universal transforms

Universal transforms are MST files which can be applied to all MSI databases. Advanced topic Creating universal transforms describes how to configure RayPack to create the transforms that are generic.

Managing Current Transforms

Managing of transforms is available from a dedicated screen in the Backstage menu.

To access the Transform menu

1. Make sure there is at least one transform applied. Either open an MST file, or create a new transform using the functionality described in this chapter
2. Click on *FILE* button and select *Transform* option from the left menu
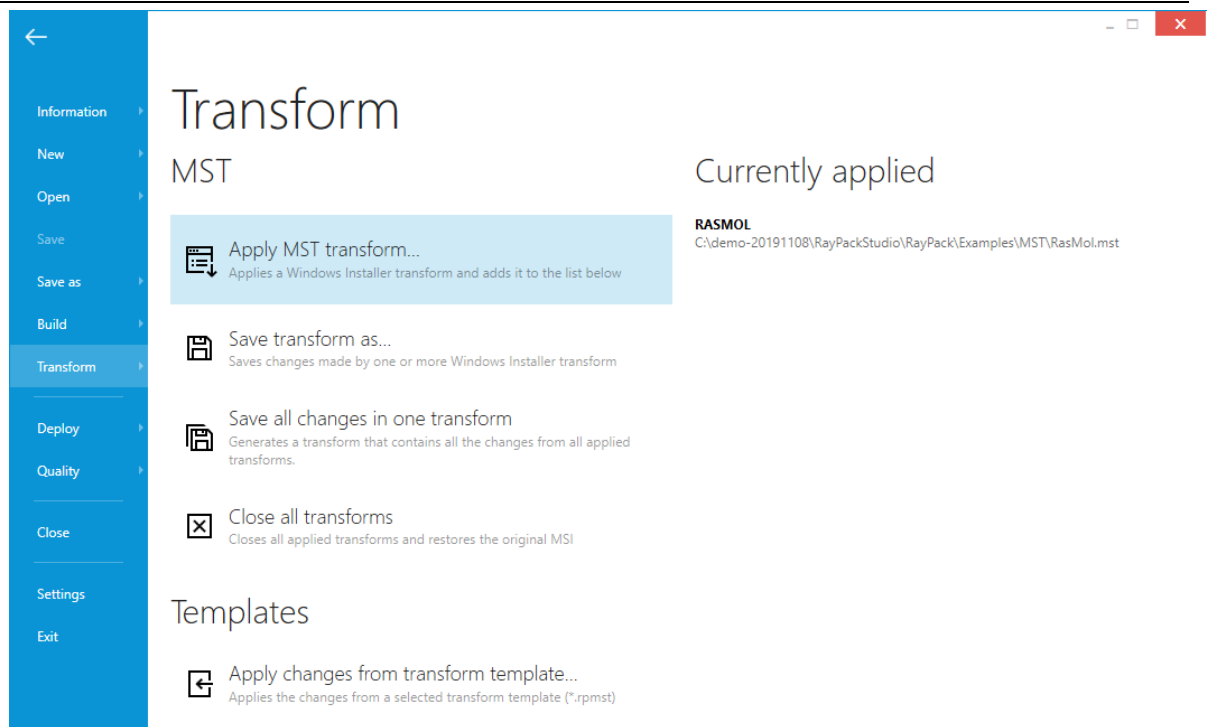
To view the list of currently applied transforms

The content of *Currently applied* section contains an ordered list of transforms (in order they were applied).

To apply another MST transform

In *Transforms* menu, click on *Apply MST transform...* button. A dialog will be shown prompting to enter the full path of the applied MST file.

To close all MST transforms

In *Transforms* menu, click on *Close all transforms* to close applied transforms. Note: If there are any unsaved changes to the current project, RayPack will ask whether you want to save them before closing the transforms.

To consolidate transforms

In *Transforms* menu, click on *Save transformed as...* to show a file save dialog prompting for a path where to save the new MST file. The file will contain all changes from all currently applied transforms.

> **Be aware:**
> In RayPack 2.1 and older, the functionality of this button was functionally equal to clicking *File> Save as* and selecting *Windows Installer transform option.* The result of this operation was always a consolidated transform. This has been changed since RayPack 4.0.

Transform Templates

The right hand side of the **Transform** menu contains two buttons used to load and save transform templates:
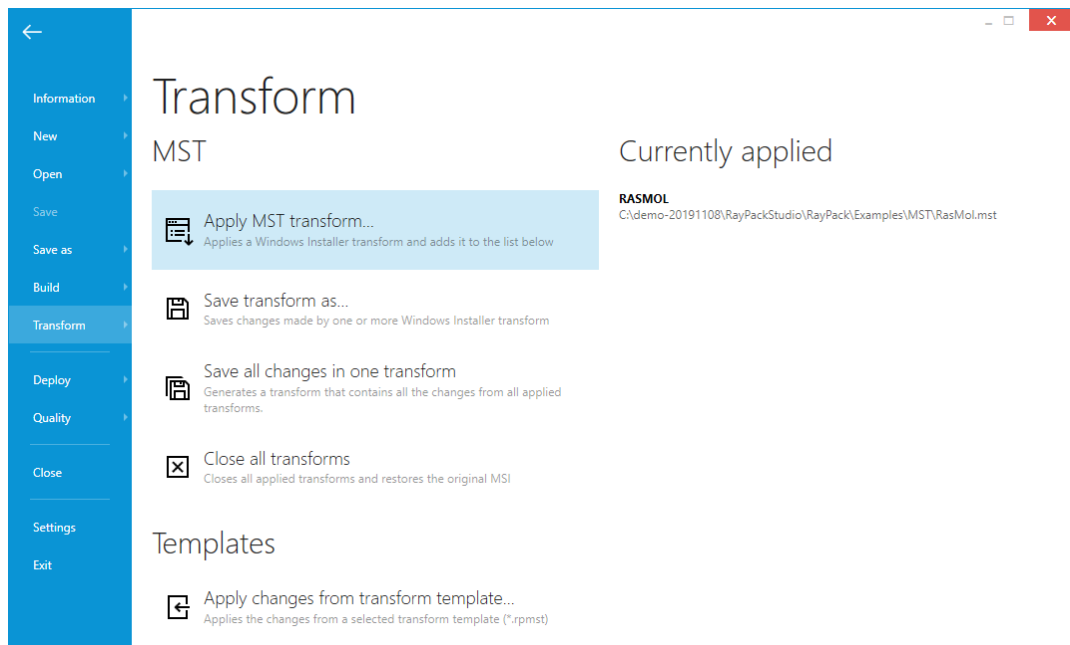
Transform template is a small XML file containing a definition of what rows and values have to be added to a standard MSI transforms. More about the format and its purpose can be read in the following section: Adjusting the MST template.

PackDesigner automates saving of transform templates and allows to apply them to the current project. The transform template can be apply to an MST project but also to an MSI/RPP project, making it a great tool for branding and simple snippets of MSI values.

To apply changes from a transform template

1. Open any MSI/MST/RPP project in PackDesigner
2. Click on **FILE** button and select **Transform** option from the left menu
3. Click on **Apply changes from transform template...** button.
4. You will be prompted for a file path of the transform template (`.rpmst`) to be used. The initial folder and file name is based on the current settings from the Projects section in the profile configuration.
5. Select a template and press **OK** to apply the changes to the current project. To verify that all

required rows are already in place (they will be highlighted) go to the **Tables** view.

> **Tip:**
> The default transform template that is coming out-of-the-box is empty. Uncomment the sample XML content to test and use it as a base for own customizations.

To create custom transform template

1. Open any MSI/MST/RPP project in PackDesigner
2. Adjust the project to contain the required changes.
3. Go to the Advanced mode > Tables view. The changes made since last time are marked green. Review them carefully, because everything marked green will be a part of the new transform template.
4. Click on **FILE** button and select **Transform** option from the left menu.
5. Click on **Save changes as transform template...** button.
6. Select a file path to the transform template (`.rpmst`) to be used. The initial folder and file name is based on the current settings from the Projects section in the profile configuration.
7. Select a template and press **OK** to save the set of currently highlighted changes to the specified location.

> **Tip:**
> It is not possible to save a transform template if there are no changes highlighted. Should this happen, RayPack will show a warning informing that no changes have been found in the package.

Fine-tuning and customizing transform templates

In chapter Adjusting the MST template you can find more information about editing of transform templates.

Automating packaging tasks using transform templates

Command line tools enable applying RPMST templates to existing MSI, RPP, and MST files. Using the silent switches it is possible to combine smaller bits and snippets represented by RPMST files, for example to automatically apply the branding, certain packaging properties, add required features or even change the Summary Information Stream.

Chapter Applying RPMST Templates contains listing of parameters and samples.

Comparing Projects and MSI Files
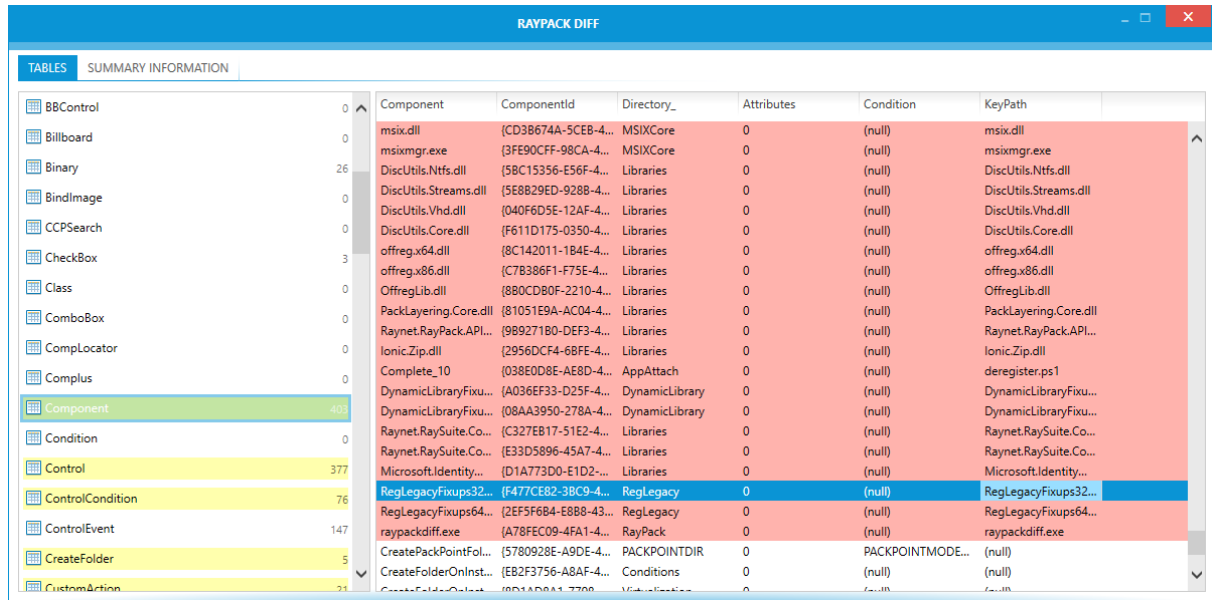
Once a project is opened (MSI or RPP) it is possible to compare it against any other MSI file, for example to understand differences between two different package versions.

To compare two MSI or RPP projects…

1. Open the new version an MSI/RPP file.
2. From FILE menu, select Transform and then press **Compare against…**
3. Select the previous version for the comparison.
4. The comparison view will be shown in a new window.

The comparison tool is divided to three sections:
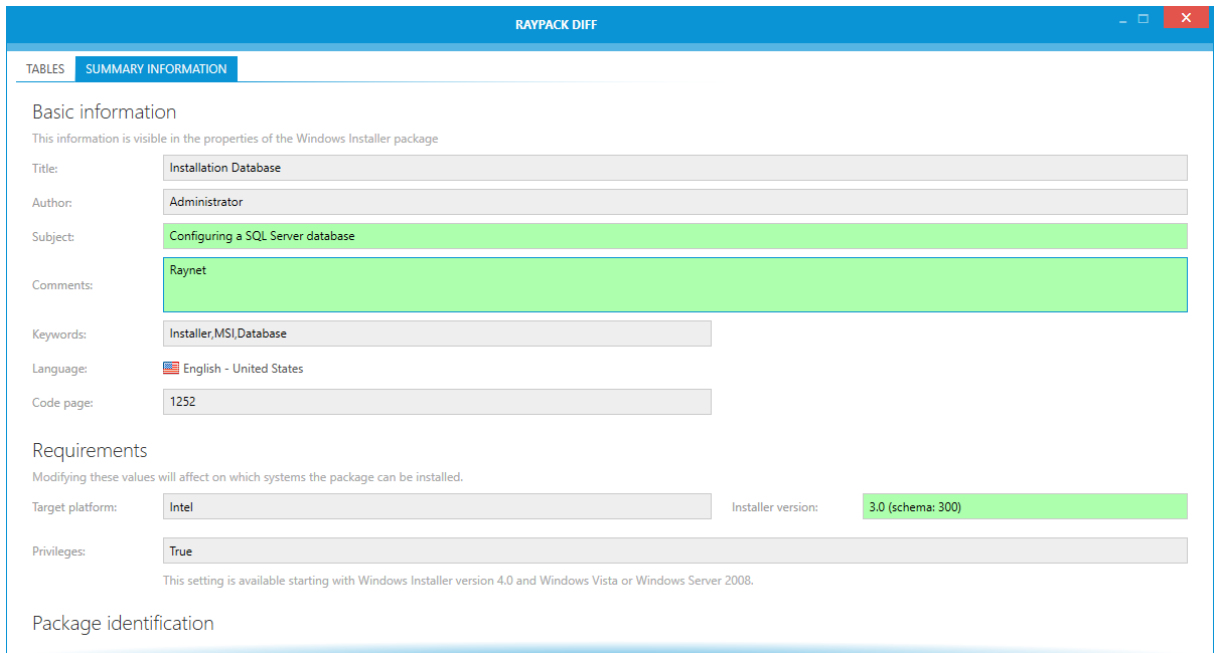
- The list of tables
- The list of rows in the currently selected table
- The information about Summary Information Stream

Compared tables

The comparison view of the Summary Information Stream

Legend

The comparison uses the following key colors:

- **Green**
  The entry is present in the new project but not in the old one (for example a new table, a new row, a new non-empty value)

- **Yellow**
  The entry has changed between versions (for example changed tables, changes values of cells/rows)

- **Red**
  The entry is not present in the new project but was in the old one (for example a removed table, removed row, removed value).

- **White**
  The entry is present and unchanged in both old and new project.

Importing Other Formats

This section outlines supported third party formats and how to import them to RayPack.

Importing 3rd Party Projects to RPP Projects

Prerequisites

To import a 3rd-party format (`.ism` and `.wsi`), the following prerequisites must be present on the machine where RayPack is installed:

To convert `.ism` file to `.rpp` format:

- InstallShield 2012 Spring or later

To convert `.wsi` file to `.rpp` format:

- Wise Package Studio 7

Note:
The availability of this option depends on the license information that you have entered for the current RayPack instance.

Be aware:
After the conversion is done, non-standard 3rd-party functionality may be not visible and editable in Visual Designer mode. It is still possible to edit them by using the Tables view.

Importing a project

To import a 3rd-party format:

1. Click on **FILE** button, select the **New** tab, and press the **Convert external project format into RayPack .rpp project** button.
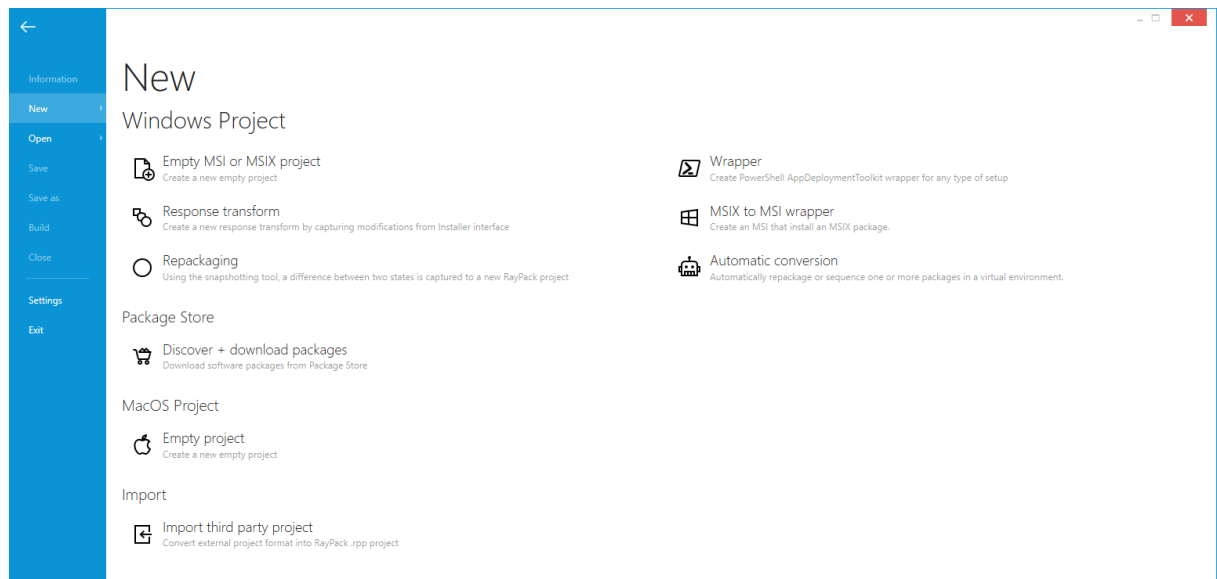
2. In the file browser dialog, select an `.ism` or `.wsi` project to be imported.
3. RayPack will convert the selected file into `.rpp` format.
4. After the conversion, another file dialog will be shown, prompting for a location where to save the new project. Select the required file path, and press **Save**.
5. The converted `.rpp` project will be immediately opened in *PackDesigner*.

Bulk conversion

A bulk conversion is available only in command-line mode. Refer to section Converting ISM/WSI files to RPP format for information about command line parameters and examples.

Importing Universal / Modern Apps into MSI Projects

> **Note:**
> The availability of this option depends on the license information that you have entered for the current RayPack instance.

To import a 3rd-party format:

1. Click on the **FILE** button, select the **New** tab, and press the **MSIX to MSI wrapper** button.
2. In the file browser dialog, select an `.msix` package to be imported.
3. Select the target location of the output `.msi` file.
4. RayPack will read basic package properties and will allow you to change them:
   - ProductName
   - ProductVersion
   - Manufacturer
   - Add / Remove Programs information availability
5. RayPack will convert the selected file into the `.msi` format.
6. You can open the resulting `.msi` file using PackDesigner in order to perform additional customization.

Package Complexity Indexing

The complexity index value is calculated based upon an extensive package property analysis. Working on packages with high complexity usually takes more time and requires more precise fine-tuning than working on those with a lower index value. As soon as a packaging project is defined, the complexity reflects changes in real-time, allowing packagers not only to make estimations on the specific workload, but actually to monitor how their treatments take effect.
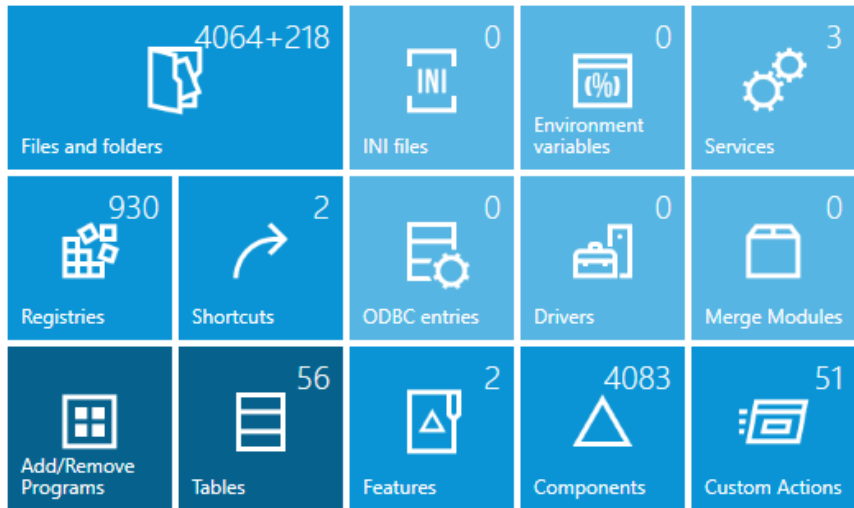
The current **Package Complexity Index** value is visible within PackDesigner's **Your Project** view. The screenshot above shows a package with a PCI of 78.

Understanding the index

The index is based on several internal factors. In order to see which factors affected the current calculation, go to the **Plugins** tab and highlight the **Complexity** item in the left panel.
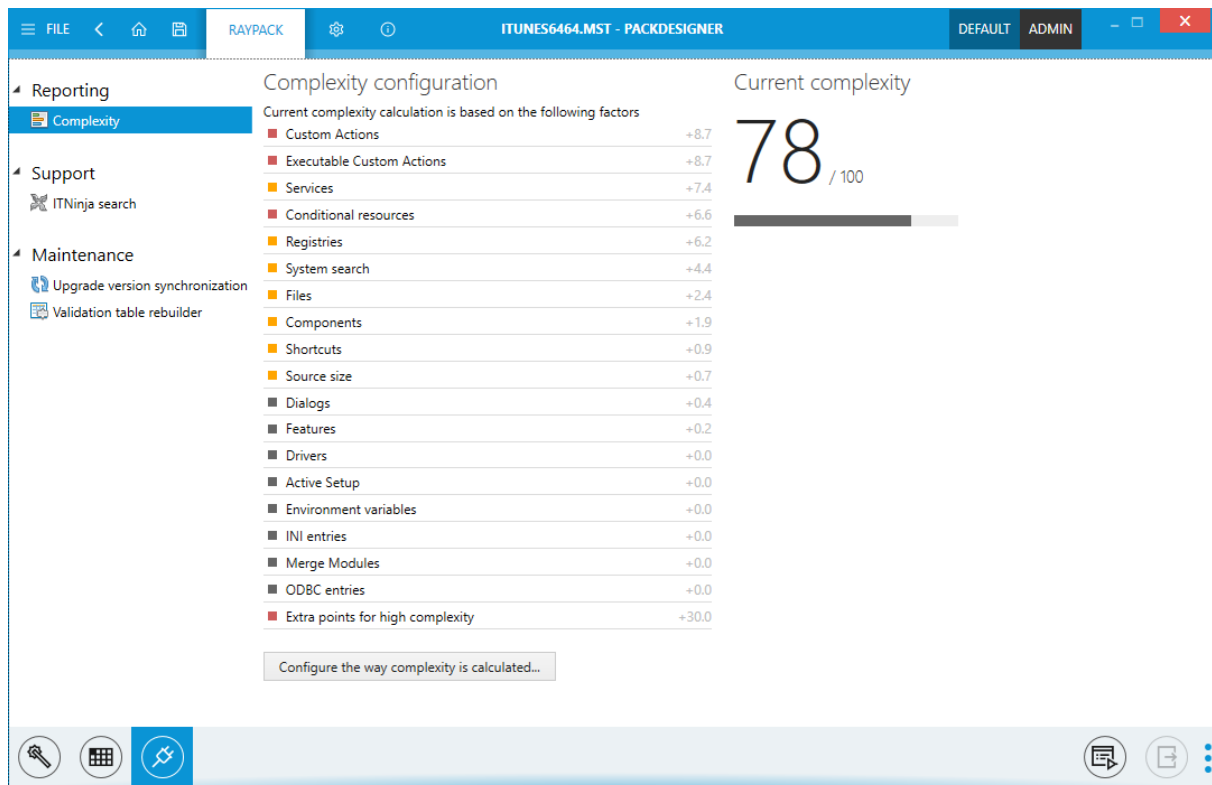
The list shows sub-indexes and their relevance for all factors that were included in the calculation. For example, the screenshot above shows that **Custom Actions** contributed 8.7 to the overall value of 78. If any item has a value of 0.0 it may mean that either its priority is low and / or that the package is not affected by it (for example - the picture above shows 0.0 for ODBC entries, because the package does not contain them). Additionally, the color of bullet next to each item denotes its status:

- **Gray color** - Low complexity
- **Red color** - High complexity
- **Orange color** - Medium complexity

The last item in the list are extra points added by RayPack for at least a single hit of **High** or **Medium** complexity for any of the sub-items. This way, a package that has relatively low values in all categories and one especially complex gets extra points to stress its importance and overall complexity. In the above example, the overall complexity (as a sum of all sub-indexes) is 48. However, at least one item has been reported as of **High** complexity (custom actions, executable custom actions and conditional resources) and so the package got an extra 30 points, with a final result of 78.

Customizing the complexity index

In order to configure how the index is calculated, modify the settings of the **Complexity** plugin. More information can be found in the advanced topic *Configuring the Complexity Index.*

Deployment

The **Deployment** tab provides connectors to deployment systems. RayPack 6.5 supports the following deployment systems.

- RayManageSoft
- System Center Configuration Manager
- Intune

Deployment to RayManageSoft is enabled only on machines having *RayManageSoft Administration Console* installed. There are no prerequisites for the SCCM connector.

To deploy the current project...

1. Open existing project in PackDesigner (RPP, MSI, and MST formats supported).
2. Press **FILE** > **DEPLOY**.
3. From the right panel select target deployment system.
4. Follow the wizard steps to specify deployment location and options (the options may vary depending on your selection).
5. Press **PROCESS** to build the package and deploy it to the selected deployment system.

Deploying to Intune

Deploying to Intune requires a one-time configuration of:
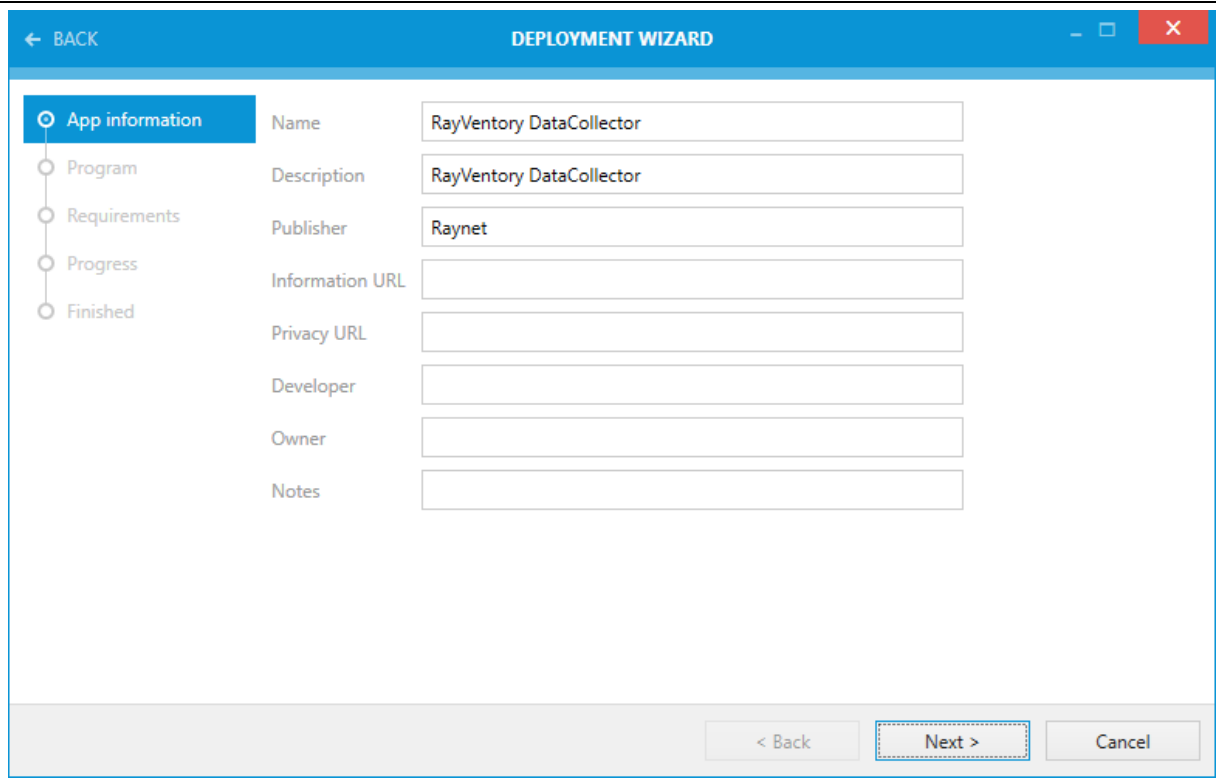
- Tenant domain
- App registration client ID and secret.

You should enter these values in the Settings screen before starting the Intune deployment wizard.

In order to build and deploy to Intune, open any supported project (RPP/MSI/MST) and select **File** > **Deploy > Deploy to Intune** from the backstage menu.

App information

This page contains basic information about the app being deployed. Some of these may already be populated by RayPack.

Program

This page contains various settings used to perform clean install and uninstall routine. You should be careful when changing these, as they may make the deployment package inconsistent. For most of cases, the defaults provided by RayPack should be just fine.
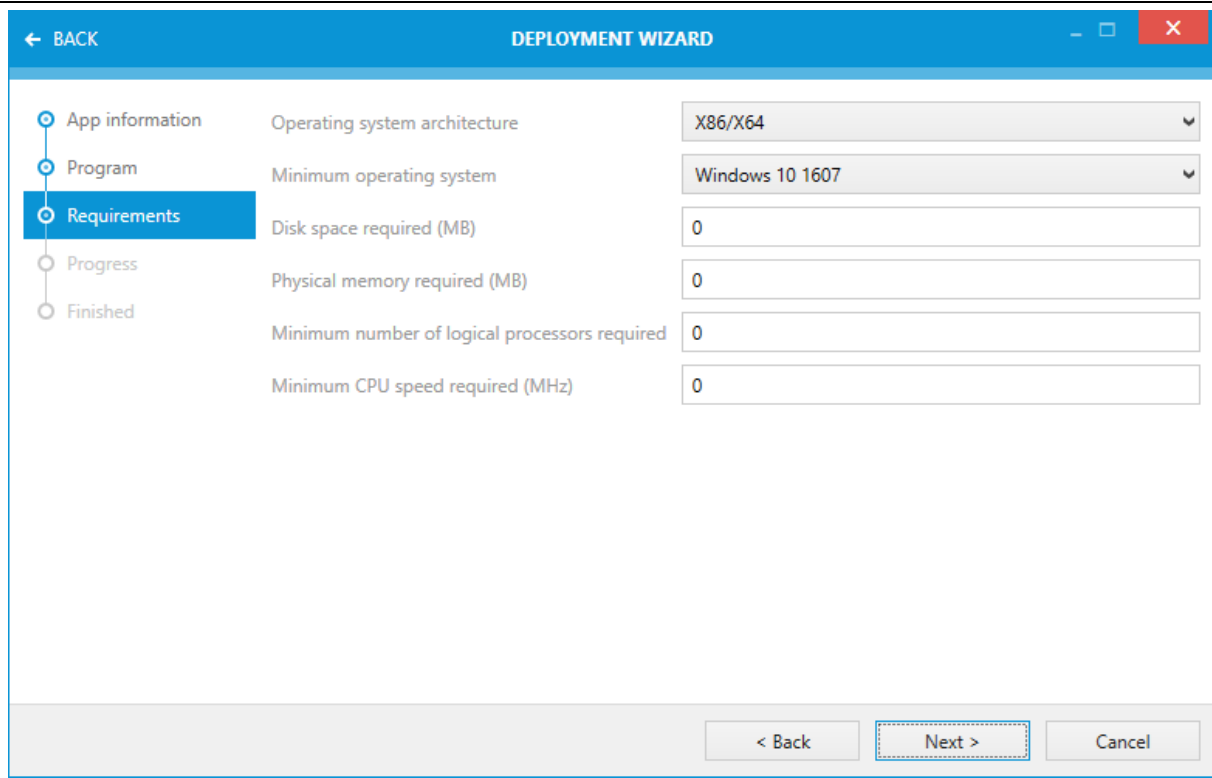
Requirements

This is the page where software and hardware requirements can be defined. For numeric values, putting the value 0 means effectively that the condition is always met.

Progress and Finished Page

Once ready, press **Next >** to start the deployment. The deployment process may take from a few seconds to a few minutes, depending on the size of the package, the network, connectivity etc. Once the process has finished, press **Finish** to close the wizard.

Pre-Quality Control

RayPack supports pre-quality control tasks and make it easier to discover system and packages collisions during the packaging process. RayPack seamlessly integrates with RayQC and RayQC Advanced which provide the underlying technology.

Prerequisites

In order to use the pre-quality control module, the following must be installed and licensed on the same machine RayPack is running.:

For automatic generation of checklists:
RayQC 2.1 or later

For collision and virtualization testing:

Checklist Creation

RayQC extends RayPack by adding a new functionality to the File > Quality menu.

In order to create a RayQC checklist from the current package:

1. Open any MSI/MST/RPP project
2. Click on the FILE menu and select the **Quality** tab.
3. The screen is divided into two sections - the left one contains RayQC-related functionality.
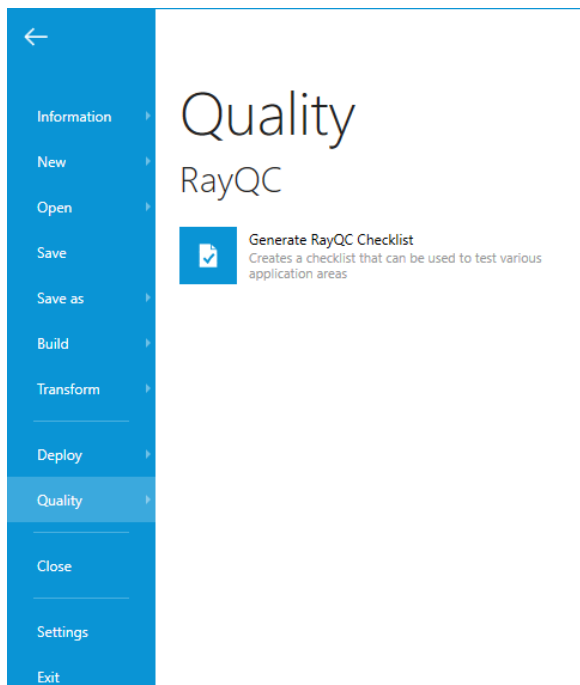
4. Click on **Generate RayQC Checklist**
5. Select the location where the checklist will be saved. The `.rqct` format will be used (a template for RayQC checklists)
6. The process may take a while. When the progress window is closed, the created checklist will be saved and available for further use.

Collision and Virtualization Testing

RayQC Advanced extends RayPack by adding a new functionality to the File > Quality menu.

A range of rule sets available in the RayQC Advanced library is available directly from RayPack, and the result of testing are shown immediately in the tables view, section Test results.

In order to create a RayQC checklist from the current package:

1. Open any MSI/MST/RPP project
2. Click on the FILE menu and select the **Quality** tab.
3. The screen is divided into two sections - the right one contains RayQC Advanced-related functionality.
4. When the screen is accessed for the first time, RayPack will try to connect to the RayQC Advanced library to get a list of available rules. This make take a few seconds, depending on the location of SQL database, network speed etc.
5. Finally, a view similar to the following should be shown:

RayQC Advanced

Select a quality control test scenario to execute on the current project. The library can be configured in RayQCAdvanced module.

Collision

[Test]

Virtualization

[Test]

System Readiness

[Windows 10 64-bit ▾]   [Test]
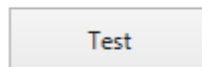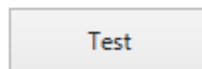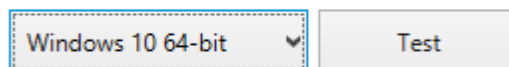
Refresh the list of available test scenarios

6. The view represents a flat representation of rulesets as they were defined in the RayQC Advanced library. The selected ruleset will be executed when its corresponding **TEST** button is pressed. For certain rulesets (for example **System Readiness**), an additional combobox is shown which allows fine-tuning of the selection (for example, changing the target operating system).
   o For some rulesets, additional windows may be shown.
7. The testing process may take a few seconds, depending on size of the package and other factors.
8. When the testing is over, PackDesigner will automatically jump to the Advanced View to the Tables section.
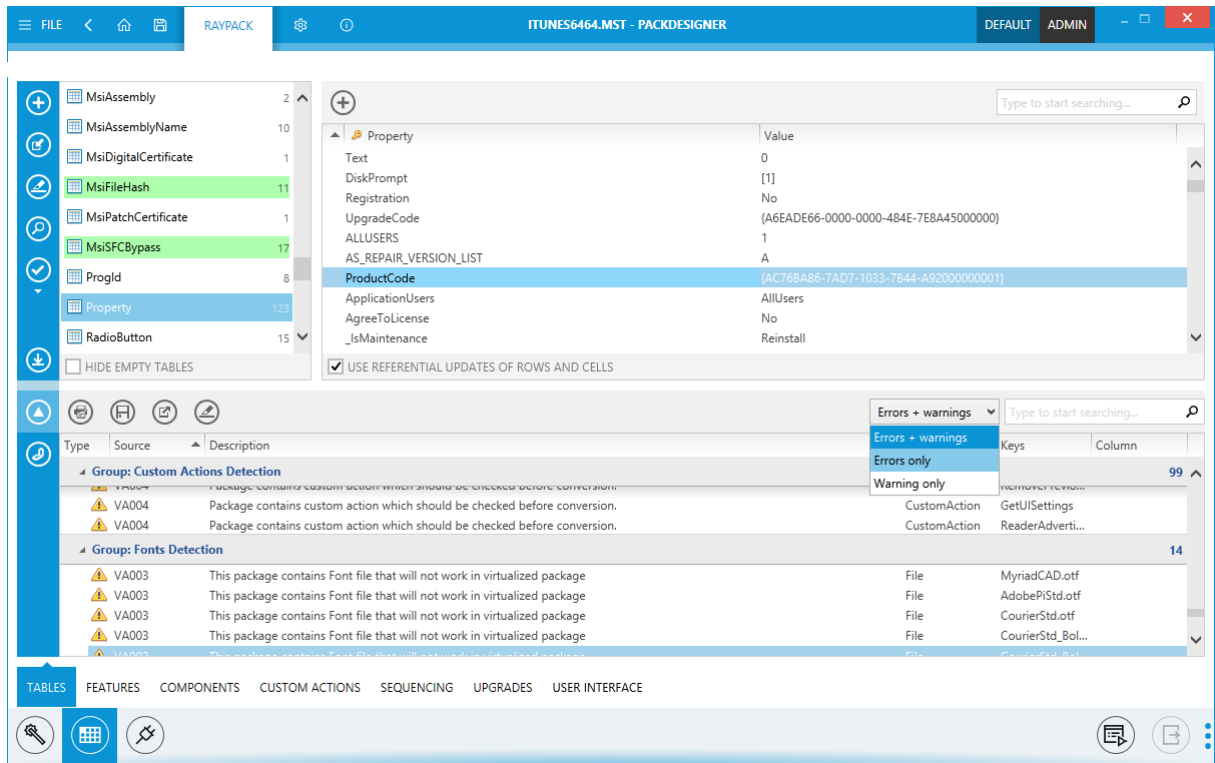
9. The list offers a standard functionality (sorting, grouping, printing and filtering). For more information, refer to the chapter Working with Validation Results

Common Dialogs

Some dialogs of the Visual Designer interface cover generic functionality, such as managing a folder structure, or using an inline editor interface for property manipulation. They are often reused as integral parts of several views.

The following common dialogs for standard procedures are covered within this section:

- Condition builder
- User object manager
- MSI formatted string field
- Object permissions
- Select a component
- Select a folder

Condition Builder

RayPack's condition builder supports packagers whenever conditional statements have to be constructed.
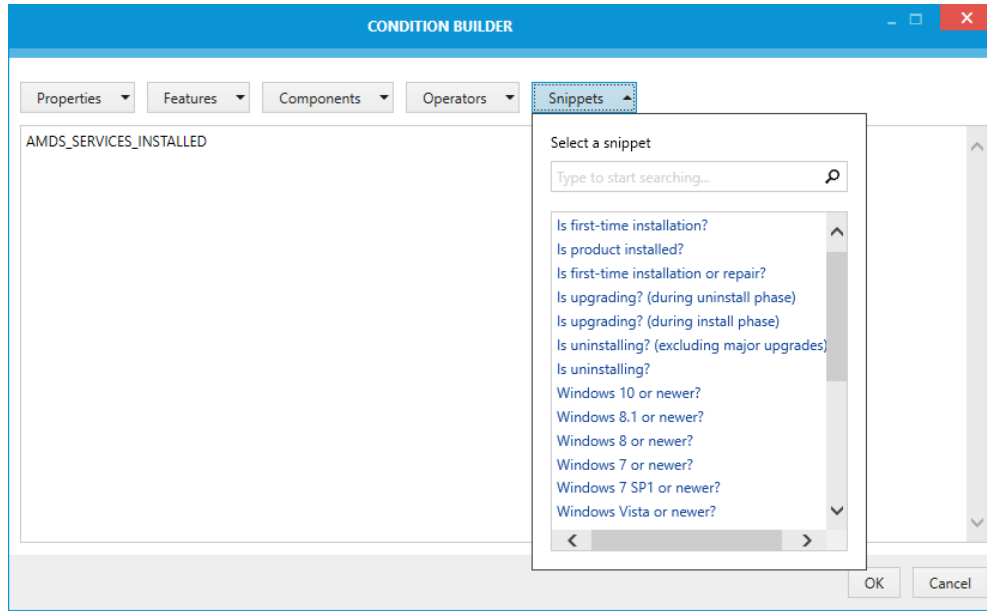
The dialog interface is divided into a button bar at the top, and a text input area for the definition of the actual condition string.

As usual in RayPack dialogs, the buttons **OK** and **CANCEL** at the bottom of the dialog window may be used to save the contents, or abort the manipulation.

How to define the conditional statement

Once the condition builder is loaded, the buttons at its top allow the utilization of a set of predefined values, grouped by their purpose. Although it is possible to manually type the desired condition string, the offered predefined items may help to save time and be sure about notation requirements.

PROPERTIES

With a click on the properties button, packagers will get a list of recognized MSI properties for conditional statements, such as ComputerName, ProductCode or VersionNT. By entering a keyword into the input field above the list, the properties are filtered down to those, which contain the keyword as part of their name, or part of their description. Hover over the link of any property item to read its description. Properties that are displayed using a bold font are present in the current package.

The radio group on the right hand side determines what kind of operator will be generated when the property is inserted into the condition. By default the value is **IS SET** which means that no operator will be used. For example, if the property `VersionNT64` is focused and the *Is set* option is selected, when the *Add to condition* button is pressed, the string `VersionNT64` will

be added to the condition box. During run-time it will be then a simple check whether the property `VersionNT64` is set.

Other options generate different operators (where % is the name of the property):

Is unset	`NOT %`
Greater than	`> %`
Greater or equal to	`>= %`
Equal to	`= %`
Equal to (case insensitive)	`~= %`
Is not equal to	`<>`
Less or equal to	`<= %`
Less than	`< %`

FEATURES

With a click on the features button, packagers will get a list of features present in the current package. By entering a keyword into the input field above the list, the features are filtered down to those, which contain the keyword as part of their name.

The radio group on the right hand side determines what kind of feature state will be generated when the feature is inserted into the condition. By default the value is **Installed** which means that the sub-condition will return true only if the feature is installed. For example, if the feature `ABC` is focused and the *Installed* option is selected, when the *Add to condition* button is pressed, the string `!ABC = 3` will be added to the condition box. During run-time it will be then a simple check whether the feature state of feature ABC is 3, which means *Installed*.

COMPONENTS

With a click on the components button, packagers will get a list of components present in the current package. By entering a keyword into the input field above the list, the components are filtered down to those, which contain the keyword as part of their name.

The radio group on the right hand side determines what kind of component state will be generated when the component is inserted into the condition. By default the value is **Installed** which means that the sub-condition will return true only if the component is already installed. For example, if the component `ABC` is focused and the *Installed* option is selected, when the *Add to condition* button is pressed, the string `?ABC = 3` will be added to the condition box. During run-time it will be then a simple check whether the component state of component ABC is 3, which means *Installed*.

OPERATORS

With a list on the **OPERATORS** button, packagers get a list of symbol tiles, each a representation of a typical operator.

Snippets are commonly used combinations of properties, operators and values. Use them to quickly define repeatedly required conditional statements.

Clicking on a property, operator, or snippet automatically adds the string representation to the conditional statement. The phrase is added exactly at the position defined by the current cursor location.

When a property is added to the conditional statement, it is displayed in a blue, bold font style. Operators are displayed in a green, bold font style.

The snippet list is configurable, the advanced section

User Object Manager
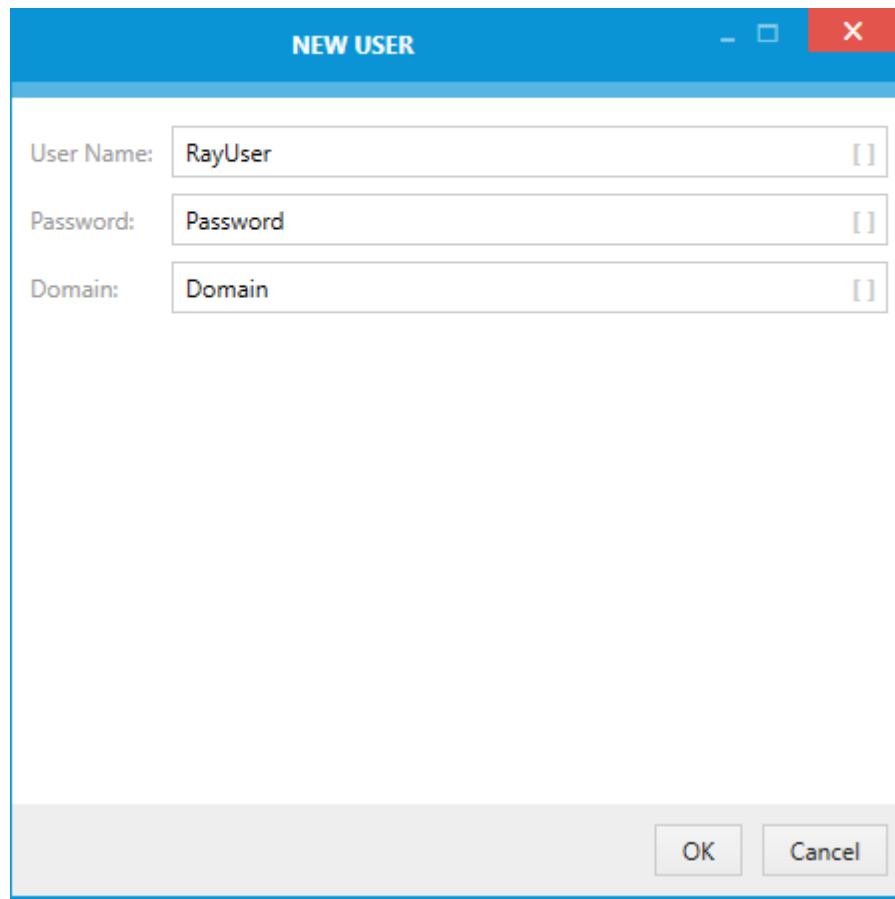
The user object manager is an interface required to create, edit, and remove user objects. These objects are required to define credentials used for database connections and the like. The user object manager is not accessible directly, but only in the context of another complex RayPack data object, which needs a user to work. This is typically given for SQL database objects and SQL scripts. User objects are custom RayPack data objects, and therefore stored within a custom RayPack installer database table: RPUser.

The main dialog of the user object manager is the Select User dialog. It shows a list of already existing user objects for the current packaging project, and allows to add, edit and remove user objects.
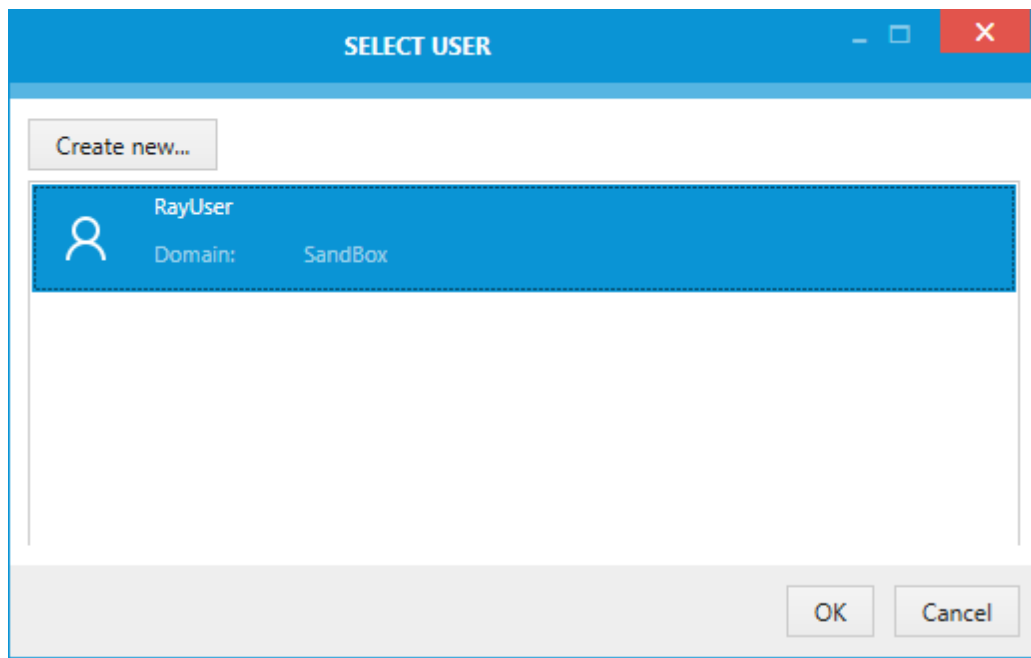
To add a user

In order to add a new user object, the **Create new...** button at the top of the Select User dialog has to be clicked.

The **New User** dialog is loaded, waiting for the following user properties to be defined:

User Name
The user name as it is known at the target system environment. This is a mandatory property each user object needs to have. The user name is a string of max. 255 characters length, which does not have to be unique within a packaging project.

The square brackets at the right-hand side of the input field indicate, that it is a special input field, allowing packagers to use Properties and other variable packaging project information for the value definition. Please refer to the MSI formatted string field section for further details on how to handle this typical kind of user interface control in RayPack.

Password
The password required by the user to log in. This is an optional user property. The password is a string of max. 255 characters length.

Be aware:
Please be aware that passwords are displayed in plain text. Therefore, they are visible for everybody who has access to the packaging project resources.

The square brackets at the right-hand side of the input field indicate, that it is a special input field, allowing packagers to use Properties and other variable packaging project information for the value definition. Please refer to the MSI formatted string field section for further details on how to handle this typical kind of user interface control in RayPack.

Domain
If the user profile is not local on the target machine, but is valid within a specific domain, this information may be entered here. The optional domain property is a string of max. 255 characters length, which may be left empty.

The square brackets at the right-hand side of the input field indicate, that it is a special input field, allowing packagers to use Properties and other variable packaging project information for the value definition. Please refer to the MSI formatted string field section for further details on how to handle this typical kind of user interface control in RayPack.

Once all user properties are defined, the **OK** button has to be used to save the new object into the `RPUser` installer database table. Hitting Cancel discards the entered information, and closes the New User dialog without actually creating a new data object.

To select a user
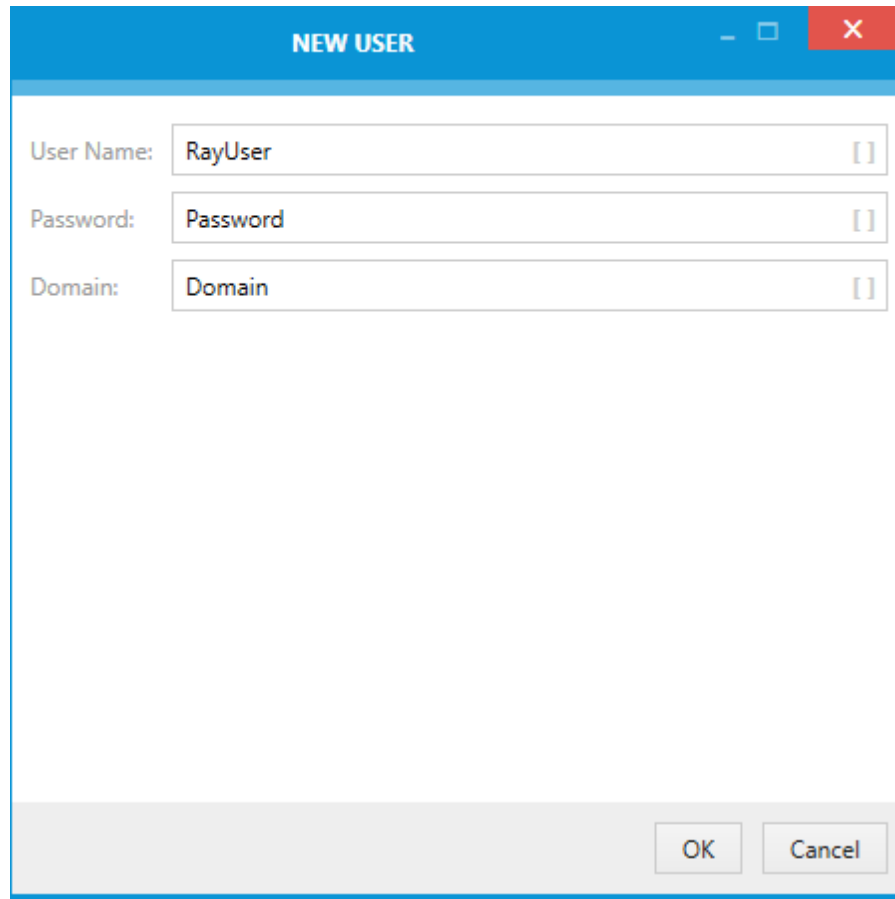
To actually select one of the existing user profiles from the Select User dialog, there are two options provided: Packagers have to

- **double-click** the desired user object item from the list
  or
- **select** the desired user object item from the list, and click **OK**

Either way, the Select User dialog is closed, and the relation to the user object is taken over into the triggering control input, that originally lead to the Select a User dialog display (e. g. the user input field within an SQL database management dialog).

To edit a user

In order to edit a user object, a **right-click** at the user profile item within the Select User dialog has to be performed. The **context menu** contains an option **Edit**, which launches the **Edit User** dialog when it is clicked:

User Name

The user name as it is known at the target system environment. This is a mandatory property each user object needs to have. The user name is a string of max. 255 characters length, which does not have to be unique within a packaging project.

The square brackets at the right-hand side of the input field indicate, that it is a special input field, allowing packagers to use Properties and other variable packaging project information for the value definition. Please refer to the MSI formatted string field section for further details on how to handle this typical kind of user interface control in RayPack.

Password

The password required by the user to log in. This is an optional user property. The password is a string of max. 255 characters length.

> **Be aware:**
> Please be aware that passwords are displayed in plain text. Therefore, they are visible for everybody who has access to the packaging project resources.

The square brackets at the right-hand side of the input field indicate, that it is a special input field, allowing packagers to use Properties and other variable packaging project information for the value definition. Please refer to the MSI formatted string field section for further details on how to handle this typical kind of user interface control in RayPack.

Domain

If the user profile is not local on the target machine, but is valid within a specific domain, this information may be entered here. The optional domain property is a string of max. 255 characters length, which may be left empty.

The square brackets at the right-hand side of the input field indicate, that it is a special input field, allowing packagers to use Properties and other variable packaging project information for the value definition. Please refer to the MSI formatted string field section for further details on how to handle this typical kind of user interface control in RayPack.

Once all user properties are modified as required, the **OK** button has to be used to update the new properties. Hitting Cancel discards the entered information, and closes the Edit User dialog without actually updating the data object.

> **Note:**
> User objects may very well be related to several parent data objects, such as SQL databases, or SQL scripts. Editing the user itself changes the user profile properties of all connections this specific user object is related to.

To remove a user

In order to remove a user object, a **right-click** at the user profile item within the Select User dialog has to be performed. The **context menu** contains an option **Remove**, which immediately removes the user profile from the Installer database table `RPUser` when it is clicked.

> **Be aware:**
> When the Remove option is selected, the user profile is immediately deleted from the packaging project installer database. Relations to object such as SQL databases and SQL scripts are updated when the user profile is deleted. Therefore, handling user profiles may lead to incomplete connection credentials for database related data objects within RayPack packaging projects. Please double-check the completeness of user related data objects before a target package is actually build from the project file.

MSI Formatted String Field

Various Windows Installer fields (for example registry value, shortcut arguments etc.) support a special Windows Installer syntax. The following selected formatted values are supported:

- `[PropertyName]` – resolves to the value of the property PropertyName or to the empty string if there is no such property.

- `[FolderId]` – resolves to the full path (with trailing slash) of the folder FolderId or to the empty string if there is no folder or property with such name.

- `[#FileId]` – resolves to the full path of the file FileId or to the empty string if no such file is defined in the package.

- `[!FileId]` (only for INI files and registry value) – resolves to the full file path of FileId using the 8.3 naming, or to the empty string if no such file is defined in the package.

- `[$ComponentName]` – resolves to the full path of directory (with trailing slash) to which ComponentName is installed or to the empty string if no such component is defined in the package or if the component is not marked for installation.

- `[%EnvironmentVariableName]` – resolves to the value of the environment property EnvironmentVariableName

- Special syntax:
    - `[\[]` – escaped opening bracket
    - `[\]]` – escaped closing bracker
    - `[~]` – NULL character

Using the formatted syntax is a great way to avoid hardcoded paths and complicated configurations/adjustments for multiple environments. For example, instead of hardcoding the database name, you can create a property containing its name and reference this property in the formatted text fields everywhere. Any subsequent update or adjustments will require just one change in the package.

Using the syntax suggestions

In several places in RayPack, fields marked with a pair of brackets, such as the value field for Registry value items displayed below, contain automatic functionality showing suggestion in as-you-type mode.
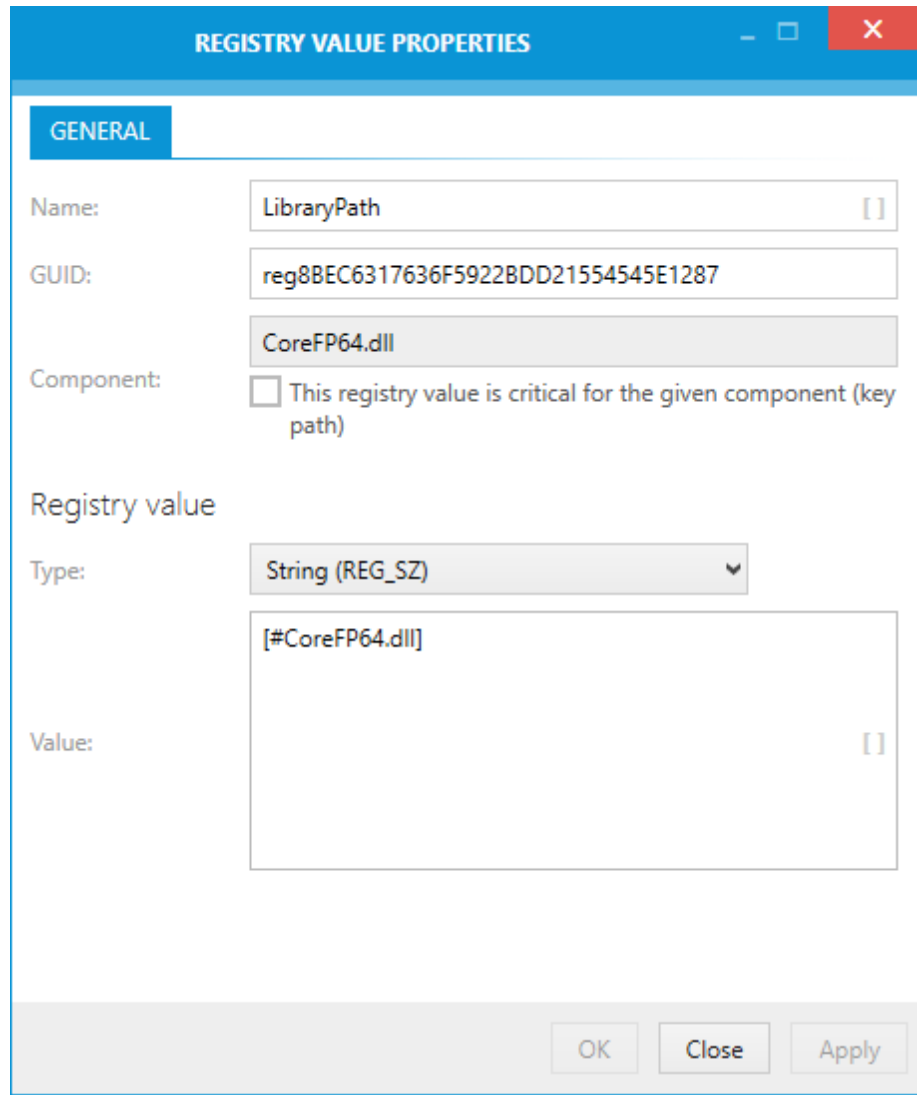
In order to get the suggestions popup, type the opening square bracket ([) into the text field. The suggestions for folders and properties are shown immediately. Continue typing to narrow the list of results. The list contains a fully expanded syntax and a preview of the resolved value on the right hand side. Simply click any of suggested entries or use arrows and ENTER/SPACE to automatically complete the typed value with the selected suggestion.

- Type [# and continue typing to show only files
- Type [% and continue typing to show only environment variables
  In this mode, the preview will show the actual value of a specified environment variable on the machine RayPack is installed
- Type [$ and continue typing to show only components
- Type [! and continue typing to show only files (8.3 naming convention, available in the registry property dialog only)
- Type [\ to show suggested escape sequences

In one text field many MSI formatted strings can be used. Simply insert your cursor in the desired

place and press the opening bracket ([) to get the suggestion list.

Note:
Once the formatted value is inserted it cannot be edited anymore with the suggestions popup. To correct the entry, remove the formatted value and type the opening square bracket ('[') to select the right formatted value.

Object Permissions

Object permissions are managed within the PERMISSIONS tab of the Property management dialogs for several object types, such as files, folders, registry keys and values. Refer to the specific chapters to get information on how to open the required properties dialog. Once the PERMISSIONS tab is visible, the following interface is available:

User based object permissions

Packagers may define permissions for several users or user groups. To do so, for each user or user group the following steps have to be executed in turn:

Define user or group
Click on the Add user button on the left side of the PERMISSIONS tab interface. A dialog is displayed, offering three different user definition methods:

- Predefined user group
  Select one of the available standard groups as target for the permission rule

- Predefined user name or SID
  Set permissions for a single user by user name. The entered value may contain the domain specification or refer to a local user account. RayPack will not validate the given value, since it will be resolved at run time.

- Dynamic at run time
  Apply the permissions on the user actually performing the installation

Once the right user or group is selected, click OK to add the new user object to the list of already existing user based permission rules.
The dialog is closed and the list on the left side of the PERMISSIONS tab interface is automatically updated with the new object.

To remove a user / group object from the list, right-click it and select Remove from the context menu. Please note that this remove is executed immediately without an additional confirm dialog.

Set permissions

Select an object from the list of available users / groups with a left click to edit the permissions for that specific user / group object.

The current permission settings are displayed. Decide to allow or deny **standard permissions** on the file by activating the checkbox in the Allow or Deny column:

- Full access - allows to use all standard Windows operating system procedures with the file
- Read - allows to open the file in read-only mode
- Write - allows to open and modify the file
- Execute - allows to run executable files

If the standard permission options do not suffice, use the **CUSTOM PERMISSIONS** button to define more precise rule-sets within the CHANGE CUSTOM PERMISSIONS dialog.

This dialog contains a more detailed list of permissions that can be controlled for a file. If one of the standard permissions (see above) has already been allowed or denied, the custom permissions dialog is a good place to check what exactly is granted by it in detail.

For further details regarding file and folder permissions, refer to Microsoft TechNet.

Inheritance

Activate this checkbox if the permissions for the file have to include inheritable permissions from the files parent object. Note that individual settings for the file always precede common inherited settings.

Configuration

Decide whether inheritable permissions should be propagated to all children of the current object whilst keeping their potentially existing custom user based permissions, or if potentially already existing permissions should be replaced by the inheritable permissions defined by the current object.

Select a Component

The select a component dialog is used wherever a interface view within RayPack requires advanced component control facilities to specify desired project behavior.

The dialog offers a flat list of all components that have already been created within the packaging project. Users can simply **select** one of the components by a left-click, and use the **OK** button to make the selection as required by the view which offered the select a component dialog.

Beyond that, the following additional options that might be required to actually be able to make the final component selection are available.

If the options of the Select a component dialog cannot be used to select the right component, users close the dialog with the **Cancel** button. If a component is selected within the dialog, the selection is not taken over into the view which was the origin of the dialog usage.

Add a new component

Users add a new component to the list by

- Clicking on the **Create** button at the upper left corner of the dialog, and selecting the **Create a new component** item from the displayed options menu.
- Hitting **Insert** on the keyboard
- **Right-clicking** anywhere inside the listing of existing components and selecting **New component** from the **context menu**.

All options lead to the same result: a newly created component with the default name component. If a component with the same name already exists, an automatically incremented index is added (e. g. NewComponent_2).
Read on to find out how that default name can be modified:

Rename a component

To rename a component, users **select** it within the listing of components, and either hit **F2** on the keyboard, or **right-click** the component and select **Rename** from the **context menu**. Either way, the component name is set into direct inline edit mode, which allows to modify the name directly within the listing. Once the desired component name is set, hitting **Enter** on the keyboard saves the changes.

The new name of the component must follow the guideline for component names:

- Component names must be unique, non-empty values
- Component names may consist of ASCII characters A-Z (a-z), digits, underscores (_), or periods (.)
- Component names must start with an underscore or a letter

If one of these rules is broken, RayPack shows an error icon. Hover over the icon to display the error message. Alter the entered name to solve the described conflict and be able to save it by hitting Enter.

To leave the direct edit mode without trying to save the changes, the **Escape** key has to be used.

Assign a component to a feature

Newly created components are by default not assigned to a feature. Since components have to be assigned to at least one feature in order to be able to build a valid Installer package from a project, it is highly recommended to establish a feature relation for every new component as soon as possible,

To accomplish that:

- Users right-click the component within the component list, and select **Assign to feature...** from the **context menu**.
- The Select features dialog is displayed. Select one or more features from the displayed list and use the **OK** button to establish an assignment to the component. If required, use the control key to multi-select items.
- Abort the assignment process by using the **Cancel** button within the Select features dialog.

To delete the relation between component and feature(s), switch to either the FEATURES or the COMPONENTS view of the Advanced mode, since this option is not available for execution from the Select a component dialog.

Search the component list

Since the list of components stored within a packaging project may grow to an uncomfortable size, the keyword search is a handy addition to the Select a component dialog. An input field is available at the upper right corner of the dialog. Simply enter the search term letter by letter to reduce the list to those components who contain the keyword within their name. To restore the full component list, remove the keyword from the input field.

Select a Folder

Selecting a folder from a packaging projects directory structure is a quite common task, therefore the Select a folder dialog was designed to handle it as efficient and convenient as possible.

The dialog shows the list of directories available within the currently opened packaging project. The display scheme follows the RayPack directory tree color scheme, which allows to quickly navigate to essential folders in no time. Besides the actual folder name, the list contains a column with the internal folder identifier, which enables packagers to access the directory tree structure as well from the packaging logic angle.

The list is sortable: Clicking on any column header sorts it ascending or descending.

Select a folder

To select a folder from the list, simply left-click it, and use the **OK** button at the bottom of the dialog to transfer the selection to the view that originally demanded the folder selection to be executed. The Select a folder dialog is closed automatically.

To close the dialog without taking over the selection into the view that originally demanded the folder selection to be executed, users click on the **Cancel** button, or simply hit **Escape** on their keyboard.

Beyond selecting a folder from the existing structure, this dialog also offers additional functionality to manipulate the directory tree:

Add a new custom folder

Users can trigger the creation of a new custom folder by using several methods. Either way, the parent of the new folder should be selected first, with a left-click on its name within the directory tree. Once the parent is selected, users can create a new folder by:

- Using the **Add or import** button at the upper left corner of the dialog, and selecting **Folder** from the displayed options menu
- Hitting the **Insert** key on the keyboard
- Right-clicking the parent folder, and selecting **New folder** from the **context menu**

The new folder is created with a default name (e. g. New folder (3)) and default identifier (e. g. [NEWFOLDER3]. Since the default values are most definitively about to be changed, the new folder is immediately set into direct inline edit mode, which allows the user to simply start typing to change the folder name.
Please make sure to follow the guidelines for folder names.

Hit enter to save the new folder name. If it is regarded to be invalid, RayPack shows an error icon. Hover over it to display the corresponding error message. Modify the name according to the given hints and hit Enter again to save the new value.

Add a new predefined folder

To add another one of the Windows Installer predefined system folders to the directory tree structure, users have to click on the **Add or import** button at the upper left corner of the dialog, and select **Predefined folder** from the options menu.

The **SPECIAL INSTALLER FOLDERS** dialog is displayed, containing a list of all known predefined folders. Those of them which are already available within the project are displayed in a pale font and icon style, with the checkbox within the outer left column already activated. Already existing predefined folders cannot be removed from within this dialog.

Activate the checkbox of one or more not yet available predefined folders, and click **OK** to add them to the directory structure and return to the Select a folder dialog.
Alternatively, users can click on Apply to add the folder(s) and keep the SPECIAL INSTALLER FOLDERS dialog open for further manipulation.

When the user returns to the Select a folder dialog, the new folders are already available within the tree structure. Since predefined folders may be nested, the arrow icons left of parent folder names have to be used to expand new child levels.

Rename a folder

Users trigger the renaming of a folder by applying one of the following actions:
- **Right-click** a folder and select **Rename** from the **context menu**.

- **Select** a folder and hit **F2** on the keyboard.

The folder name is set into the direct inline edit mode. See the section above for details on how to proceed the renaming procedure.

> 📎 **Note:**
> Renaming is not available for predefined folders.

MSIX / APPX Projects

Introduction

Microsoft® Operating Systems have a deep history with Windows application installers. Availability of different techniques spanning from legacy installers through Windows Installer, Click-Once, and App-V created several challenges for developers packaging their apps for legacy and modern environments. Due to the fast growth of mobile and other non-PC like tablets, phones, game consoles, and HoloLens, it has become critically important to unify the installation frameworks and come with a solution that is cross-platform and supports wide range of apps and technologies.

The answer to these challenges has been announced. The new format – named MSIX – is internally based on the APPX technology stack which provides a solid containerization foundation, yet expanding on previously unavailable enterprise oriented features. This is also symbolized by its name where APPX and MSI are combined to MSIX. The technology is natively implemented in Windows 10 (starting from builds 1809 October Update) and is combining the best of MSI, App-V, and Click-Once. Specific goals of the MSIX are:

- Multiplatform format (not limited to Windows OS).
- Security, containerization, and isolation of running apps.
- Fast install and clean uninstall routines made easy, preventing any kind of Windows rot.
- The ability to publish apps in the Microsoft Store / Microsoft Store for Business and Education.
- Built-in licensing and monetization features and tons of other APIs including the integration with Cortana, Sets, Timeline, etc.
- Expanded enterprise features, including easy customization of existing vendor packages (Modification Packages), extended compatibility features, and fixes (Package Support Framework).

Where migration to the Universal Windows Platform (UWP) is not possible (especially for apps which are no longer maintained and for certain Line-Of-Business products) Windows 10 implements a bridge to the UWP world, which allows running old Win32 apps on the new platform, with the new MSIX installer and all benefits of it, yet requiring no changes to the code. As the adoption of Windows 10 grows, more and more apps are expected to be packaged in the new format, and slowly modernized to use the UWP benefits. This should finally join the development and packaging world in a coherent world, using same tools and principles.

Opening MSIX Projects

Once an MSIX or APPX project / package is opened, the following view is shown:
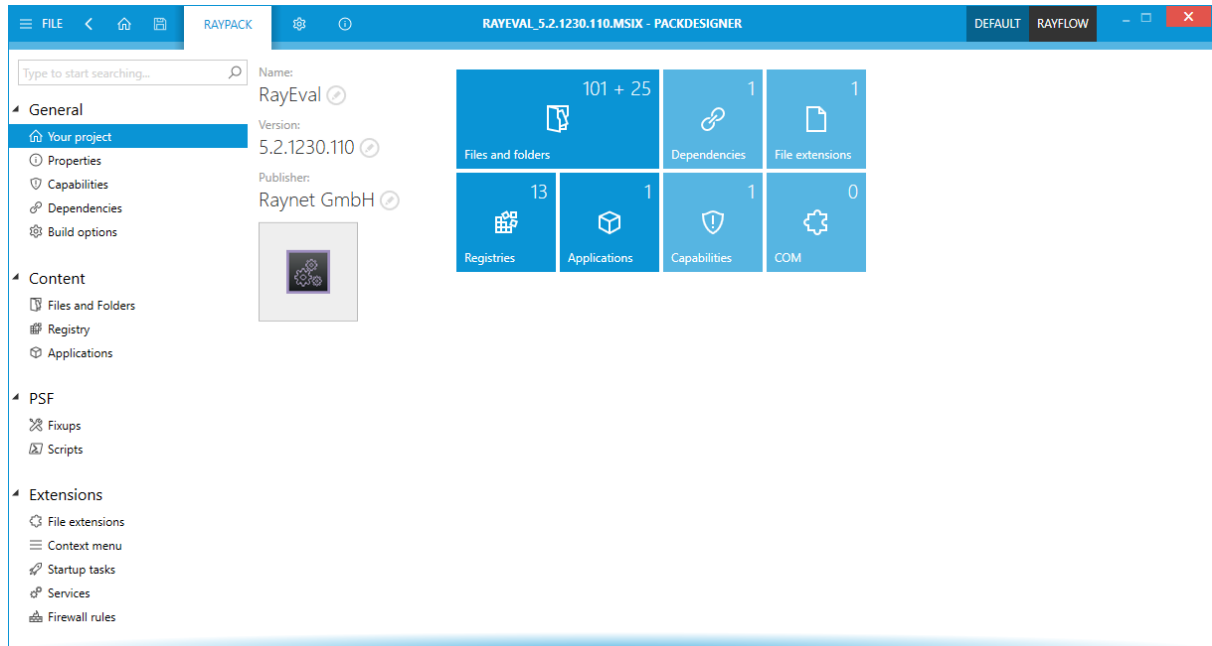
The left sidebar contains common categories of actions and features supported by RayPack 6.5.

- **Your Project** - shows the dashboard with an overview of available features.
- **Properties** - the properties of the package, including its identity.
- **Capabilities** - the definition of functions which are required for the current package to work.
- **Dependencies** - the definition of dependent packages, supported target platforms, and base packages (for Modification Packages).
- **Build options** - various options used for building, including signature options and installer options.
- **File and folders** - the list of files and folders in the current package.
- **Registry** - the list of Registry keys and values in the current package.
- **Applications** - the list of applications (entry points) and their visuals.
- **Fixups** - the options for compatibility settings via PSF fixups.
- **Scripts** - the options for first-launch scripts, that can be executed before the app starts or upon closing.
- **File extensions** - the definition of the supported file extensions.
- **Context menu** - the definition of the verbs supported by this package.
- **Startup tasks** - the definition of startup tasks supported by this package.
- **Services** - the definition of packages services, supported since Windows 10 2004 (May 2020 Update).
- **Firewall rules** - the definition of firewall rules.
- **COM** - the list of COM published by the package.

Managing and Creating Modification Packages

Modification Packages are specially prepared MSIX packages retaining their own identity (name, version, publisher) and signature, but running within the same container as a preselected existing MSIX package. They are a great way for:

- The modification of vendor packages
- Delivering plugins
- Creating brandings

Since an MSIX Modification Package does not differ internally from a normal package, you can create and open them using the standard techniques (**File** > **New**, **File** > **Open**, etc.). To declare that the current project is a modification package, the dependency to the base package must be defined within the **Dependencies** screen.
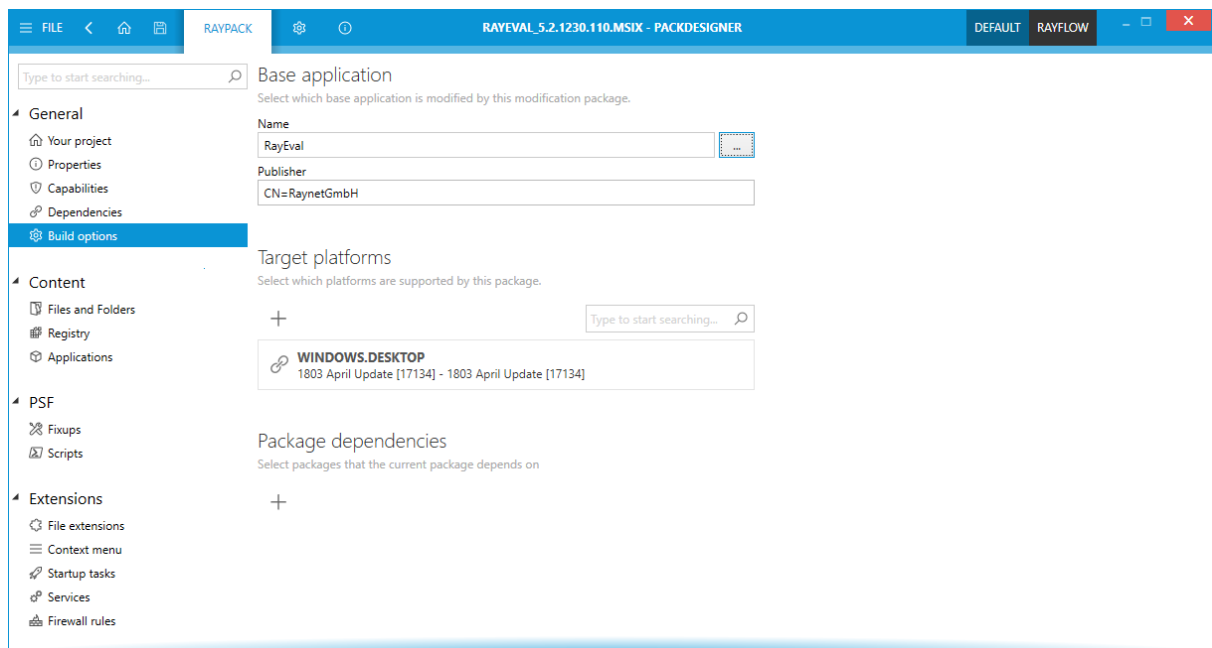
You can add the dependency manually (by providing the values for **Name** and **Publisher**) or have these values set automatically by pressing the **...** button.

Viewing Differences Between Base and Modification Packages

If a base package has been defined using the **...** button, some changes are visible in the product UI:

1. The **Capabilities** screen gets disabled. You cannot set capabilities for Modification Packages
2. In the **Files and Folders** view, the view changes to show the files present in the base package as shown below:
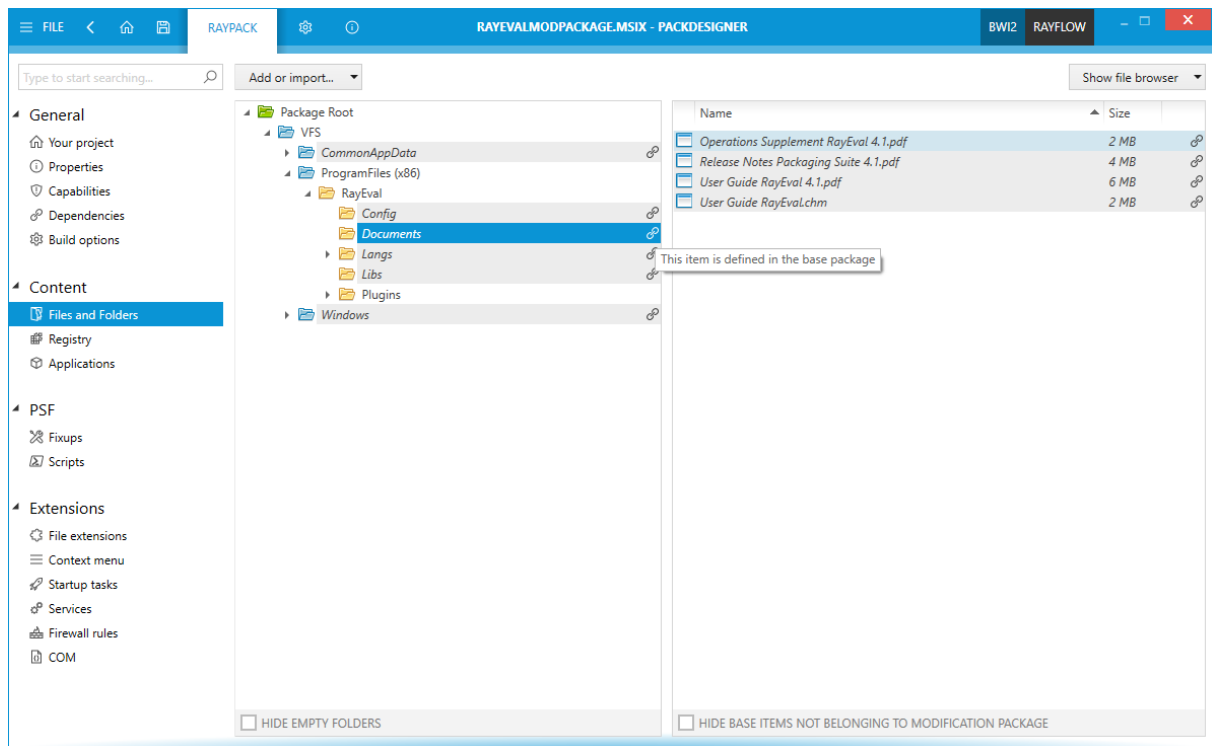
The items from the base package cannot be deleted or renamed. If a new item is added with the same name, it overrides the base item. Removing an item which overrides a base item restores the base item.

You can disable this enhancement by checking the checkbox **HIDE BASE ITEMS NOT BELONGING TO MODIFICATION PACKAGE.**

MSIX Core Support

MSIX Core is a way to have your MSIX packages run on older versions of Windows. While not retaining all features (including security, delivery optimization and several others which are exclusive to Windows 10 supported builds) they are still installable and provide a bridge experience and a single deployment (same MSIX package) for both modern and legacy systems.

In order to enable support for MSIX Core, a dependency on the target platform MSIXCore. Desktop is required. You can configure it in the **Dependencies** screen:

Package Support Framework

Package Support Framework is an open-source toolkit which provides package-specific shims for common compatibility issues and troubleshooting.

Being an extensible framework, the Package Support Framework supports shims for common problems (like file or registry access) using a concept of retours. The plugins bundled with RayPack allow the following:

- **Redirecting file access**
  https://github.com/microsoft/MSIX-PackageSupportFramework/blob/master/fixups/
  FileRedirectionFixup/readme.md

- **Electron framework**
  https://github.com/microsoft/MSIX-PackageSupportFramework/blob/master/fixups/
  ElectronFixup/readme.md

- **Dynamic Library**

- **RegLegacy**
  https://github.com/microsoft/MSIX-PackageSupportFramework/blob/master/fixups/
  RegLegacyFixups/readme.md

- **Tracing**

- **Breakpoint for a debugger**

To define a fix-up for one, more, or all applications and press the **Create new** button.
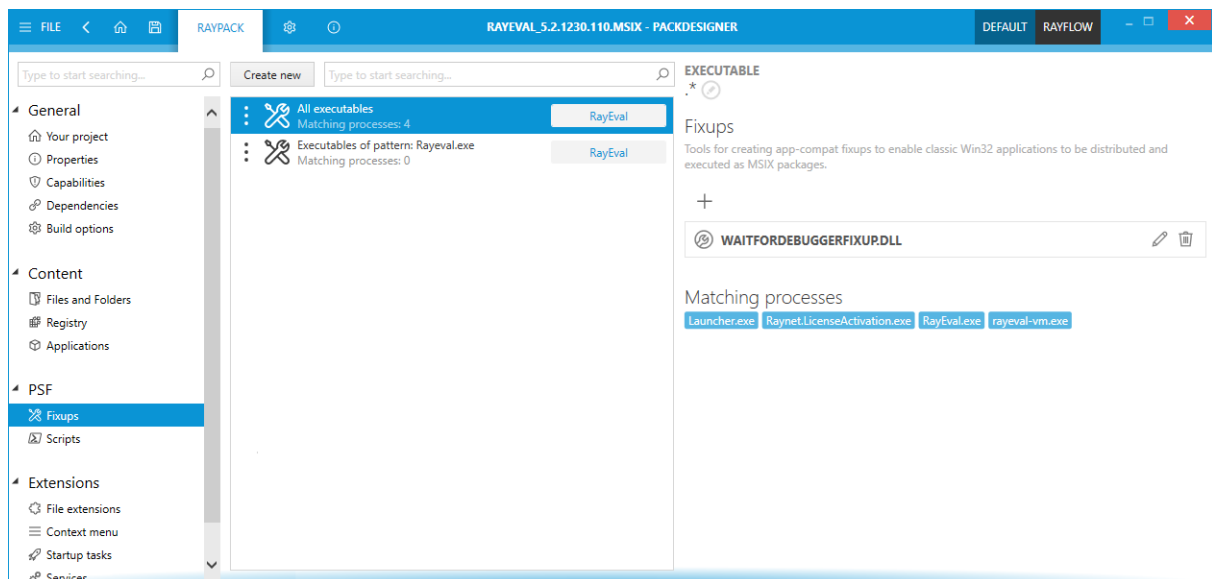
Depending on the fix-up plugin, a dedicated editor may be shown. For example, the following screenshot shows how to make the application rewrite the access to a specific path:
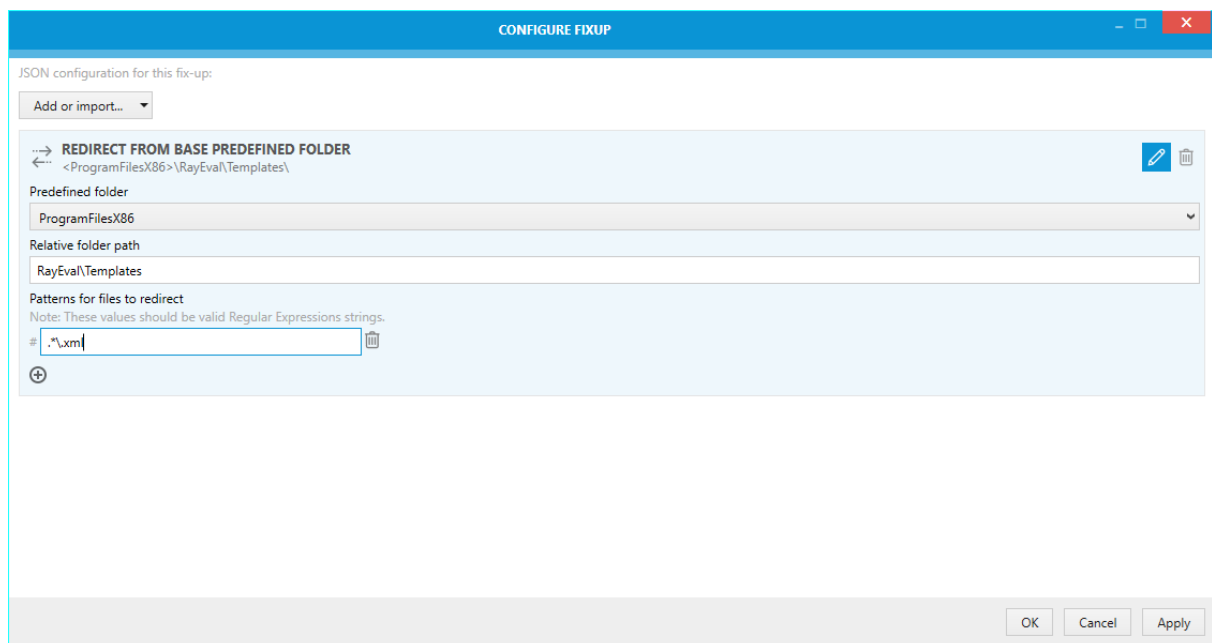
For other plugins, the JSON configuration editor may be shown if it is supported.

Building MSIX and .appinstaller files

The **Build options** menu provides commonly used options, used during the building MSIX files from RPPX projects or MSIX packages.
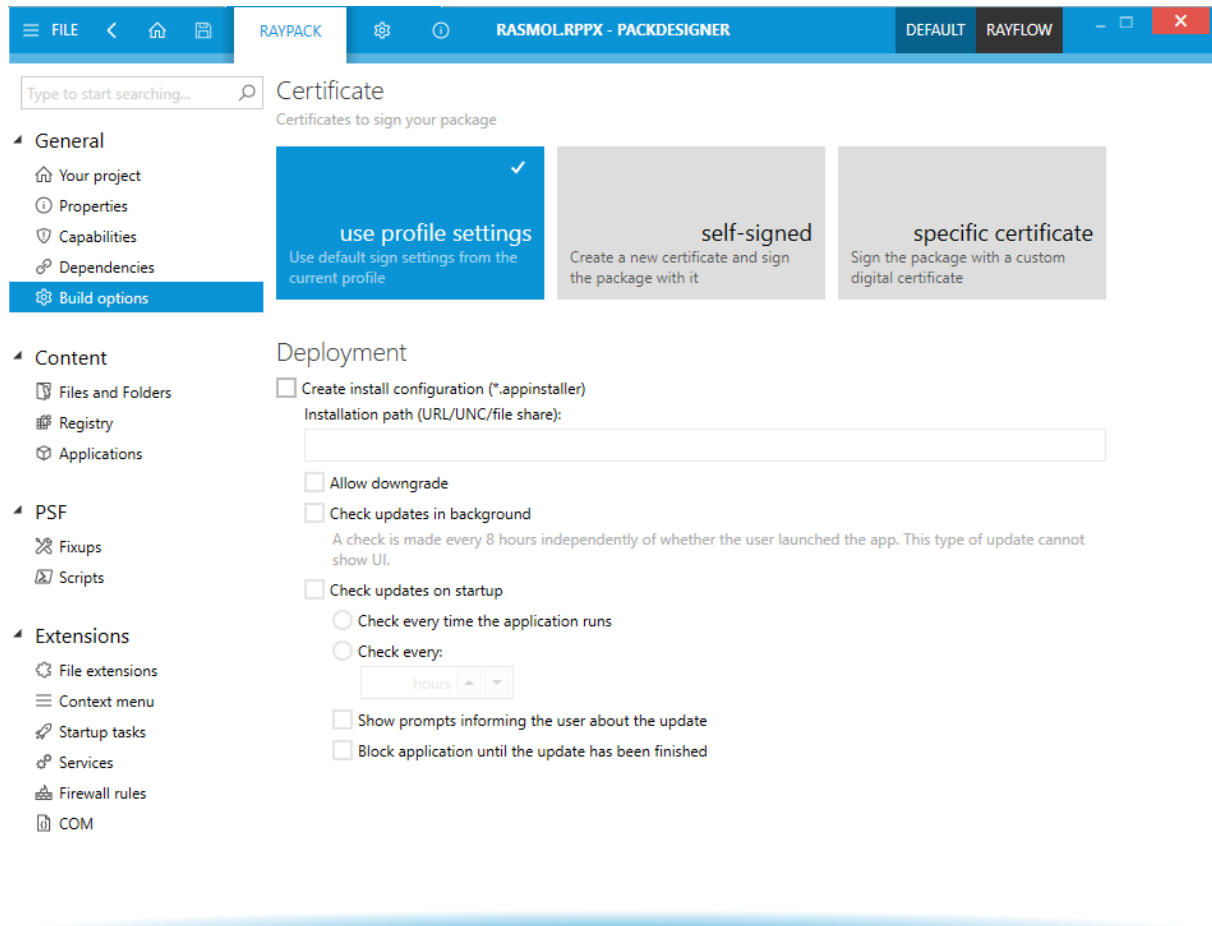
Certificate

This three-state menu provides three different ways of signing MSIX files:

- **use profile settings**
  When selected, the settings from the User Profile are used to build the package. This is recommended, as the **signing + tagging** screen is a central profile-based place to aggregate signing options.
- **self-signed**
  When used, a test certificate is created and put next to the MSIX file. The certificate is by default not trusted by any signing authority, it is the user's responsibility to import it to a correct certificate store. This is a good option for testing and troubleshooting if no specific code signing facility is provided.
- **specific certificate**
  Use this option if you have your own certificate, which is different than the one configured in

the current profile. RayPack will prompt for a password if certificate file requires it.

Deployment

These settings control whether and how to create supporting *.appinstaller* files. These files are used to setup simple file-based update facility, for simple managing of MSIX updates. Once an app is installed through an *.appinstaller* file (and not directly via *.msix* package), subsequent updates are checked on start-up or with a given time interval.

Create install configuration (**.appinstaller*)
This checkbox controls whether the automatic creation of *.appinstaller* files is enabled. Once disabled, the other options are also disabled.

Installation path (URL/UNC/file share)
This is a globally reachable path, from which the updates and the installer are going to be pulled. Usually, this is done via an UNC share. After creating an MSIX + *.appinstaller* package make sure to copy them over to this location. The path must not include the MSIX name nor *.appinstaller* file.

Allow downgrade
This setting controls whether newer apps can be replaced by older versions.

Check updates in background
This setting controls whether automatic updates are checked in the background by Windows 10.

Check updates on startup
If enabled, it can check for updates as the user launched the application. Various settings are available to refine the timing and behavior of an update:

- **Check every time the application runs**
  In this case, Windows 10 looks for updated version of an app every single time it is launched by the user.
- **Check every XXX hours**
  In this case, the update is checked upon start-up but not sooner than the specified number of hours since the last check.
- **Show prompts informing the user about the update**
  This setting enables, or disabled prompts shown to the user, if an update is found.
- **Block application until the update has been finished**
  If this setting is activated, the user is unable to launch the app until the update is finished.

Building MSIX app attach (VHD) files

With MSIX app attach, the application is completely detached from the OS it runs on, and can be dynamically attached and detached without any noticeable delay. This means a clean deployment, no need to prepare golden images and all benefits of MSIX technology. The technology is currently in preview, and can be tested starting from Windows 10 May 2020

Update (2004).

The configuration for MSIX app attach builds is the same as for any MSIX build (see more information about configuring for MSIX build in section Building MSIX and .appinstaller files). The only difference is the format, which is going to be a VHD with expanded content and a set of permissions required for a proper attaching. Additionally, RayPack writes 4 scripts which can be used to test the deployment: two for staging and registering, and another two for de-staging and de-registering.

More information about MSIX app attach:
https://docs.microsoft.com/en-us/azure/virtual-desktop/app-attach

MacOS projects

Upon selecting **create a new project** from the Home Screen, the decision on how the new project will be created can be chosen.

> **Note:**
> Availability of this option may depend on your license.
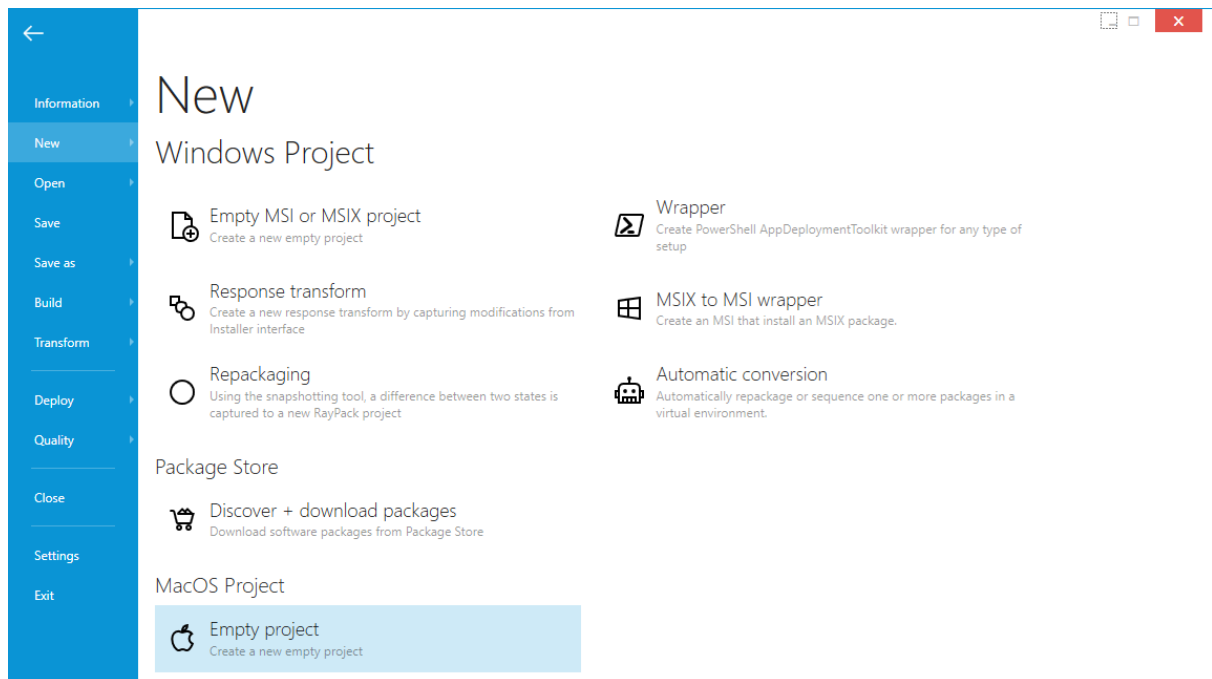
In order to create a new MacOS project, select the **Mac Project** radio button, and then press **empty project.**
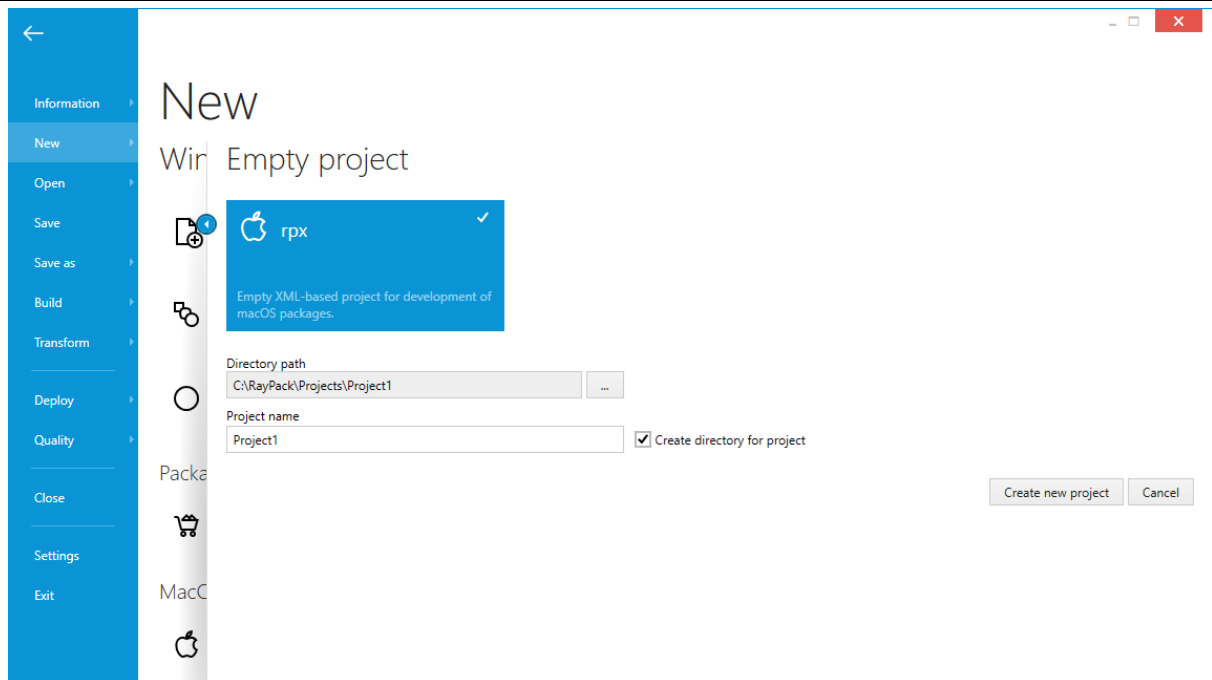
After providing required details, an empty `.rpx` project will be created in the selected folder.

> **Note:**
> RPX is internal RayPack project. In order to deploy packages to a macOS system, a `.pkg` file has to be created from the project..

Your Project

Your project view contains a summary of project content and its basic meta data (title, version and description).
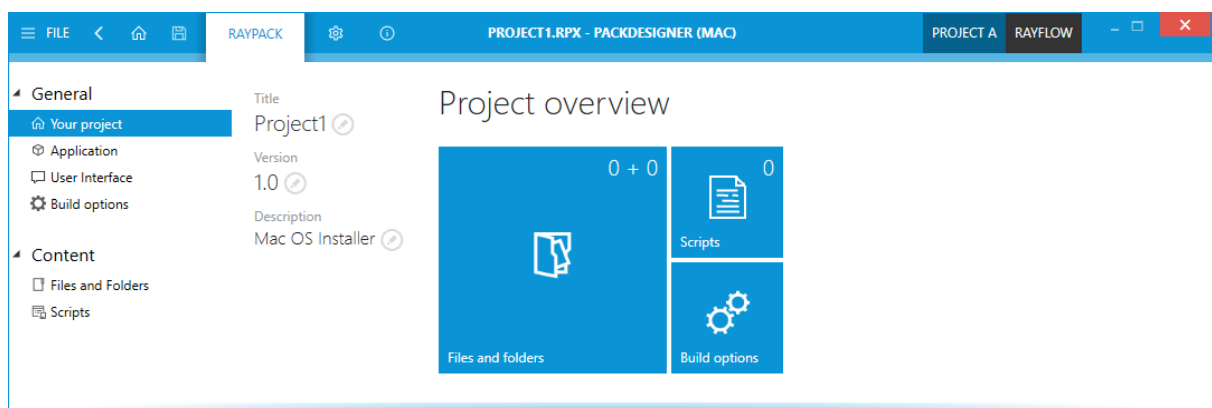
Tiles in the overview can be clicked to navigate to the respective screen content.

Application

The **Appication** view extends basic details and provides a way to change the data that can be found in the following list.

- Project title
- Version
- Description
- Install location (by default root folder)
- Unique package identifier
- Comments
- Authentication options

User Interface
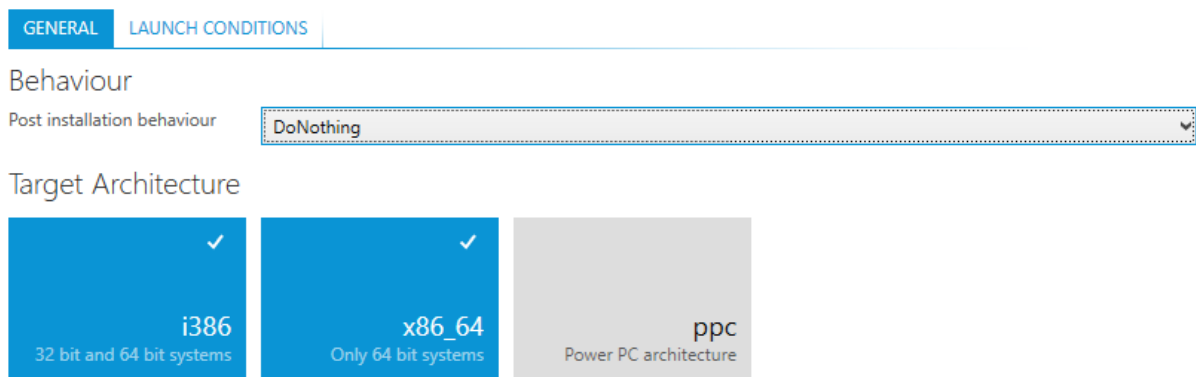
The **User interface** view aggregates several UI-related settings.

- Package background image
  - o Source path
  - o Alignment
  - o Scaling options
- UI language
- Custom rich-text for Welcome, Readme, License, and Conclusion messages

Build Options

The **Build options** screen defines settings that are used when building `.rpx` files into a proper `.pkg` format.

General

- **Post-installation behavior**
  Defines whether the installation require a restart, shutdown or logout after finishing the main process.

- **Target architecture**
  Defines which platforms are supported by this package installation. Multiple choice is allowed.

Launch conditions

- **Minimum allowed version**
  Select which is the minimum OS version supported by this package installation. Mac OS X 10.5 Leopard to macOS 10.13 High Sierra are supported.

- **Maximum allowed version**
  Select which is the maximum OS version supported by this package installation. Mac OS X 10.5 Leopard to macOS 10.13 High Sierra are supported.

Files and Folders

The **Files and Folders** view contains the definition of folders and deployable files.
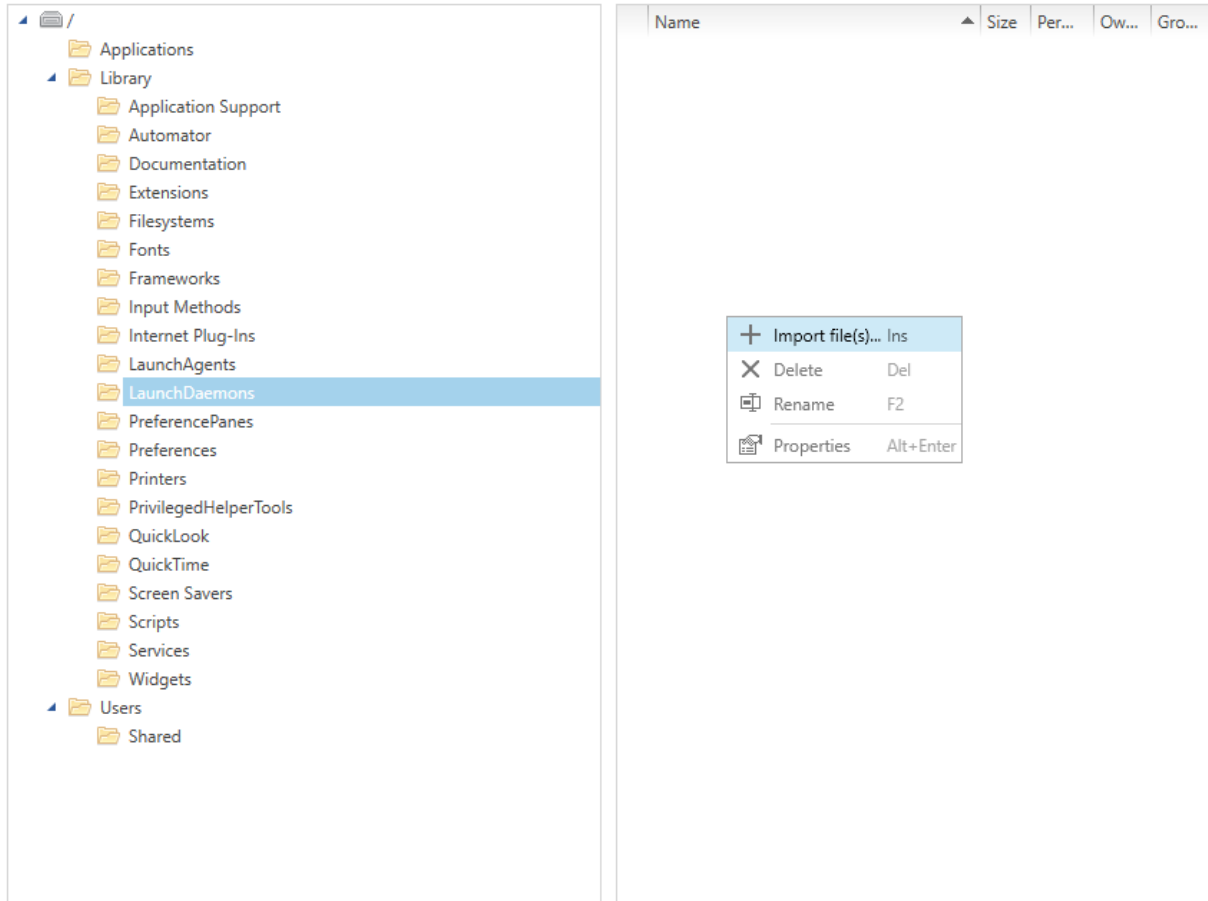
The left tree shows predefined folders that are recognized by MacOS systems.

- In order to create a new subfolder, right click on a parent folder and select **New folder** option.
- In order to import files to a folder, select it and invoke **Import file(s)...** function from a context menu (for a bulk import) or right click the empty area in the right panel and invoke **Import file(s)...** to be able to import single or selected files.
- In order to select the installation folder, invoke **Set as installdir** from a context menu of the selected item.

To change properties of a permissions of a folder...

1. Select a folder (other than a predefined one).
2. Right click on it and invoke **Properties** from its context menu.
3. A dialog will be shown, allowing to change the name, owner, group and access rights.
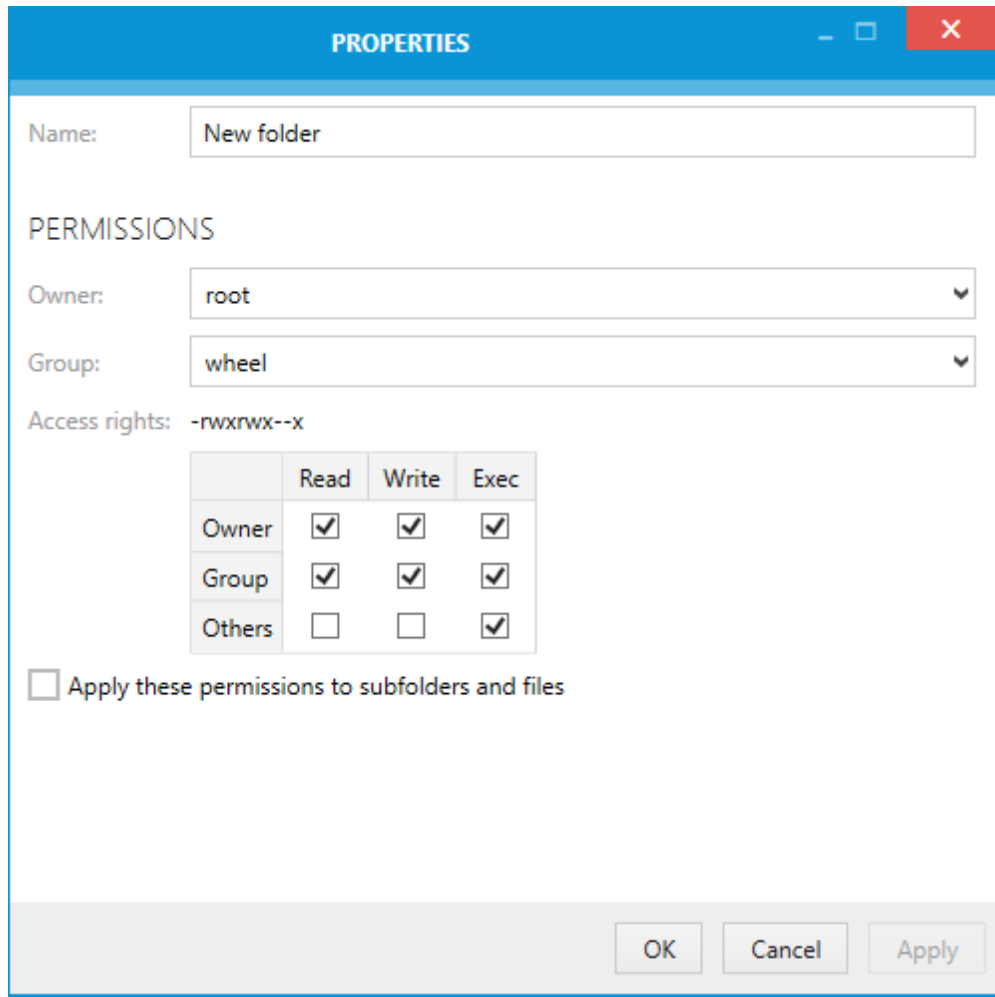
To change properties of the permissions of a file...

1. Select a file in the right panel.
2. Right click on it and invoke **Properties** from its context menu.
3. A dialog will be shown, allowing to change the name, owner, group, and access rights.

Scripts

The **Scripts** view provides management capabilities for scripts executed by the package installer.
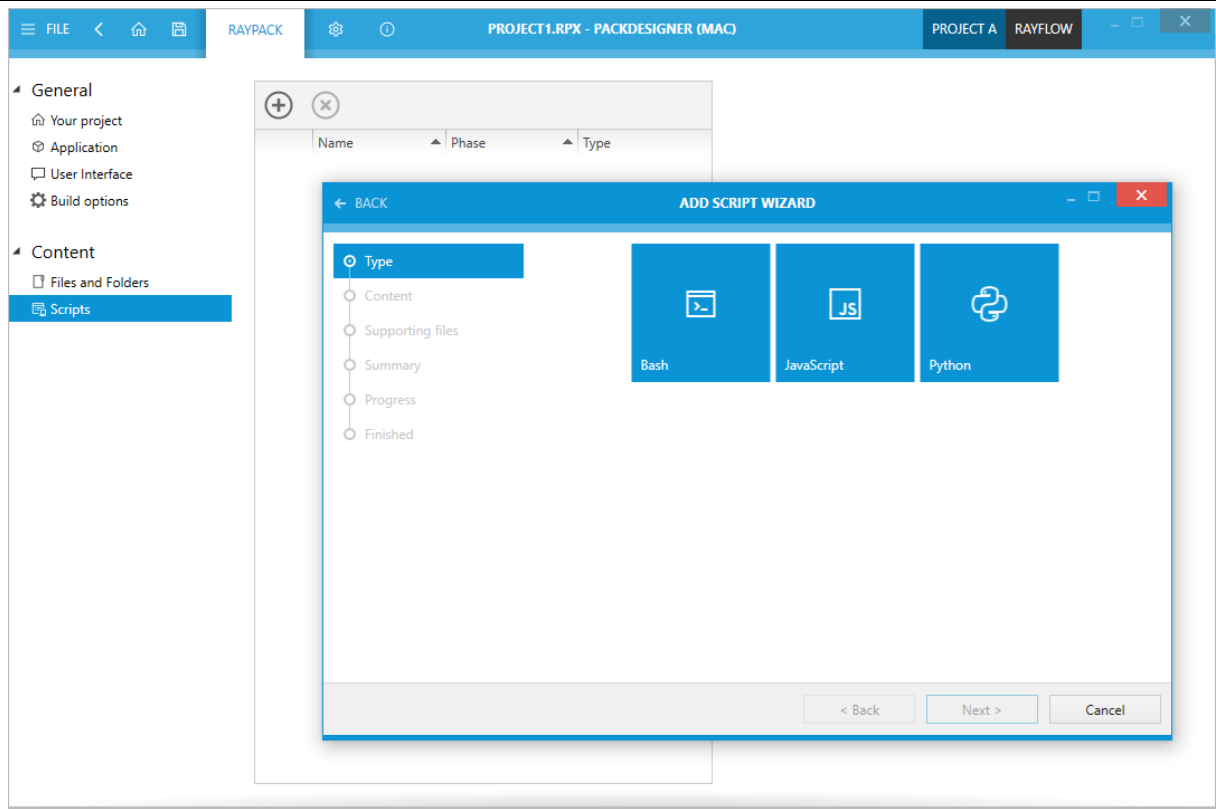
To add a new script, press the **+** icon, and then follow the steps presented by the **Script** wizard.

The following language are supported.

- Bash
- JavaScript
- Python

Building .pkg Files

In order to build a MacOS package in the `.pkg` format, press **FILE** > **Build** and select the `.pkg` extension.
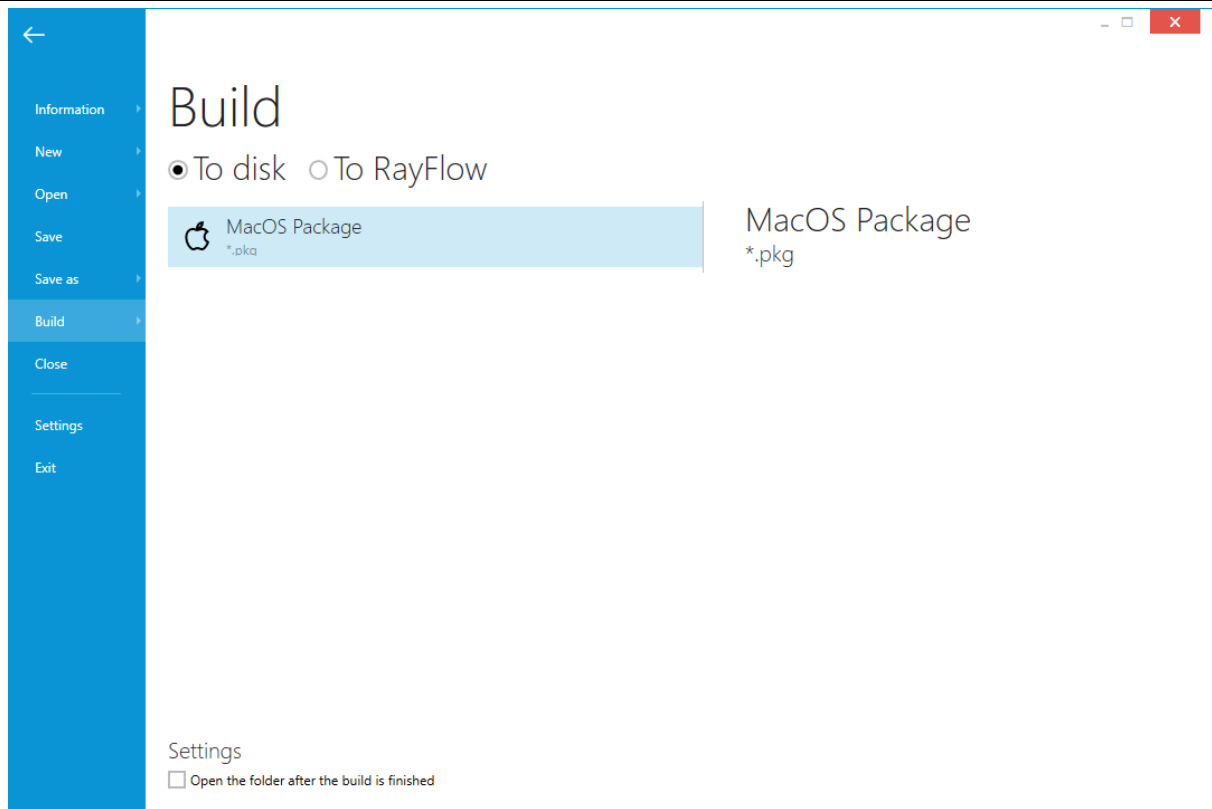
The project and its content will be built. Build options can be changed inside the **Build options** screen.

PackBot

PackBot enables IT professionals to convert MSI and non-MSI setups to other formats, using a preconfigured pool of virtual machines. By using a combination of different options, the following scenarios are achievable:

- Repackaging of a single app on a virtual machine.
- Silent bulk repackaging of many packages at once.
- Bulk conversion from MSI and EXE to App-V 4.6, App-V 5.x, and Thin-App.

The flexibility of workflows is backed by smaller features which help to setup and configure apps automatically:

- The automatic recognition of silent command lines (for recognized setup types).
- The ability to make manual post-installation configurations.
- A dynamic and efficient management of idle resources to minimize overall repackaging time.
- The automatic use of the current RayPack profile, exclusion lists, and shared resources.
- A parallel processing of tasks.
- The automatic gathering of log files and analysis of exit codes.

Basic Concepts

Each PackBot session consists of tasks. A task can be understood as a request to convert one format into another using a virtual machine. RayPack supports the following input formats:

- Legacy setups (`.exe`)
- Windows Installer setups
- Custom setups and scripts (for example `.cmd`, `.vbs`, etc.)

The following output formats are available:

- **Windows Installer formats**
  - o Windows Installer database

o RCP project (PackRecorder project)
o RPP project (PackDesigner project)
- **App-V formats**
  o App-V 4.6
    - standard RayPack conversion
    - conversion via locally installed Microsoft App-V 4.6 Sequencer
  o App-V 5.x
    - standard RayPack conversion
    - conversion via locally installed Microsoft App-V 5.x Sequencer
- **Thin-App format**
- **MSIX (Desktop Bridge) format**

Each session may consists of one or more repackaging tasks, and has the following workflow:

1. The user defines the packages and the pool of virtual machines used to process them.
2. Packages are processed sequentially (if there is just one machine defined) or in parallel (if more than one machine is available).
3. The results are saved and users see the confirmation with logs, errors, and success messages.

For each task that converts to the Windows Installer format or Thin-App format or APPX / UWP format, PackBot snapshots the current state of the machine, performs the installation using the required package options and then makes the second snapshot. Snapshots are compared and the output is saved on a physical machine. For some formats (MSI, RPP, APPX / UWP) some additional post-processing may be executed on the physical machine.

Sequential Processing

Packages are processed sequentially if at least one of the following prerequisites applies:

- There is just one package in the queue.
- There is just one machine used by the defined packages.
- There is just one actual machine used by defined packages.
- The number of parallel operations has been limited to 1 by the user.

One of these assumptions introduces a concept of an actual virtual machine. This is understood as the following: any two machines are considered to be "unique" if:

1. They are pointing to different `.vmx` files (Workstation), or
2. if they have the same computer name (Hyper-V).

If two machines are defined from a single `.vmx` file but are using different snapshots, they are considered to be only one machine. In that case (should there be no more machines defined in the pool) the processing is sequential.

Parallel Processing

Packages are processed parallel if the following conditions are valid:

- The user has defined more than one task to process.
- The selected tasks use at least two different virtual machines (see the description of sequential processing for the definition of machine uniqueness).
- The user selected a parallelism level of 2 or higher.

If this is the case, packages are processed in parallel. Parallel processing makes it faster to repackage large numbers of products in a fully automated way.

Before starting with each package, PackBot finds the best candidate to provide the maximum efficiency. It allocates the virtual machines pool as effectively as possible, limiting idle times and race conditions between packages. This may lead to a non-linear processing (for example, PackBot may decide that it will be on average faster to start with first and third package in the list, leaving the second package for a later processing). The selection algorithm takes the following into account:

- Constraints on all open packages (it prefers packages having more constraints over packages having less constraints).
- The order in the list.
- The idle state of virtual machines.
- Some more factors are considered by the algorithm.

This process is fully automatic and requires no attention from the user.

Example 1

A user has a virtual machine representing his packaging environment. He clones this machine three times. He now has four machines all pointing to a different physical file. He defines a queue of 100 packages and assigns them to all four machines of the pool. He uses parallelism level 4, which means that four machines can be running at once. In this scenario, the repackaging of 100 applications will on average will be four times faster, because all four machines are running at the same time, processing four packages at once.

Example 2

A user has one virtual machine with the sequencer installed, and another one without sequencer. This two machines use different `.vmx` files. He now defines a queue of 100 packages. Out of these, 50 packages are to be virtualized and are assigned to the machine having the sequencer installed, and the other 50 are simply to be repackaged and are assigned to both machines. In this scenario, PackBot will run two packages at once, making sure that the App-V packages always run on the machine with the sequencer and that the other tasks run as much as possible on the machine without sequencer to not block the App-V related tasks. On average, the whole process takes two times less time as sequential processing.

Detailed information about process flow and runtime decisions are described in the chapter *Parallel Processing*.

Silent and Bulk Processing

A task defines a setup to be started and a corresponding command line to install the product. If the command line is fully silent (it ensures that no popups and prompts are shown, typically `/S` or `/Q`), then the package is installed, processed, and converted silently. If all tasks in the list are defined with a fully unattended command line, then the whole processing can be silent and run fully unattended without any user attention.

1. The machine is started and the user is signed in.
2. The first snapshot is captured.
3. The package is installed using the command line delivered by the user.
4. PackBot captures the second snapshot, compares it to the first one, and generates the output.

In order to process packages silently:

• Import all required setups to the list.
• Review the command lines used to install the products and ensure they run from the beginning to the end in an unattended mode.
• All virtual machines need to have auto-logon enabled (read more about it in *Best Practises and Recommendations* chapter).

Be aware:
A fully unattended installation does not give the user an option to interact with it (for example to manually configure an app, change settings in the UI, etc.). If this is required, consider to repackage the applications by using the **Delayed second snapshot technique**.

Interactive Processing (Aka "Delayed Second Snapshot")

In order to perform bulk repackaging or a simple repackaging on a virtual machine which can capture custom user configurations, the user can decide to delay the second snapshot. By using this option, the processing of each task looks like the following:

1. The machine is started and the user is signed-in.
2. The first snapshot is captured.
3. The package is installed using the command line delivered by the user.
4. PackBot stops at this point and displays a window prompting the user to continue when he is finished with his configuration.
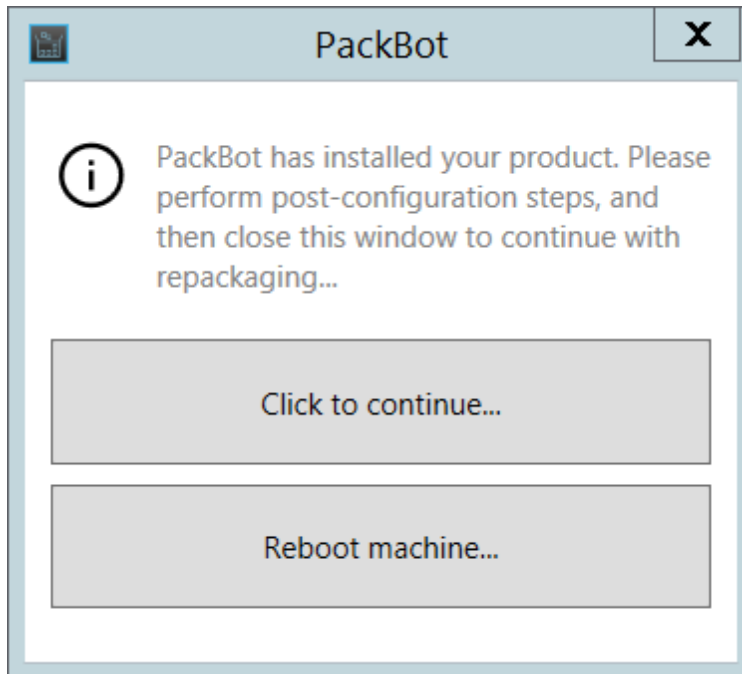
5. The user configures the product (starts the application, changes settings, etc.).
6. The user presses the **Click to continue...** button to continue.
7. The user can also reboot the machine by either pressing **Reboot machine...** or manually from **Start menu**. In any case, PackBot restarts after the reboot is complete and continues the processing. The user is able to reboot regardless of task settings once the **Delayed second snapshot** options is active
8. PackBot captures the second snapshot, compares it to the first one, and generates the output.

> **Be aware:**
> Delaying the second snapshot blocks the further execution until the user intervenes and presses the button to continue. Users should watch the console of their virtual machines to check which machines require attention and are ready to continue.

Creating Tasks

When PackBot is started, initially the list of tasks is empty and the following view is shown.
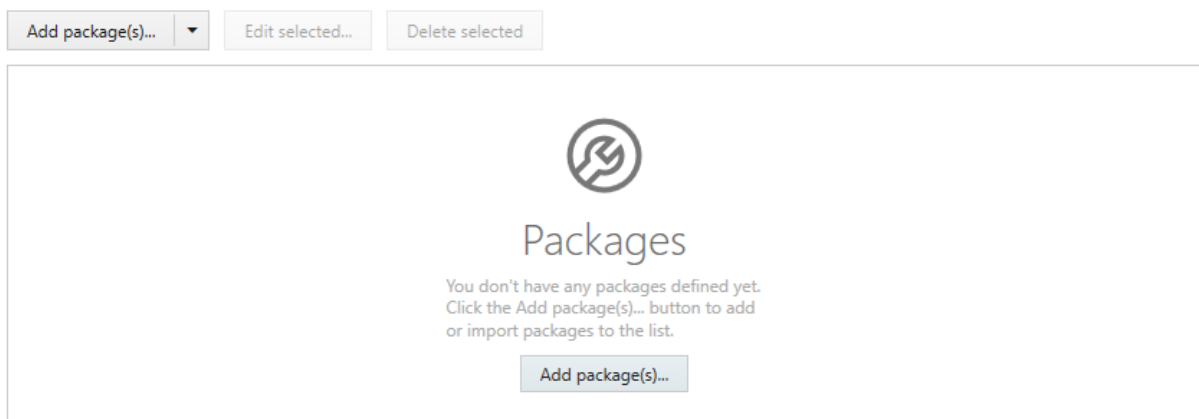
Click the **Add packages(s)...** button to show a file picker. Importing of each tasks starts with selecting its setup files first. After a setup has been selected, a new window is shown in which the user is able to configure the details for the imported file.

Some of popular setup technologies (for example: NSIS, InnoSetup) are automatically recognized and in case of a positive match detailed options and command lines are extracted and

automatically inserted into respective fields.

Close the overlay window to return back to the overview showing the list of imported packages.
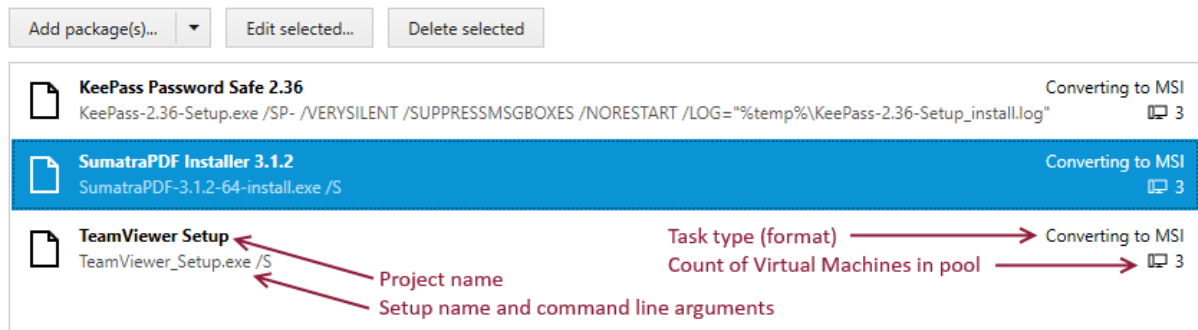
More information about editing tasks can be found in the following chapter: *Task Settings*.

Bulk Import

There are three different methods of importing packages in a bulk mode (many packages at once):

- Importing files from a whole folder including its children
- Importing a multi-selection of files from the same folder
- Importing a list from a `.csv` file

To Import a Whole Folder...

1. In the packages selection, press the small triangle in the split button **Add package(s)...** to reveal the popup menu.

2. Press the link **Import from directory...**
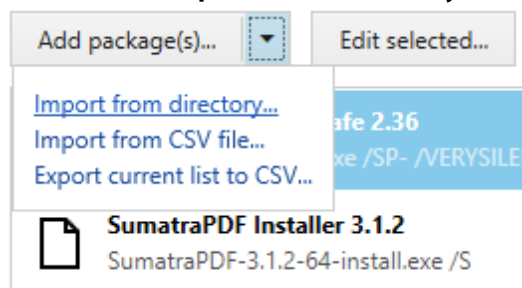
3. Select the folder to import.

4. RayPack will process the folder and show a list of the setups that have been found.
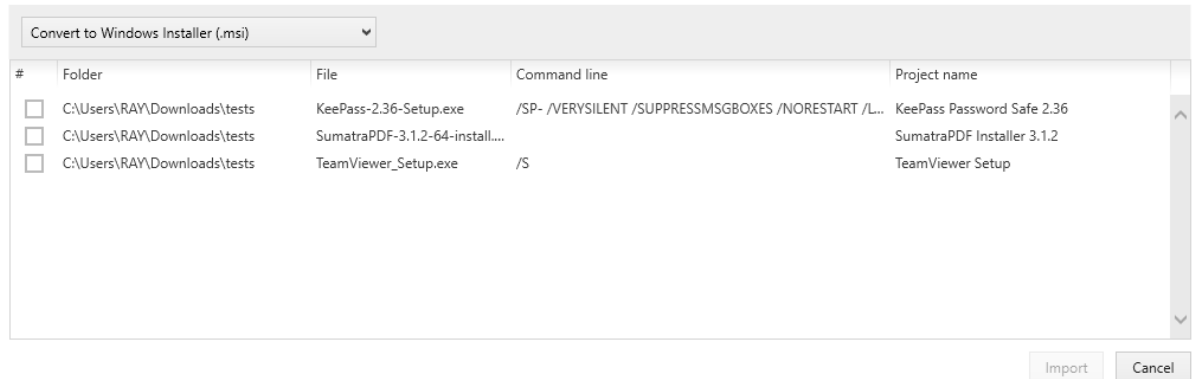
5. Use the checkboxes to select the packages to be imported.

6. It is also possible to change the command line and the project name or to customize them in later steps.

7. Use the drop-down menu to select the target format.

8. When ready, click on **IMPORT**. The packages will be imported to the current selection. It is possible to repeat the process multiple times, previous packages will not be overwritten.

To Import a Multi-Selection...

1. In the packages selection, click the **Add package(s)...** button.

2. Select multiple files from the same folder by holding the **CRTL** and **SHIFT** buttons.

3. Click on **OPEN** when finished with the selection. The packages will be imported to the current selection. It is possible to repeat the process multiple times, previous packages will not be overwritten.

To Import a List from a .CSV File...

1. In the packages selection, click the small triangle in the split button **Add package(s)...** to reveal the popup menu.

2. Click on the **Import from CSV file...** link.
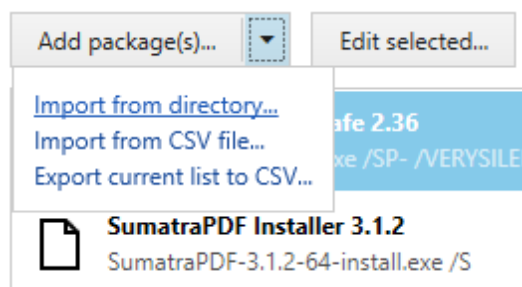
3. Select the `.csv` file.

4. Confirm the selection by clicking the **OPEN** button. The packages will be imported to the current selection. It is possible to repeat the process multiple times, previous packages will not be overwritten.

Task Settings

After adding a task, it is configured using sensible defaults. Review all options and adjust them accordingly to have the package processed, installed, configured, and converted according to the specific requirements.
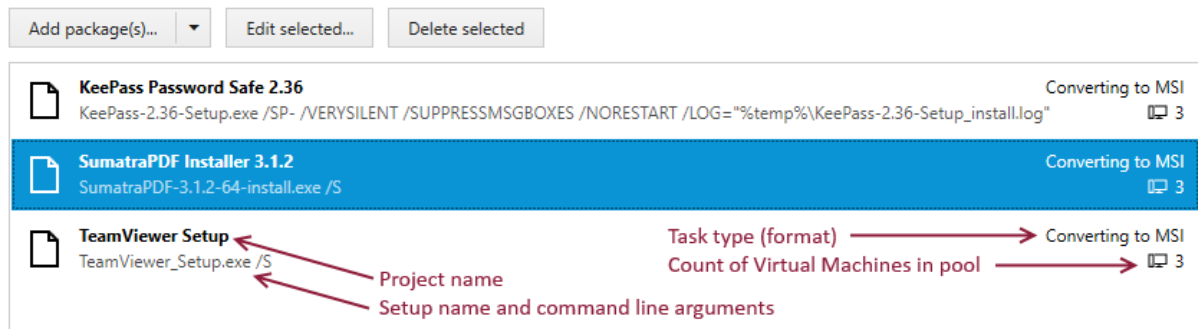
In order to edit a task, either:
- Double click its item, or
- select the item and press the **Edit selected...** button.

A new dialog is shown for each of the edited packages.

The GENERAL Tab

This tab contains the basic settings of the setup that is being processed.

- **Project name**
  The name of the project. This value serves two purposes. It is used as a display name in the overview and it is also used as the name of the folder to which the results are saved.

- **Installer file**
  This is the full path to a local installer / setup of the package. Either write this value manually or use the **...** button to browse for a setup file. When using the **...** button to select a setup, RayPack may automatically change other parameters according to the default settings and setup parameters.

- **Silent command line arguments**
  Command line arguments that are passed and executed on the Virtual Machine. If a command line that installs the product silently is provided, then the repackaging is fully automated. Otherwise, it is necessary to take care of the interactive job by observing the console and installing the product as if installing it manually.

  RayPack recognizes popular installation frameworks and may offer a silent command line by default. For example, the picture above shows a setup created with InnoSetup. When RayPack determines it, it writes the full silent command line and logging options. It is still possible to manually adjust the command line.

- **Reboot after installation**
  This option should be used for packages that require reboots (due to drivers, service installation, or any runtime / log-in first-time operations). Enabling the reboot for other packages does not bring any functional gains, but will prolong the processing because the machine has to be rebooted and has to restart. This option is available for all formats except of App-V 4.6 / 5.x using the Microsoft App-V Sequencer. There are three modes available:

  o **Never** - the machine is not going to be rebooted (even if a reboot has been requested by the setup).
  o **Always** - the machine is going to be rebooted (even if the setup does not require it).
  o **Auto** - the machine is going to be rebooted only if there is a pending reboot request

(recommended).

- **Delay second snapshot**
This single setting defines the timing for the second snapshot. A user can opt-in for a so called **delayed snapshot** in which RayPack does not perform the second snapshot automatically, but rather waits for the user to signal it. This option can be used in case there are post-installation steps which need to be carried out giving the user time and methods to manually configure the package after the main installation is done and have the changes captured by the second snapshot. Additionally, the prompt allows user to perform a reboot if required.

- **Target format**
This is a drop-down menu that contains possible output formats. The default format for a Windows Installer repackaging is MSI. Depending on the format selection, there may be more options available:

    o **Conversion to App-V 4.6 / 5.x**
    The additional checkbox **Use App-V Sequencer installed on a Virtual Machine** is shown. If this option is enabled, the sequencing to App-V will be executed on a virtual machine (which implies that the Sequencer is installed in the snapshot that is used by PackBot). Using this option also disables the checkbox **Reboot machine after installation** because this is not supported by Sequencer APIs. Uncheck this option to convert to App-V formats using RayPack conversion which supports rebooting as well.
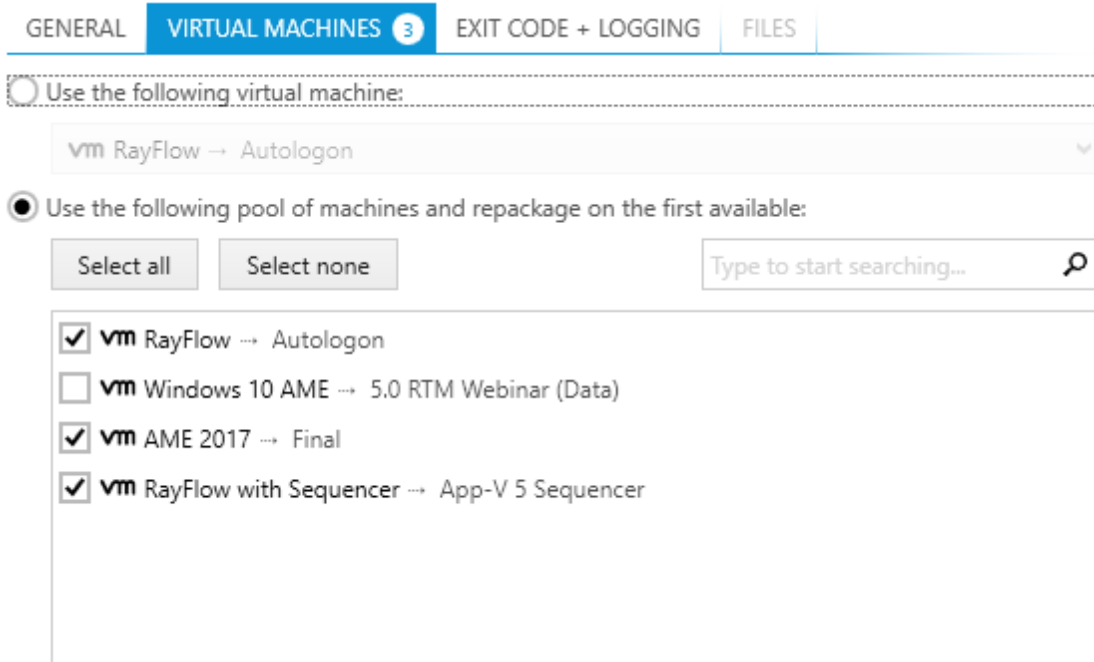
    o **Conversion to Thin-App**
    Conversion to Thin-App requires Thin-App binaries. A radio button is shown prompting the user to select which machine has Thin-App installed - it can be either the host or the virtual machine itself.

The VIRTUAL MACHINES Tab

This tab defines a pool of Virtual Machines that is used for this task.

There are two scenarios which are supported:

- **Single machine mode** (Use the following virtual machine)
  Specifies a one-to-one mapping. Only the selected machine is capable of repackaging this task.

- **Multiple machines mode** (Use the following pool of machines...)
  Specifies more than one machine capable of running the task. This scenario can be particularly effective when a single machine has been cloned many times and all cloned instances are equally matching the repackaging needs. Note that RayPack cannot guarantee which machine will be used, as this is a runtime decision made dynamically based on the workload, other packages, the virtual machines idle state, and many other factors. For more information about processing with multiple machines, refer to the chapter *Parallel Processing*.

> **Be aware:**
> At least one machine must be selected. Failing to do so prevents the user from moving to the next step of the parent wizard.
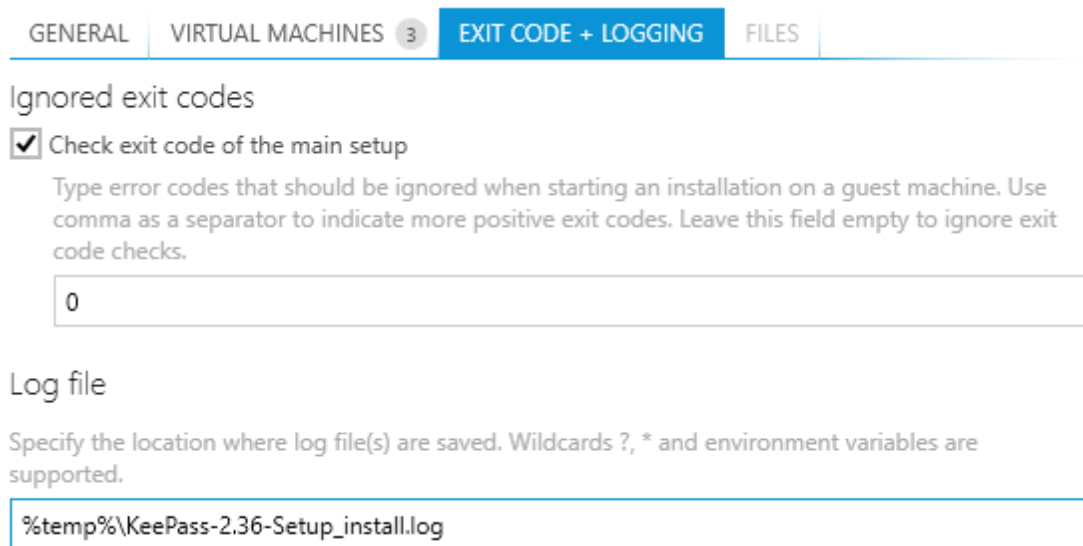
Automatic Selection of Virtual Machines

When the user is changing the target format, the current selection of virtual machines may change. There are special settings in the RayPack profile which define which machine types are to be used for which default types of tasks. For example, the user may define that all App-V related tasks are to run on a specific machine having the sequencer installed. For more information, refer to the chapter *PackBot*.

The automatic selection of a virtual machine works until the user manually overrides this selection.

The EXIT CODE + LOGGING Tab

This tab defines how the exit code is handled and where the log files are generated during the installation.

- **Check exit code of the main setup**
  This defines whether exit codes should be checked when the process is finished or not. When the setting is active, it can also be defined which exit codes are indicating success. By default, only **0** is handled for generic setups and a bunch of reboot-related exit codes for MSI files are recognized as well. Separate valid exit codes using commas.

  Be aware:
  If this checkbox is active and the setup fails to return the expected exit code, the installation will end prematurely and PackBot will not generate a package out of a setup that is considered to be **failed**.

- **Log file**
  This setting can provide a great value in case of bulk repackaging by specifying the location of the log files that are related to the setup. It is possible to use environment variables and wildcards to configure path(s) for the log files. After the installation, files matching this pattern are copied back to the host machine regardless of whether the setup succeeded or failed. This on the other hand can provide valuable clues for failed setups to a user that is using bulk

repackaging.

Some types of setups (InnoSetup, Windows Installer) provide command line configuration for log files. Once RayPack determines that a setup provides the logging functionality out of the box, it will adjust the silent command line parameters and the log file name pattern accordingly. For other cases, a manual configuration of the pattern may be needed.

Once the package is done (either successful or as a failure) and the log files are available, a link will be shown in the overview. The log files are always copied to the folder where the repackaged output is stored on the host machine.

> **Be aware:**
> If no log file that is matching the specified pattern is found, RayPack will continue without any error.

The FILES Tab

This tab defines any additional files that belong to the setup and must be copied to the virtual machine.
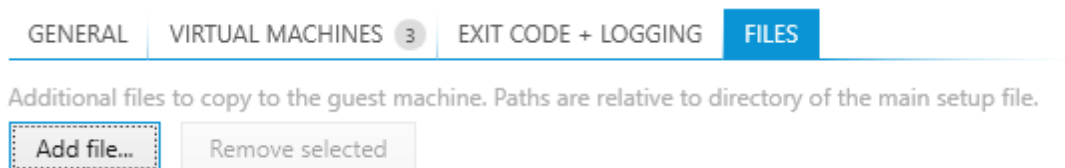
This setting should be used when the main setup has additional files that are required. A typical example are `.cab` files that are required to install an `.msi` package or `.ini` files with the configuration for `.exe` setups, etc.

RayPack automatically identifies supporting files for recognized setups. In case some additional files are required, those can be added by providing a relative path (where the base folder is the folder in which the main setup resides). For example, when repackaging `C:\temp\setup.msi`, in order to attach `C:\temp\files\transform.mst` simply use `files\transform.mst` as a relative file path name.

Options

The **Options** page is the final step before starting PackBot.

This page of the wizard allows users to change two minor settings:

Output Folder

This is the root folder where all the repackaged / virtualized projects are saved.

> **Note:**
> A subfolder is created for each task. The name of the subfolder is the same as the name of the project that has been configured in the previous step of the wizard.

Parallelism Level

This number indicates how many tasks can run at once. By default, this is set to the number of unique machines that are defined in the pools. Setting this number to a higher value speeds up the packaging of many packages at once, provided that the machine is capable of running that many different virtual machines. On the other hand, when running on a slower hardware, the number may be decreased to get an overall better performance. Modern computers should be able to run up to two or three VMware Workstation machines without much performance penalty. ESX and Hyper-V servers are usually powerful enough to start much more machines at once and when working solely on them, the number can be safely increased as well.

Advanced Scanning and Exclusion Options

These options are equivalent to the PackRecorder settings, described in the following chapter: Coverage + Exclusions.

Processing and Overview

When the packages are processed, a simple overview is shown with the current progress of the packages being processed at the moment and the queue.
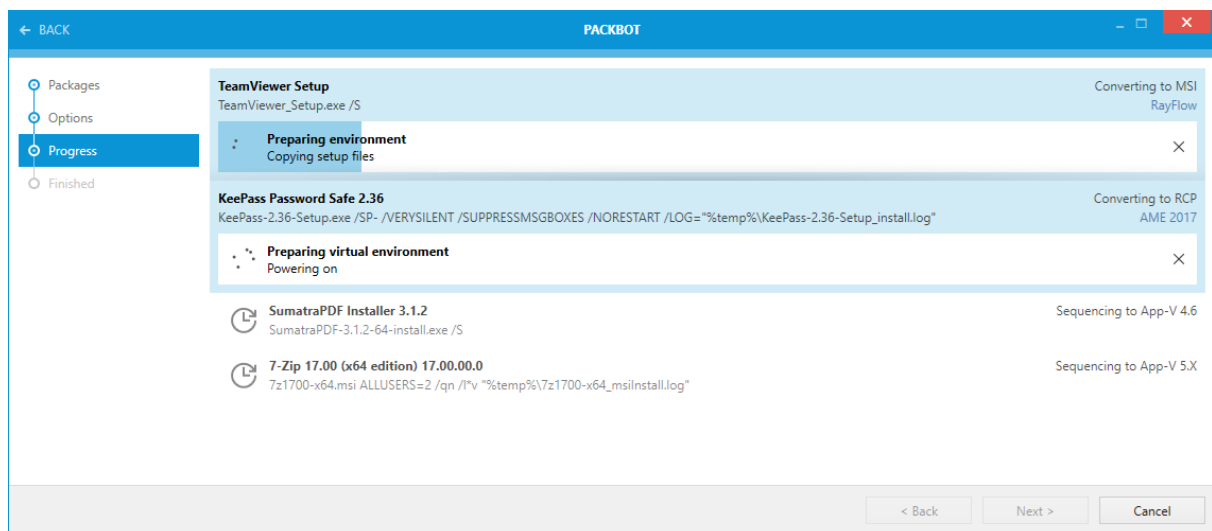
The view contains the following information:

- **Currently processed packages**
  The packages that are currently processed are shown with a blue background and a progress bar, which is informing about the current activity. For example, the picture above shows that TeamViewer files are currently being copied to the Virtual Machine.

- **Name and command line**
  The name of the project and the command line shown underneath are most prominent information in this view.

- **Supporting files** (not shown in the picture above)
  If a package has one or more supporting files, the counter is shown underneath the command line of the project.

- **Task types (formats)**
  On the right side, a small text caption informs about the type of a task (for example *Converting to MSI, Sequencing to App-V 5.x*, etc.).

- **Name of the virtual machine**
  Active tasks have the name of the virtual machine that is processing them. The name can be used to identify which machine has been allocated by PackBot and to check its console if a user interaction is to be expected. Additionally, for Hyper-V machines, clicking the name opens the RDP console allowing to connect to the virtual machine.

- **Package queue**
  Packages that are pending are shown below. Their items show information similar to currently process package, with the exception of the name of a virtual machine. This is caused by the fact that the processing is non-deterministic and PackBot allocates the machines dynamically just before starting the task.

Active tasks can be canceled by pressing the **X** button in their progress bar. Canceling may take a few seconds depending on the progress, eventually the machine is returned back to the pool and ready to pick-up the next task from the list. In order to cancel all pending tasks, press the **Cancel** button. Finished tasks cannot be canceled and pending tasks cannot be canceled individually.

Interaction with the Repackaging Process

If one of the following is true:

- The second snapshot has been delayed, or
- the product cannot be installed silently, either by deliberately leaving a non-complete command line or implicitly by prompts shown by an otherwise silent install,

then it is required to interact with the installation. In most cases, this would be simply progressing the wizard and doing an extra post-installation job, but there are two special considerations which come from PackBot:

- **Continuing with delayed snapshot**

If the second snapshot is delayed, the user is expected to press a button when ready. The window is shown in the virtual machine and may look like the following:
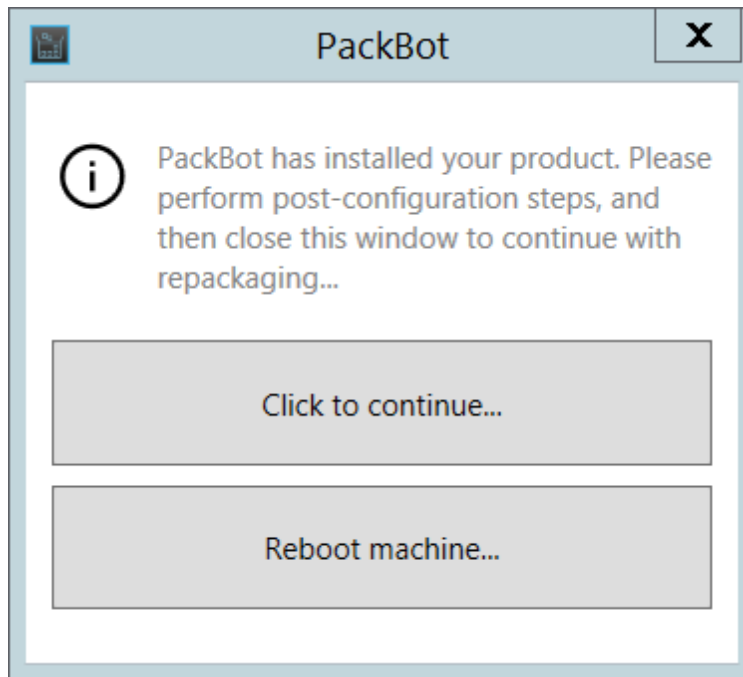
- **Rebooting the machine**
  If the second snapshot is delayed, the same window can be used to trigger a reboot of a virtual machine. The machine can be rebooted as many times as required.

Exit / Finish Conditions

Once all packages are finished, the **Summary** page is shown automatically. The result depends on the individual status of the repackaged tasks:

- If at least one task has been canceled, the overall status is also **Canceled**.
- If at least one task has failed, the overall status is **Failed**.
- If no task has been canceled or has failed, the overall status is **Success**.

Successfully Repackaged Apps

PackBot displays a basic info for each package that has finished successfully.

From this view, users may click on the task type (**Converted to MSI**) to jump directly to the output folder. When a log file is available, a button **Show log** is present and pressing it opens the log file created by the setup.

Troubleshooting Failed Packages

If a package fails, a view similar to the following one may be shown:

If there is a log file available, it is possible to click on the error message. In this example, clicking the link **Process returned unsupported exit code 1603** opens an `.msi` log file that may be used to analyze the problem with the setup.

Parallel Processing

If more packages are converted at once, depending on the required scenario, a parallelism is in use to make sure that the total time needed to finish all tasks is as low as possible.

Bulk Processing Using a Single Virtual Machine

If a single virtual machine is used or multiple virtual machines scoped to the same physical VM files (for example two different machines configurations using the same VMX file but different snapshots), the following logic is applied:

- The machines are running sequentially, this means that the repackaging of a task starts only when the virtual machine is not running.
- The processing of each task starts with restoring the required snapshot and powering on the machine.
- After repackaging, the output is sent to the host and processed there. This step is executed only if conversion to MSI, RPP, App-V (without Sequencer), or Thin-App (with local VM) is selected.
- During post-processing the virtual machine is not needed anymore and is shut down (the machine can optionally be left running if there are no more tasks for this machine and PackBot has been configured to do so).
- Post-processing tasks never run simultaneously and are automatically queued.
- The tasks are executed in the very order defined in the Wizard.
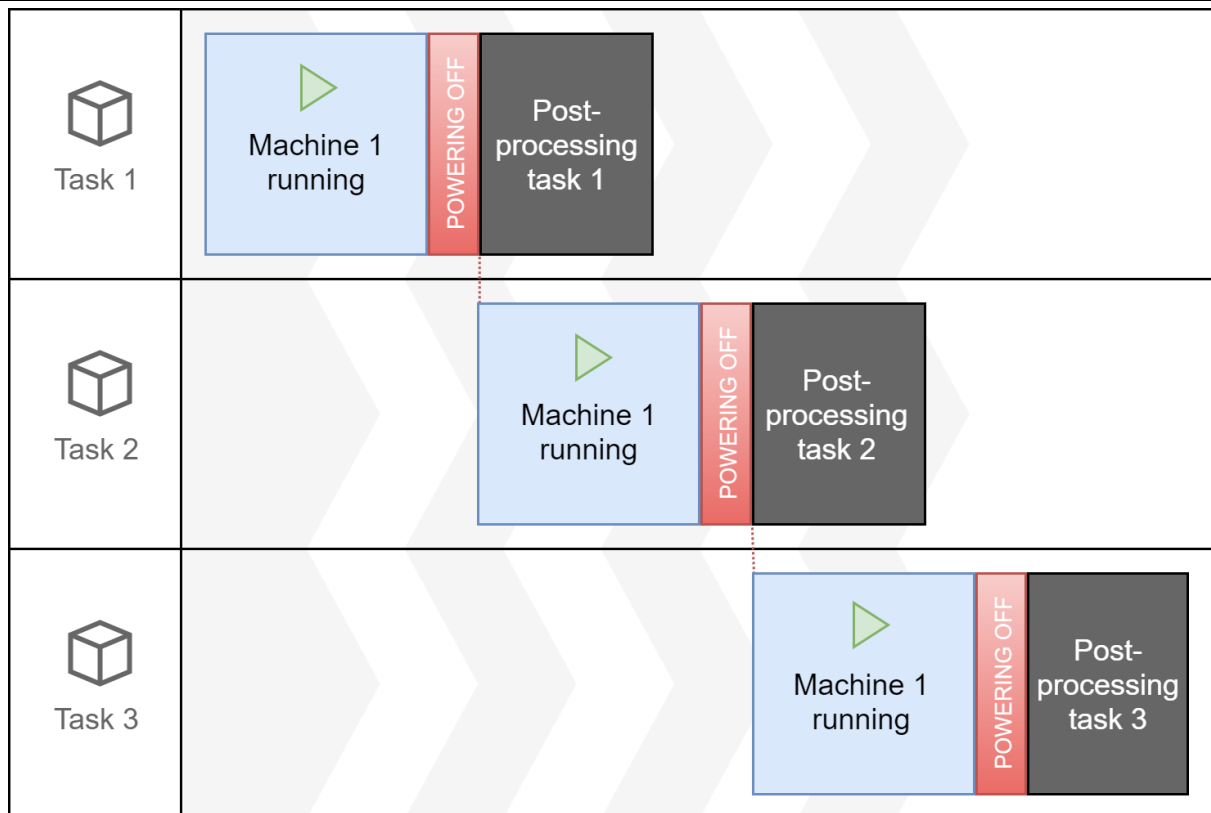
Because post-processing is detached from the virtual machine, the next task starts even if the previous task is not physically ready (the only requirement is that its virtual machine is not needed anymore). This strategy saves time which may save a few minutes in the typical repackaging scenarios.

Bulk processing using multiple virtual machines

If multiple virtual machine are used and they can be started parallel (which excludes machine configurations which are in fact based on the same VMX file), the following logic is applied:

- The machines are running in parallel, but to a maximum degree specified during the Wizard.
- RayPack determines the best virtual machine candidate for a given task. To find the candidate it uses the following information:
  - o Which machines are available
  - o Which machines are assigned to which packages
  - o Additional logic that minimizes waiting time is also applied. This means that it is possible that the tasks will not be processed in the order defined by the user, if RayPack determines that it may be faster to process tasks in different order.
- The processing of each task starts with restoring of required snapshot and powering on the machine
- After repackaging, the output is sent to the host and processed there. This step is executed only if conversion to MSI, RPP, App-V (without Sequencer), Thin-App (with local VM) is selected.
- During post-processing, the virtual machine is not needed anymore and is shut down (the

machine can be optionally left running if there are no more tasks for this machine and PackBot has been configured so).

- Post-processing tasks never run simultaneously and are automatically queued. For example, the following flow shows that post-processing of task 2 is not executed directly after powering off its virtual machine, because at this point of time RayPack is busy with post-processing of the first task. Similarly, post-processing of task 3 is not attached to the virtual machine, as it needs to wait for both task 1 and task 2 to be ready with their post-processing routines.
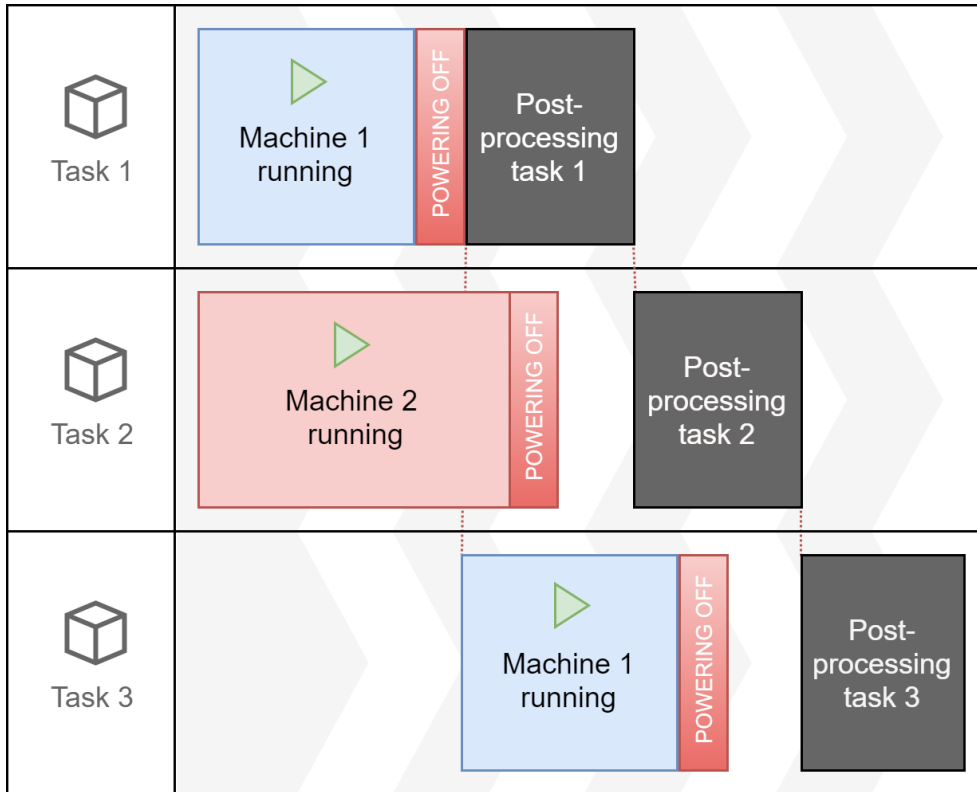
Because of that, if the host computer is capable of running 2 machines at once with 3 packages, the total time required for the whole processing can be as low as 50% of the original time requirement if only one virtual machine would be used.
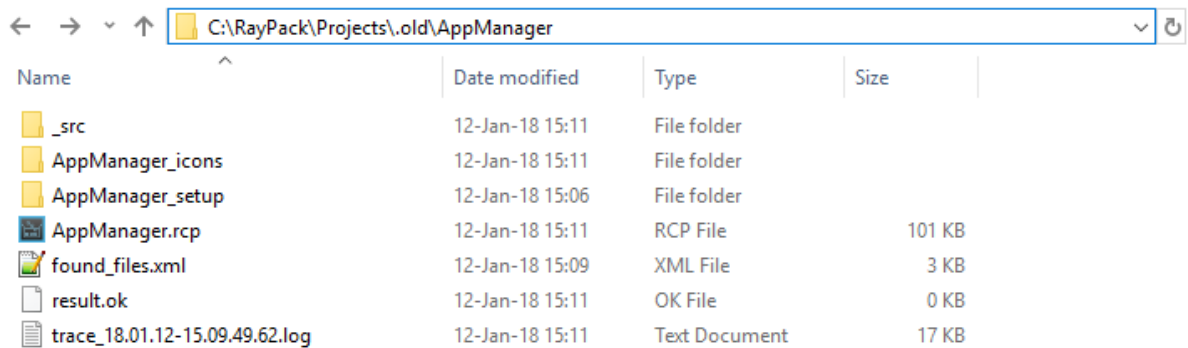
Detection of Vendor MSI Setups

PackBot automatically detects vendor MSI installations. Many setups are prepackaged to a single container `.exe` file which acts merely as a wrapper and executes the actual Windows Installer database extracted to a temporary location. If this happens, repackaging to the MSI format is usually a bad choice, because it breaks the compatibility and upgrade possibilities of older vendor packages. In most cases, the recommended approach is to create a Windows Installer transform.

PackBot solves this problems in two ways which are automatically working under-the-hood without user intervention if a conversion to RCP or MSI format is requested.

All Conversions (excluding App-V 4.6/5.x using Sequencer)

For all conversions (but not the one requiring the Sequencer to run) RayPack monitors the background installations and detects Windows Installer sessions. Should an MSI setup be started during the repackaging (often implicitly, for example started by an executable wrapper) RayPack recognizes the installation and it sources and once the repackaging is over, these essentials files are copied and stored next to the actual project. For example, a content similar to the following is created when repackaging to the RCP format:

Aside of standard folders, the **<ProjectName>_setup** contains all original vendor MSI installations and their log files that were captured during repackaging.

> **Note:**
> If converting to RCP format, it is possible to review vendor installations using the **General** > **Original setups** screen.

Conversion to MSI

If a project gets converted to the MSI format, then an extra logic is applied if there is at least one Windows Installer running as a part of the installation. The actual action done by PackBot depends on the Vendor MSI settings. There are two main actions that can be done with an original vendor MSI:
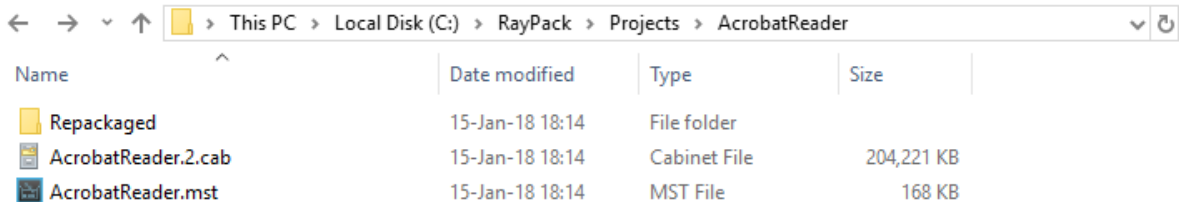
1. Disregard the original MSI and create a new repackaged MSI.
2. Create a transform that modifies the vendor MSI.

Both have slightly different use cases.

- The original MSI should be disregarded and a fresh copy should be created if the setup that is repackaged is a so-called "wrapper MSI" which does not do anything on its own, but rather serves as a wrapper / host. If this is the case, creating a repackaged MSI is usually a better option, although it is somehow contradictory to Best Practises which say that the repackaging of Installer databases should be avoided.
- An MST should be created if breaking internal structures and upgrade possibilities should be

avoided. By only adding changed content via a relatively small MST file and supporting files those are kept untouched.

Because the use cases are different and cannot be always predicted before starting a setup, RayPack by default combines them and produces both options. For example, this is the result of repackaging Adobe Reader which is distributed as an executable setup, but is being installed using a "hidden" MSI:

The folder **Repackaged** contains a fresh new copy of repackaged MSI, and in the original folder there is a transform and CAB files containing additional source files. If a non-default setting was selected to convert to either only MSI or MST, then the files are stored directly in the project folder without any additional subfolders.

Of course, in this case taking **MST** as the result is the right choice - this way the upgrade logic of a product that is actually being installed as a native Windows Installer setup can be preserved.

Best Practises and Recommendations

Recommendations for Virtual Machines Configurations

The following recommendations are optional, but failing to apply them may result in a limited functionality of certain features (especially silent / bulk repackaging):

1. **Disable User Account Control.**
   When UAC is enabled, some tools may not work correctly due to lack of permissions, and the packages may not be able to run silently (user interaction is required).

2. **Enable Auto-logon.**
   When auto-logon is enabled, the virtual machine boots the desktop directly without prompting a sign-in and without showing any Lock Screen. This is also important for bulk / silent mode, so that the user does not have to retype his password every time.

3. **Review the Firewall Settings.**
   PackBot uses TCP / IP communication between guest machine and the host machine. Make sure that used ports are not blocked. By default, PackBot uses the port range `48654-48999` starting from lowest port numbers. The general rule is that your host machine must be visible from your VM guest so that PackBot can send the data back. If you are unable to ping your host machine from guest virtual system using host DNS name, then PackBot would also fail being unable to setup up a connection between two machines.

General Recommendations

- To get the fastest conversion possible, select RCP as the target format. By doing so, no post-processing is required and the queue of packages can be processed more efficiently. It is always possible to build the RCP to any format by either using a command line or the PackRecorder UI.
- Do not use reboots for packages that do not require them. A reboot can incur additional overhead, trigger Windows Update etc., and it also takes time to restart and boot the machine again.

PackWrapper

PackBot enables IT professionals to wrap existing setup files (in Windows Installer or non-Windows Installer formats) using a PowerShell-based wrapper.
Using a wrapper over specific setups has the following advantages:

- Unified single installation methods (implementation of command line and installation routine is transparent to the caller / deployment system)
- Silent, semi-silent (basic UI), or interactive installations and uninstallations
- Ability to include extra steps, both before and after main installation / uninstallation using PowerShell functions

The installation can run in a fully unattended mode or with a basic UI. The system tray can also be used to notify about the installation (enabled by default):

RayPack is using the widespread PowerShell AppDeploymentToolkit. The framework is open source and also free for commercial use. According to its website:

"The PowerShell App Deployment Toolkit provides a set of functions to perform common application deployment tasks and to interact with the user during a deployment. It simplifies the complex scripting challenges of deploying applications in the enterprise, provides a consistent deployment experience and improves installation success rates."

It is possible to download the full source code from the website of the publisher:
https://github.com/PSAppDeployToolkit/PSAppDeployToolkit

More information about the toolkit:
http://psappdeploytoolkit.com

Creating PowerShell (PSADT) and Intune packages

In Order to Create a Wrapper

1. In main menu, press **FILE** > **NEW** to show the **New Project** backstage menu.

2. Click on **Wrapper** to start a new **PackWrapper** wizard.
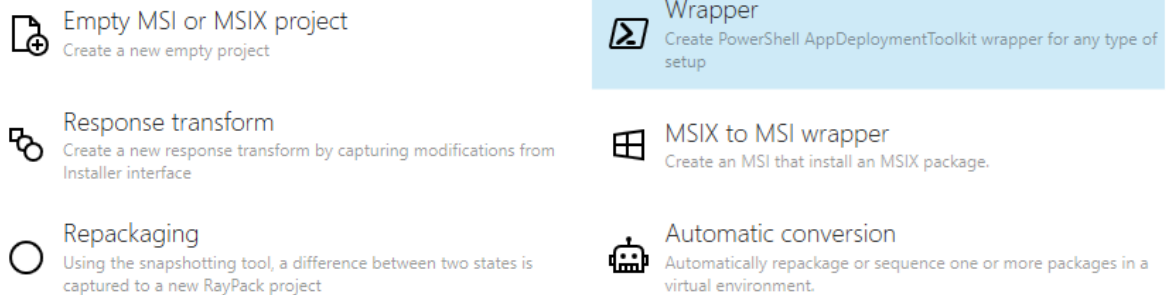
3. Follow the steps described in next sections.

Selecting Setup File

In this step, the main installation entry point has to be selected. Depending on the vendor, the package may be in a Windows Installer format (`.msi`), MSIX package (`.msix`) or any other, most often executable format (`.exe`). Select the full file path to the setup by pressing the **Browse** [**...**] button. After a setup is selected, its details are presented underneath.

> **Note:**
> These values are read from the setup file that has been provided. For some executable installers, their metadata may be out-of-date or irrelevant. In this case, fix them by entering the correct values before continuing to the next page.

If MSI file is selected as the original installer, a selector for additional transform files will be shown underneath:

Transform files defined at this steps will be executed by the wrapper and copied to the source files folder, together with the main `.msi` file.

Once ready, press **NEXT >** to go to the configuration of the selected setup file.

Automatic Recognition of Command-Line Switches

Once an executable or an MSI is selected as a setup type, RayPack tries to analyze it and create a command line that can be used to install the product silently. This functionality supports popular setup and frameworks like NSIS, InnoSetup and many more. The recognition is executed automatically, no further input from the user is required. For example, if a selected setup has been authored in InnoSetup. the following will be generated:
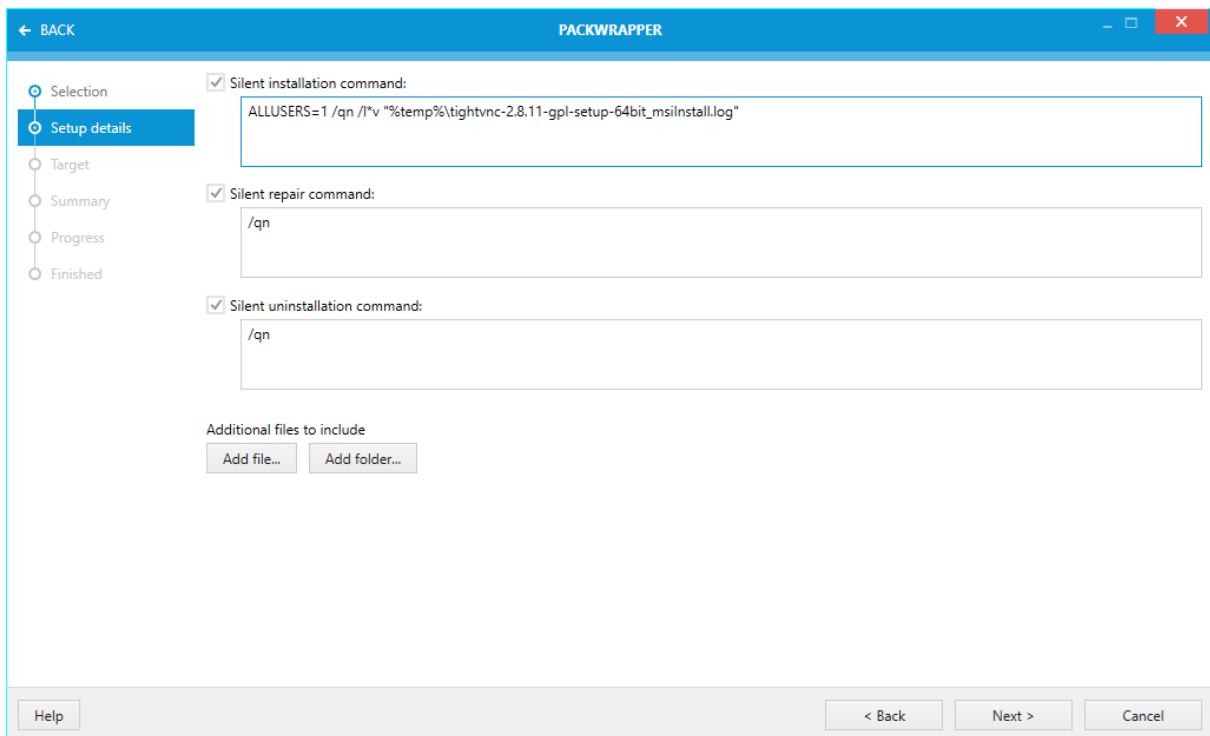
Configuring the Setup

In this step, the setup is to be configured to be executed silently according to the specific requirements.

There may be small differences regarding the available options.

- For Windows Installer setups, both the installation, repair and the uninstallation command are shown.
- For non-MSI setups the user is able to manually write the repair and uninstallation command, either by calling the original setup with extra arguments, or by starting a completely different command (for example `%programfiles%\myapp\unins.exe`). For the latter, it is possible to use environment variables and PowerShell syntax (for example `$env:ProgramFiles`).

If the setup requires additional files (including but not limited to `.cab`, `.inf`, `.ini`, and other files) these can also be added by pressing the **Add file...** button. The list may already be prepopulated with values if RayPack detects that some files must be automatically included.

Once the configuration is ready, press **NEXT >** to go to the folder selection.

Target Selection

In this step, the target folder has to be configured.
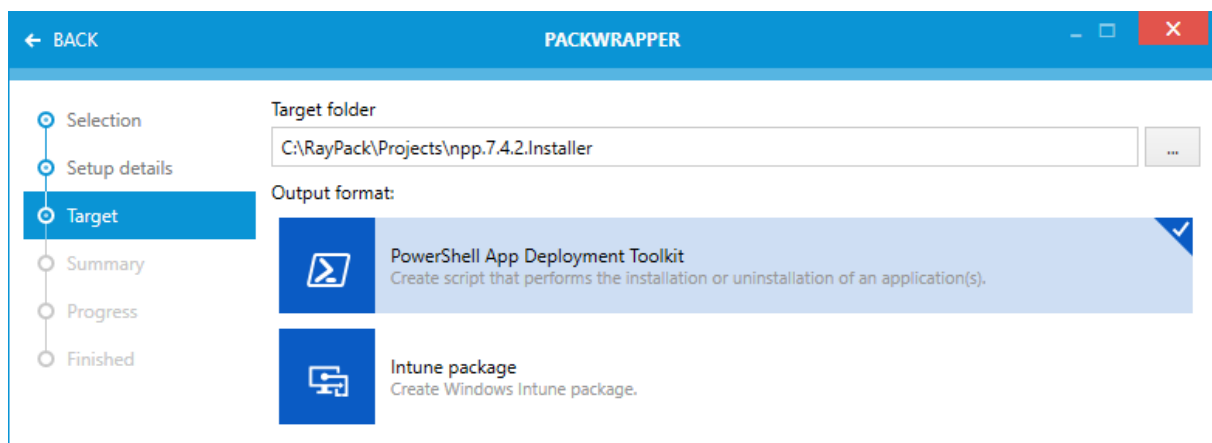
By default, the target folder is set as a combination of the main project folder and the file name without extension of the selected setup. The path can be changed by pressing the **Browse** [**...**] button or by entering the desired path into the textbox.

The **Output format** lets you decide about the type of the deployment:

- **PowerShell AppDeploymentToolkit**
  Creates a classic PSADT deployment with all necessary source and toolkit files.

- **Intune package**
  Creates an Intune (`.intunewin`) self-contained package.

> 📎 **Note:**
> Some Intune functions may require extra dependencies to be present on the local machine. You may be prompted to download them, which RayPack does automatically. Without these dependencies however the process will be aborted.

Once the folder is configured, press **Next >** to go to the **Summary** screen and confirm the settings by pressing the **Process >** button.

Processing

As a part of the processing, RayPack does the following:

1. The selected setup file and supporting files are copied to the target folder to the `/Files` subfolder. Folder relations between files are preserved.
2. The PowerShell AppDeploymentToolkit bundle is copied to the target folder to the `/AppDeploymentToolkit` subfolder.
3. The package configuration and miscellaneous files are copied to the target folder.

All files are required for the package to work correctly.

Using the Wrapper

Deployment

Once a setup is wrapped, copy all generated files to the chosen deployment tool. Callouts for installation and uninstallation are standardized for all products, regardless of their underlying installation technique.

Unattended Installation

To install wrapped applications silently, use the following command line:

```
Deploy-Application.exe Deploy-Application.ps1 -DeploymentType Install -DeployMode Sile
```

Unattended Uninstallation

To uninstall wrapped applications silently, use the following command line:

```
Deploy-Application.exe Deploy-Application.ps1 -DeploymentType Uninstall -DeployMode S:
```

Default Mode (Basic UI)

It is also possible to run `Deploy-Application.exe` directly. This will by default install the product with a basic GUI that informs the user about the progress:
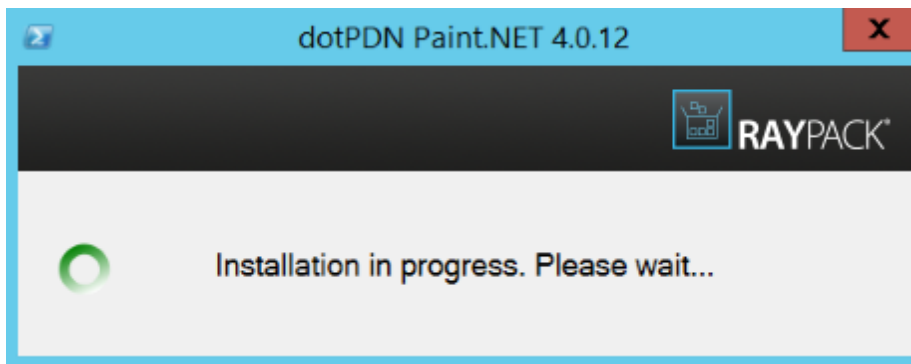
Customizing the Toolkit

In order to customize install and / or uinstall routines, open `Deploy-Application.ps1` with a text editor (preferably PowerShell editor or any editor with syntax highlighting).
Relevant parts are located around line `152`, and they may look like the following:

```
If ($deploymentType -ine 'Uninstall') {
    ##*===============================================
    ##* PRE-INSTALLATION
    ##*===============================================
    [string]$installPhase = 'Pre-Installation'

    ## Show Welcome Message
    Show-InstallationWelcome -CheckDiskSpace -PersistPrompt

    ## Show Progress Message
    Show-InstallationProgress

    ## <Perform Pre-Installation tasks here>

    ##*===============================================
    ##* INSTALLATION
    ##*===============================================
    [string]$installPhase = 'Installation'

    <YOUR INSTALL SCRIPT>

    ##*===============================================
    ##* POST-INSTALLATION
    ##*===============================================
    [string]$installPhase = 'Post-Installation'

    ## <Perform Post-Installation tasks here>

    ## Display a message at the end of the install
    Show-InstallationPrompt -Message "$appName $appVersion has been
successfully installed." -ButtonRightText 'Close' -Icon Information -
NoWait
```

```
        }
ElseIf ($deploymentType -ieq 'Uninstall')
{
        ##*===============================================
        ##* PRE-UNINSTALLATION
        ##*===============================================
        [string]$installPhase = 'Pre-Uninstallation'

        ## Show Welcome Message
        Show-InstallationWelcome

        ## Show Progress Message
        Show-InstallationProgress

        ## <Perform Pre-Uninstallation tasks here>


        ##*===============================================
        ##* UNINSTALLATION
        ##*===============================================
        [string]$installPhase = 'Uninstallation'

        <YOUR UNINSTALL SCRIPT>


        ##*===============================================
        ##* POST-UNINSTALLATION
        ##*===============================================
        [string]$installPhase = 'Post-Uninstallation'

        ## <Perform Post-Uninstallation tasks here>
}
```

Standard PowerShell functionalities and modules are supported. It is also possible to use the dedicated PowerShell AppDeploymentToolkit module. The module itself is included by default, so there is no need to do any extra action other than calling the required methods. A comprehensive documentation of available methods and examples can be found under the following address:

http://psappdeploytoolkit.com

PackPoint

PackPoint is a feature, which is available for professional and enterprise edition users, allowing them to reuse a central repository of common resources (templates, merge modules, exclusions, predefined properties, etc.). Once established, PackPoint is an essential companion for the maintenance of high level teamwork and standardization measures in packaging factories.

In order to use the PackPoint organization structure, the storage location for the PackPoint has to be defined during the RayPack setup. If a PackPoint is defined, the initial population with resources is executed when RayPack is launched for the first time. All core resources, which are appropriate for permanent maintenance beyond product upgrades and local instance endurance, are automatically put to the PackPoint location. The material preserved within the PackPoint location is the base for any synchronization with the local resources on the packaging machine.

> **Note:**
> PackPoint is not available for Standalone Repackager installations of RayPack. Please use either the custom or typical setup type to enable PackPoint initialization during the application installation.

If the currently running application instance uses a PackPoint, the path to the storage location can be reviewed from the Troubleshooting view of the About area. However, it is not possible to change the PackPoint location for an existing RayPack instance by using the controls of the application interface. However, it is possible to influence the operational mode of PackPoint. Please refer to the settings section for further details.

> **Warning**
> Even though it is not recommended to do so, users may modify the PackPoint configuration by manually adjusting the content of the `PackPoint.config` file, which is stored within the RayPack installation directory (which is typically leading to a config file path like `C:\Program Files (x86)\RayPack\PackPoint.config`)
>
> Please take into account, that manual modifications may have severe impact on the automated synchronization features of the PackPoint repository. If modifications are needed, make sure to have a copy of the original configuration file at hand if the changes turn out to cause negative side effects or to be erroneous in general.

Here is an overview of resources that are handed over to PackPoint control if a central repository has been setup for a RayPack packaging environment:

Resource location without	Resource location in PackPoint
`[InstDir]\Resources\ICE.xml`	`[PPDir]\ICE.xml`
`[InstDir]\Resources\MergeModules`	`[PPDir]\MergeModules`
`[InstDir]\Resources\PredefinedSearchs`	`[PPDir]\PredefinedSearchs`
`[InstDir]\Resources\Property`	`[PPDir]\Property`
`[InstDir]\Resources\Validation`	`[PPDir]\Validation`
`[InstDir]\ConfigurationTemplates\`	`[PPDir]\Default`
`[InstDir]\ConfigurationTemplates\Filters`	`[PPDir]\Default\Filters`
`[InstDir]\ConfigurationTemplates\PackageTemplates`	`[PPDir]\Default\PackageTemplates`
`[InstDir]\ConfigurationTemplates\Profiles`	`[PPDir]\Default\Profiles`

Note:

`[InstDir]` is the RayPack installation directory, which is typically something like `C:\Program Files (x86)\RayPack\`

`[PPDir]` is the PackPoint location directory, typically something like `C:\RayPack\PackPoint\` for a local PackPoint instance as permitted by professional edition licenses.

Both directory locations are freely configurable during the RayPack installation, therefore the actual locations on your packaging machine may vary.

Types of PackPoint resources

Essentially, two types of PackPoint resources can be distinguished:

- **Resources for which a local copy is created**
  Examples of this type of resources are profiles, exclusions and templates. In order to ensure that a user has permissions to change them, the global PackPoint profiles and templates are copied to the local machine and will be used instead. If there is no access to PackPoint, defaults will be taken from `[INSTALLDIR]Resources\Default`. These resources are critical and required for RayPack to work.

- **Resources accessed directly from PackPoint**
  Most of the customizable resources (Custom Actions, predefined folders, system search, conditions, Merge Modules etc.) are accessed directly from PackPoint. If PackPoint becomes unavailable, certain features may not work (for example, the Merge Modules browser may show no predefined modules etc.). These resources are not critical and RayPack will run even if they are not accessible.

PackPoint update mechanism

Whenever RayPack is launched, the version value of the local resources and the PackPoint

version are compared. If they differ (which may occur when the PackPoint has been updated during maintenance or product update phases), one of the following procedures is executed:

- **Strict mode (PackPoint overrides local settings)**
  Any change that has been executed on the PackPoint location will be taken over into the local settings. Any local changes will be lost. This is the default behavior.

  During this process:

  o Existing local RayPack configuration ("`%appdata%\RayPack\*.*`") is being updated from the PackPoint configuration templates (`[PPDir]\Default`).
  o Local filters and profiles are overridden
  o Package Templates are replaced completely (from PackPoint to local).

- **Loose mode (Local settings are preserved)**
  The local resources and PackPoint resources are merged to generate a synchronized overall state.

  During this process:

  o Existing local RayPack configuration ("`%appdata%\RayPack\*.*`") is being updated from the PackPoint configuration templates (`[PPDir]\Default`).
  o Filters and profiles are merged (only not existing entries are added, exiting entries are remaining the same).
  o Local package templates are preserved

  This synchronization allows to take over local adjustments into the overall PackPoint, whilst at the same time being able to receive updated PackPoint resources into the settings for the local application instance. In order to reset the changes performed to local settings, users should remove them from the local machine, and relaunch RayPack. The synchronization mechanism will automatically create a new local copy of the resources preserved within the PackPoint.

Be aware:

If the defined PackPoint location is unavailable due to broken files, missing user rights, or missing connectivity, RayPack is still operational. It will use the local settings. If both, local settings and PackPoint resources are unavailable, RayPack will operate in a failsave mode, using default resources for vital settings information.

Nonetheless, it is highly recommended to repair the PackPoint and the local settings, in order to avoid data loss and working result quality that is of less quality than possible with the settings defined for your individual packaging environment.

Please use the settings area to define the PackPoint mode of the currently active RayPack product instance. If you encounter any issues with the setup or maintenance of PackPoint, please contact our support team - we will gladly assist.

PackPoint versioning

The following files are used to determine whether the resources must be updated:

- `%AppData%\RayPack\PackPointVersion.txt`
  Contains a single value being the local version number of PackPoint resources

- `[PPDir]\Default\PackPointVersion.txt`
  Contains a single value being the global version number of PackPoint resources. `[PPDir]` denotes the location to which PackPoint has been installed.

Any time PackPoint is updated, the versions are updated respectively. To force an update of local resources from PackPoint, simply increase the value in global `PackPointVersion.txt` file.

Manual PackPoint updates and migration

In order to force update of PackPoint, use command line tool r pcmd. exe. The following sections contains parameter description and syntax:
PackPoint migration

Building App-V Packages

RayPack 5.0 supports automatic conversion from MSI / RPP / MST files and repackaging from legacy apps for the following App-V formats:

- App-V 4.6
- App-V 5.0 (+SP2, SP2 Hotfix4 and SP3)
- App-V 5.1
- App-V for Windows 10 (versions 1607 and 1703)

The following table summaries most important functional differences between different versions that RayPack supports:

	App-V 4.6	App-V 5.X up to 5.0 SP2	App-V 5.0 SP2 and newer	App-V for Windows 10
Shortcuts	✅	✅	✅	✅
Files	✅	✅	✅	✅
Registries	✅	✅	✅	✅
Extensions	❌ not supported	✅	✅	✅
ProgID	❌ not supported	✅	✅	✅
Verb	❌ not supported	✅	✅	✅
MIME	❌ not supported	✅	✅	✅
EnvironmentVariables	✅	✅	✅	✅
Shell Extensions	❌ not supported	❌ not supported	✅	✅
COM	❌ not supported	✅	✅	✅
Services	❌ not supported	✅	✅	✅

	App-V 4.6	App-V 5.X up to 5.0 SP2	App-V 5.0 SP2 and newer	App-V for Windows 10
Fonts	❌ not supported	✅	✅	✅
Application Paths	❌ not supported	✅	✅	✅
URL Protocol handlers	❌ not supported	✅	✅	✅

In order to get an App-V 4.6 / 5.X package, the following two scenarios are supported.

For simple applications already available in MSI format
Simple applications relying on native features and not on **Custom Actions** projects may be converted directly to App-V format. This method is fast and reliable and does not require usage of virtual machines. For more information about App-V conversion, refer to the *Building packages* chapter.

For complex applications already available in MSI format and legacy applications
Legacy projects must also be repackaged first. Complex MSI applications that rely on custom actions may not always be fully working after the conversion process. In this case, perform repackaging first to make sure the installation logic is expanded into simple entities (files, registries etc.) using the Repackaging Wizard or PackBot. After that, convert the repackaged output into the selected virtual format.

Configuring App-V Conversion

The global configuration for App-V 4.6 and 5.X conversion is available within the **Settings** > **Conversion** screen.

Options for App-V 5.x

| APP-V 5.X | APP-V 4.6 | THINAPP | MSIX + UWP |

Launcher

☑ Create MSI wrapper

☑ Copy App-V launcher to the output folder

   The launcher can be used to quickly test and troubleshoot the produced App-V packages.

Version

Select the target package version used for the converting process:

| 5.0 | ⌄ |

Target operating systems:

All supported systems ✎

Global options

☑ Enable static dependencies

☑ Treat INSTALLDIR as Primary Virtual Application Directory (PVAD)

☐ Allow virtual applications full write permissions to the virtual system ⚠

COM options

☑ Allow all named objects to interact with local system

☑ Enable COM in-process

☐ Enable COM out-of-process

COM Mode:

| Isolated | ⌄ |

Launcher Options

Create MSI Wrapper
If this option is selected, each time when an App-V application is built, a wrapper MSI that installs the package is created in the same folder. The MSI wrapper can be used to automate the deployment in systems which have no native support for App-V 5.X, but it can also be used to quickly install the packages in a test environment.

Copy App-V Launcher to the Output Folder
If this option is selected, each time when an App-V application is built, the App-V launcher utility is placed in the same folder as the generated App-V application. The tool can be used to start the package locally, before sending it to a deployment server.

For more information about App-V launcher, refer to *App-V launcher* section.

Version

This switch defines the target version of App-V packages. Depending on the selection, some features may be unavailable and the package may be not fully compatible with previous versions of App-V formats.

Target Operating Systems

If a packager does not define restrictions here, the generated package will be generated for compatible usage on all of the listed operating systems. However, by clicking on the **edit** button at the right-hand side of the current setting info label, the list of available OS becomes visible, ready for full or partial selection by checkbox activation.

Global Options

Enable static dependencies
When enabled, executable files contained within the package are scanned for a presence of static dependencies. We recommend to leave this setting enabled.

Treat INSTALLDIR as Primary Virtual Application Directory (PVAD)
When this option is enabled, the INSTALLDIR is converted to a root folder and all other folders are saved into VFS. If the option is disabled, then everything gets converted to VFS.

Allow virtual applications full write permissions to the virtual system
This option requires *App-V 5.0 SP2 Hotfix 4* or newer

COM Options

Enable COM Subsystem
Activate this checkbox to enable the COM subsystem in general. The following options will become editable as soon as the COM subsystem is active.

Enable COM in Process

COM In Process allows a COM object to be retrieved from the GUID based COM registration by the standard Windows API. It is then loaded into the process space of the calling application itself.

Enable COM Out of Process

Enabling this option triggers the launch of the `TaskHost.exe` when a COM API call is detected. The `TaskHost.exe` handles the communication between the calling virtual application and the COM object that fully remains on the host.

Options for App-V 4.6

Launcher Options

Create MSI Wrapper

If this option is selected, each time when an App-V application is built, a wrapper MSI that installs the package is created in the same folder. The MSI wrapper can be used to automate deployment in systems that have no native support for App-V 5.X, but also to quickly install the packages in a test environment.

Copy App-V Launcher to the Output Folder

If this option is selected, each time when an App-V application is built, the App-V launcher

utility is placed in the same folder as the generated App-V application. The tool can be used to start the package locally before sending it to a deployment server.

For more information about App-V launcher, refer to *App-V launcher* section.

Treat INSTALLDIR as Primary Virtual Application Directory (PVAD)
When this option is enabled, INSTALLDIR is converted to a root folder, and any other folders are saved into VFS. If the option is disabled, everything gets converted to VFS.

Streaming Options

Use Custom Streaming URL
The default setting for the streaming URL is derived from the environment variable. However, if this checkbox is activated and an individual path is entered, it will be used as streaming URL for this specific virtualized package.

Machine Configuration

Terminate Child Processes
If this option is active, any child process of an application that runs within the virtual environment will be terminated as soon as its main application is closed.

Target systems

If a packager does not define restrictions here, the generated package will be generated for compatible usage on all of the listed operating systems. However, by clicking on the **edit** button at the right-hand side of the current setting info label, the list of available OS becomes visible, ready for full or partial selection by checkbox activation.

PackageStore.com

RayPack has a built-in view allowing to browse and discover packages available in our innovative service PackageStore.com.

Configuring PackageStore integration

In order to use this module, one of the following must be true:

- The current product license includes PackageStore integration option enabled, or
- The PackageStore API token has been set-up in the Settings screen.

Discovering the packages

In order to browse the packages available in the Store, press **FILE** menu to bring the backstage menu, then click on the **New** tab and click the button **dicover + download packages.**
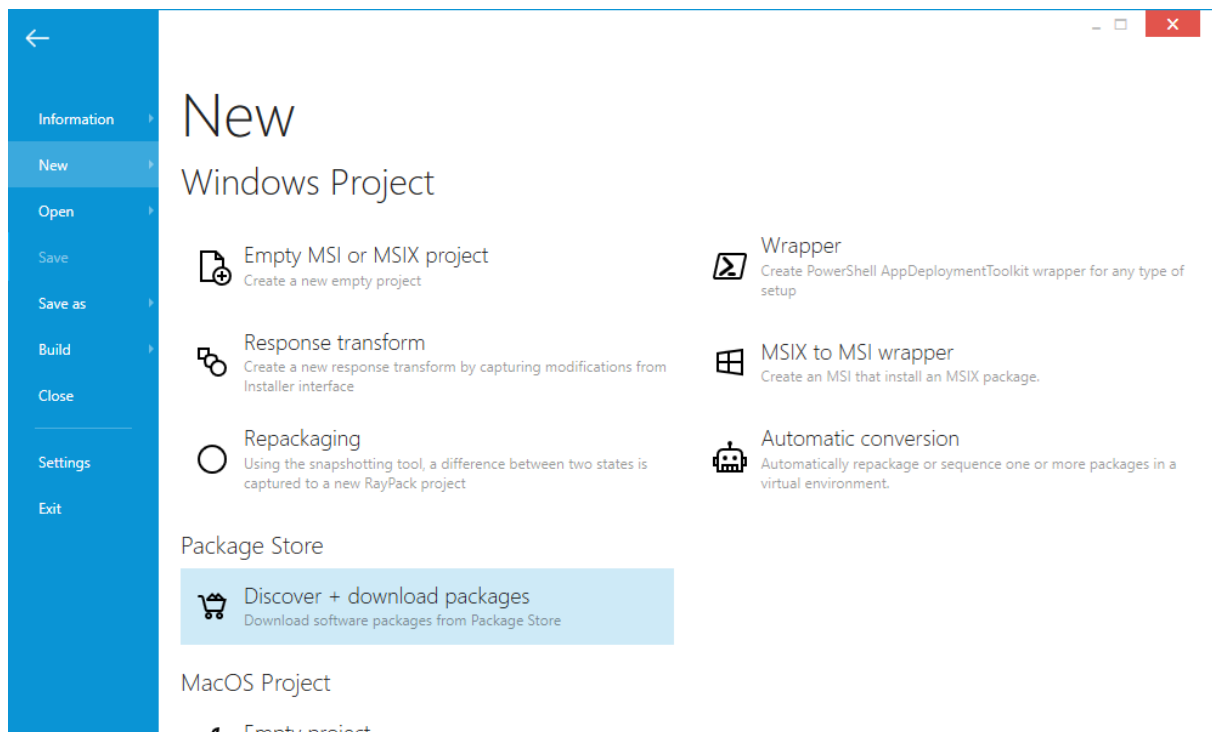
> **Note:**
> Interaction with Package Store requires internet connection.

Depending on the internet connection, loading the content may take a few seconds. When the content is ready, a list of available packages is presented.
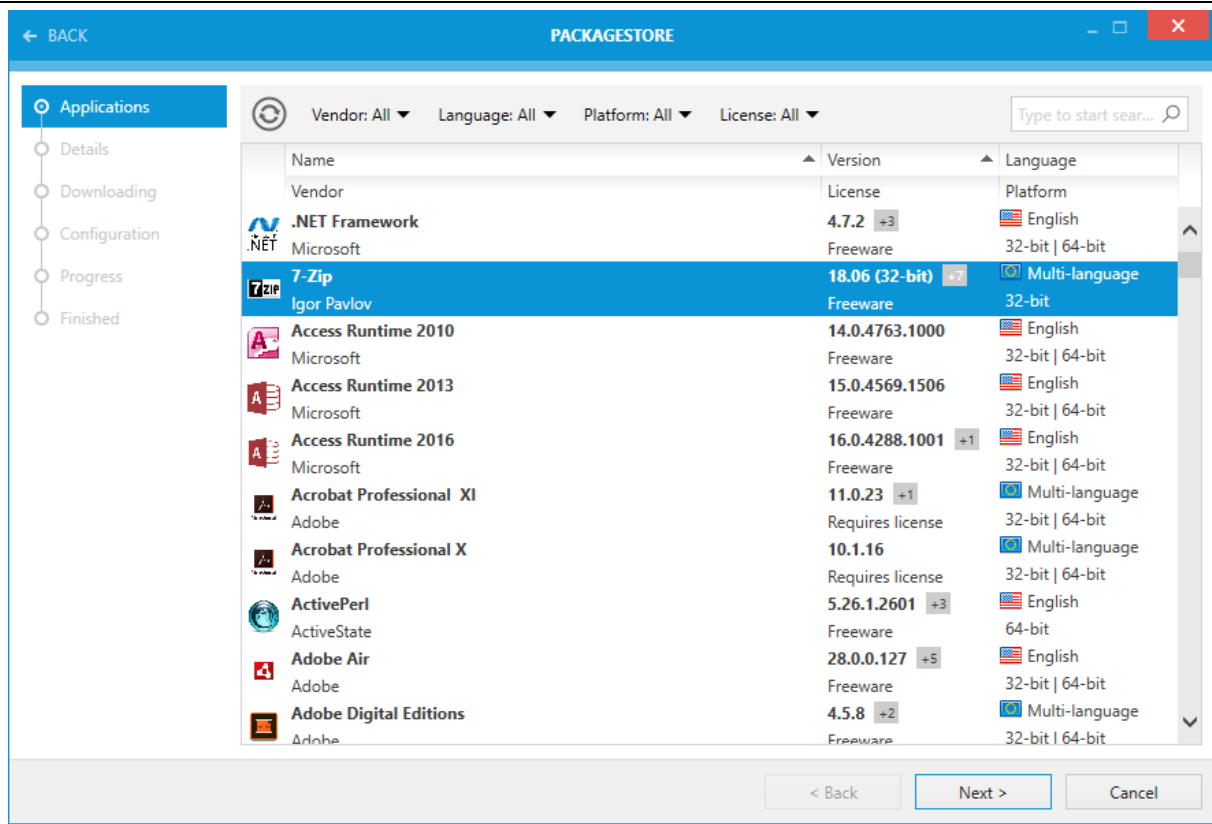
The grid shows the applications and their details. The following details are available.

- Product name
- Vendor
- Language
- Version
- License
- Platform

Filters which are available above the list can be used to refine the query (for example show only products in a specific language or from a specific vendor).

After selecting any item, its details are loaded into the right details panel. Should the selected application have more than one version in the Store, a dropdown list will be shown allowing to select particular version (in this case, the main grid also has a small badge informing that there are more versions available, see the screenshot above).

Press **Next>** to continue to the version selection.
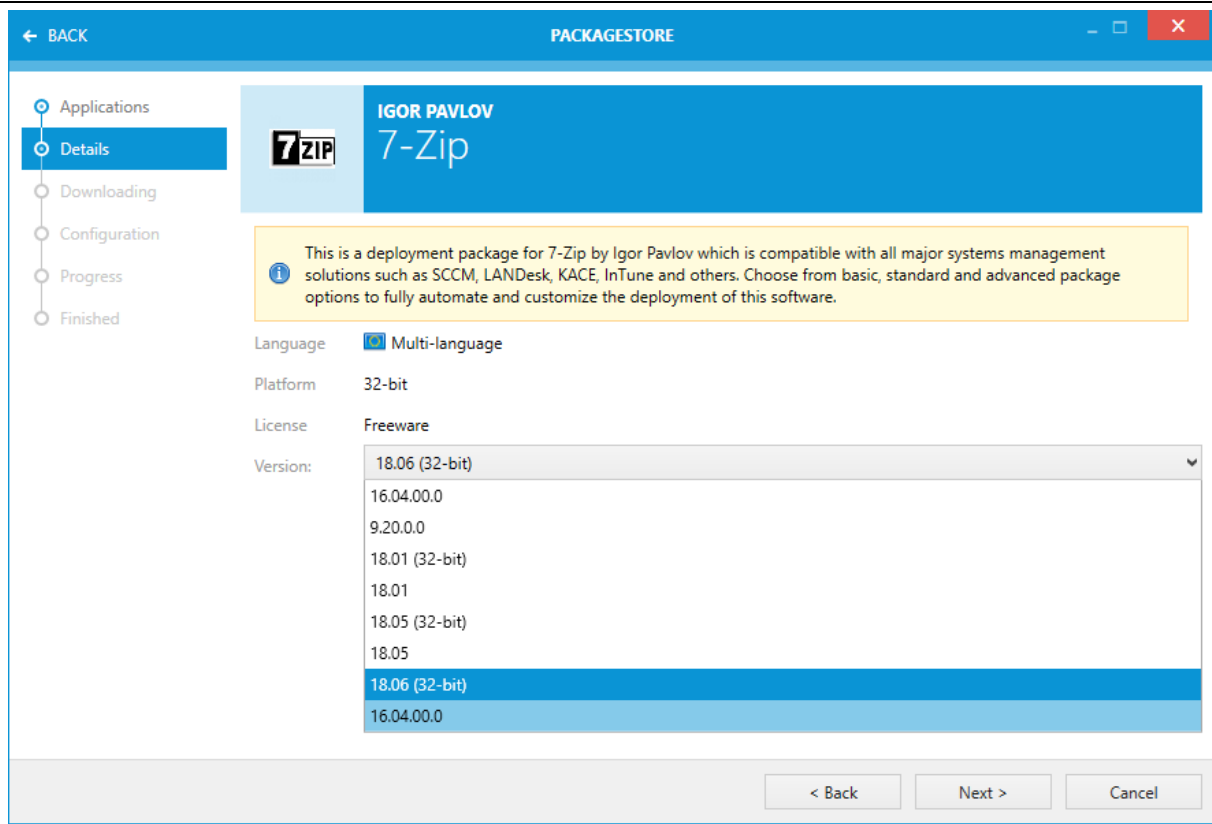
After selecting one of the versions from the drop-down menu, press **Next>** to download the selected package.

Downloading product sources
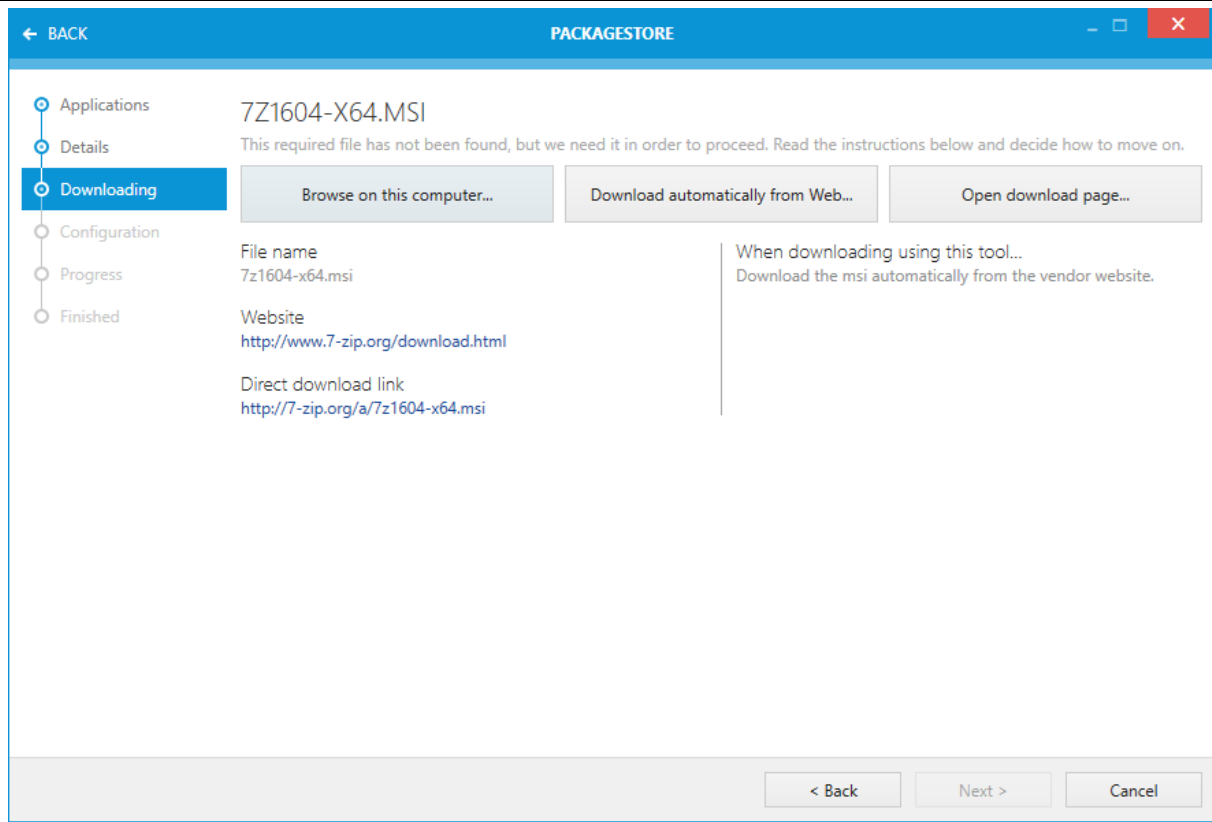
Depending on availability, up to three options are available:

- **Browse on this computer...**
  For recognized setups, this option lets user select a setup file (EXE/MSI/other format) that has
  been previously downloaded. Any required post-processing is executed automatically.
- **Download automatically from Web...**
  For recognized setups that are freely downloadable, RayPack offers an option to automatically
  download, extract and process the sources from the official vendor website.
- **Open download page...**
  If a link to a download page is known, pressing this buttons open the download link in the
  default web browser. After downloading the sources, use the first option (*Browse on this
  computer...*) to continue with the downloaded file.

Press **Next >** to go to the customization page.

Customizing the products

Depending on product and available options, a selection of configuration and user inputs is
presented to the user. This section is product-specific and offers high-level customizing which is
divided into:

- **STANDARDS**: The settings that are shared or similar in many products, including working with
  reboot, machine installation, silent setup etc.

- **ADVANCED**: The settings that are product-specific, discovered and described by our packaging specialists.
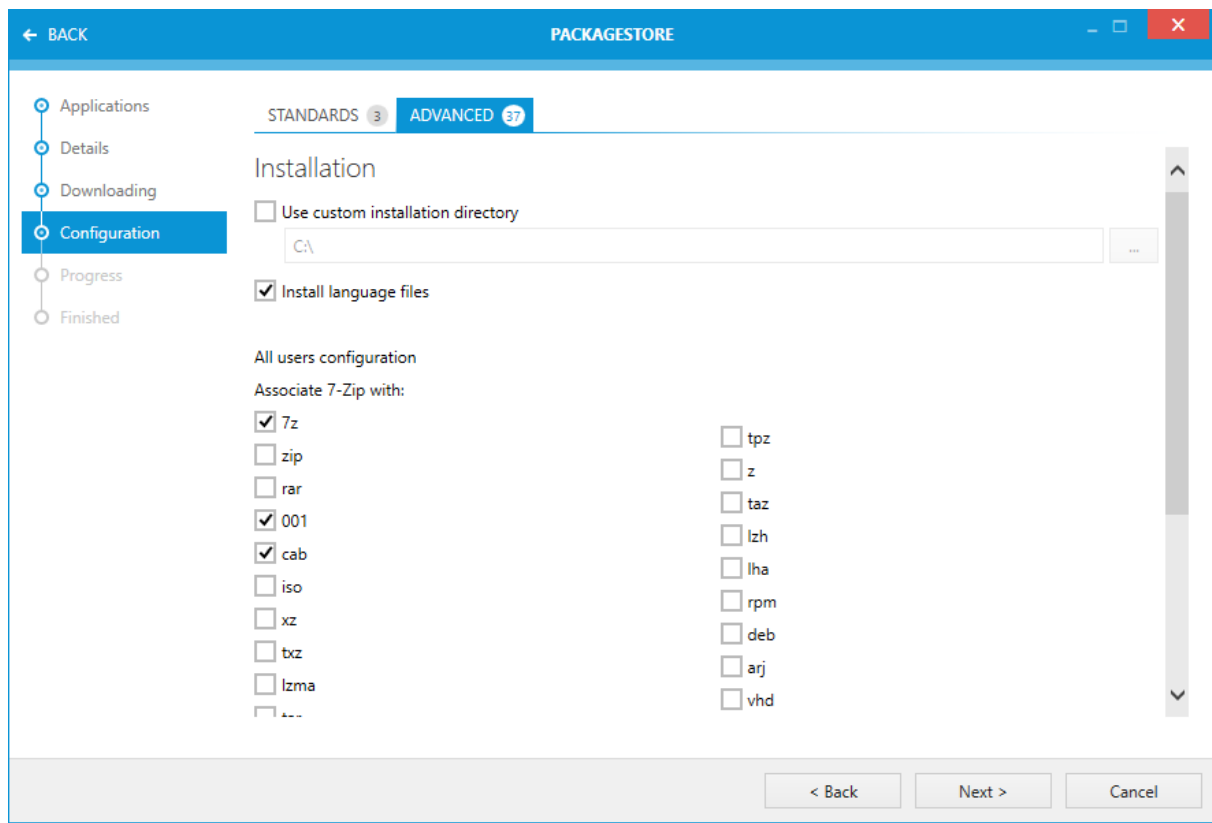
Once the button **Next >** is pressed, the creation of the package starts. After the wizard is finished, the user is able to open the project in PackDesigner (if any MSI file was involved).

Working with RayFlow

Introduction

RayFlow is a workflow process management tool with the ability to support diverse workflow processes. The possibility to be customized to fit the user's needs and requirements makes it one of the most efficient and user friendly workflow management tools. This guide shows how to configure and manage RayFlow, so that IT departments can stay ahead, save time, increase productivity, and decrease IT costs.

RayFlow is based on the client-server architecture in which all the information, data, and configuration is stored on the RayFlow server. Users work on this server remotely through the RayFlow web and Windows-based clients.

Where to find the latest information about RayFlow

For further information on RayFlow, including its features, functionality and latest updates visit www.raynet.de.

Enabling RayFlow features in RayPack

RayFlow connection are stored in profile configuration. The minimal configuration requires entering the URL address containing a valid, running instance of RayFlow server.
See RayFlow configuration for information how to configure this feature.

Signing-in to RayFlow

Saving and opening files from RayFlow requires that the current user is signed-in to the RayFlow instance specified in the configuration screen.

The sign-in procedure makes sure that:

- The user has permissions to a specified RayFlow project.
- The user has permissions to see / edit required RayFlow tasks.

In order to sign-in, press the **RayFlow** button located in the top right corner of the screen.
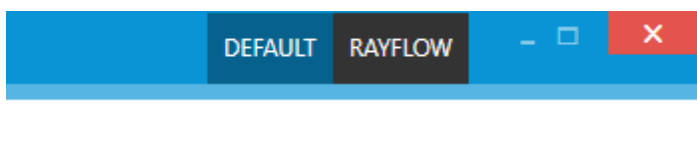
If the server is not valid or not specified, the following warning will be shown:
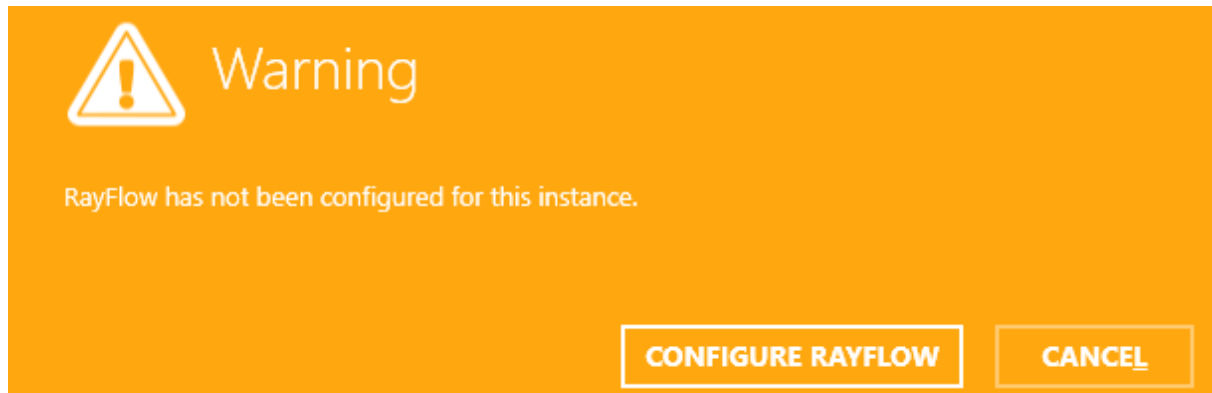
Otherwise, the sign-in overlay will be displayed over the current window:
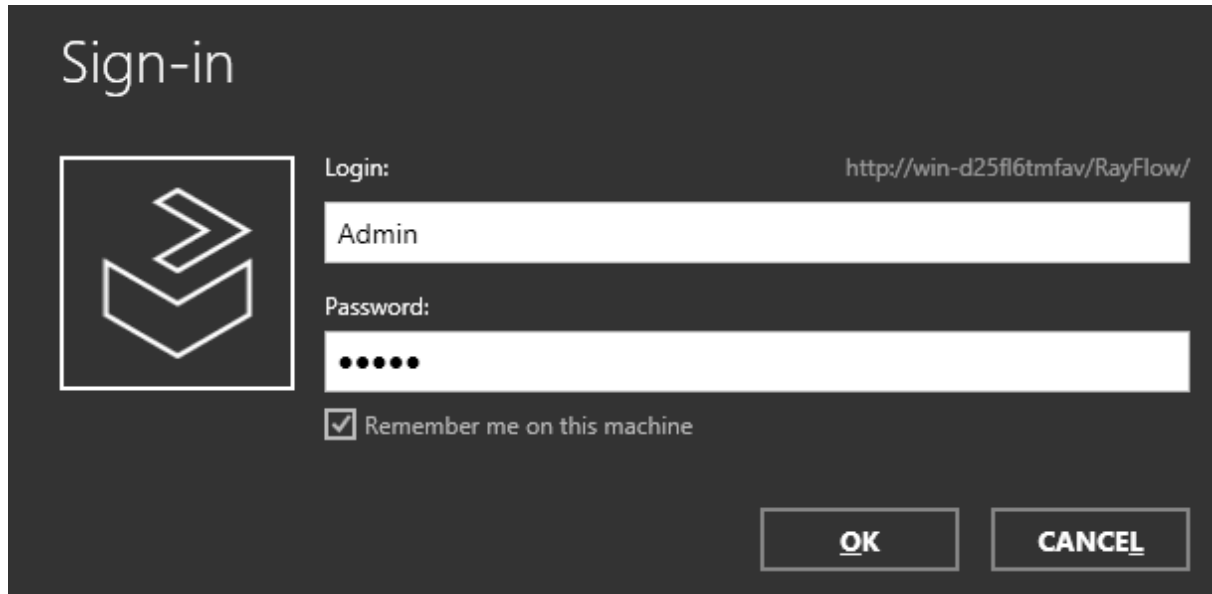
The overlay contains the following information:

- The URL address of the current RayFlow instance, as configured in the profile settings.
- The text fields for the user name and the user password.
- A checkbox to remember the credentials on the current machine.

> **Note:**
> The RayFlow credentials have to be delivered by the local RayFlow administrator.

Login and password are required to sign-in to RayFlow.

Once the credentials are verified, a selection of projects available in the current RayFlow

instance will be shown.

> **Note:**
> Only projects to which the current user has view permissions are displayed.
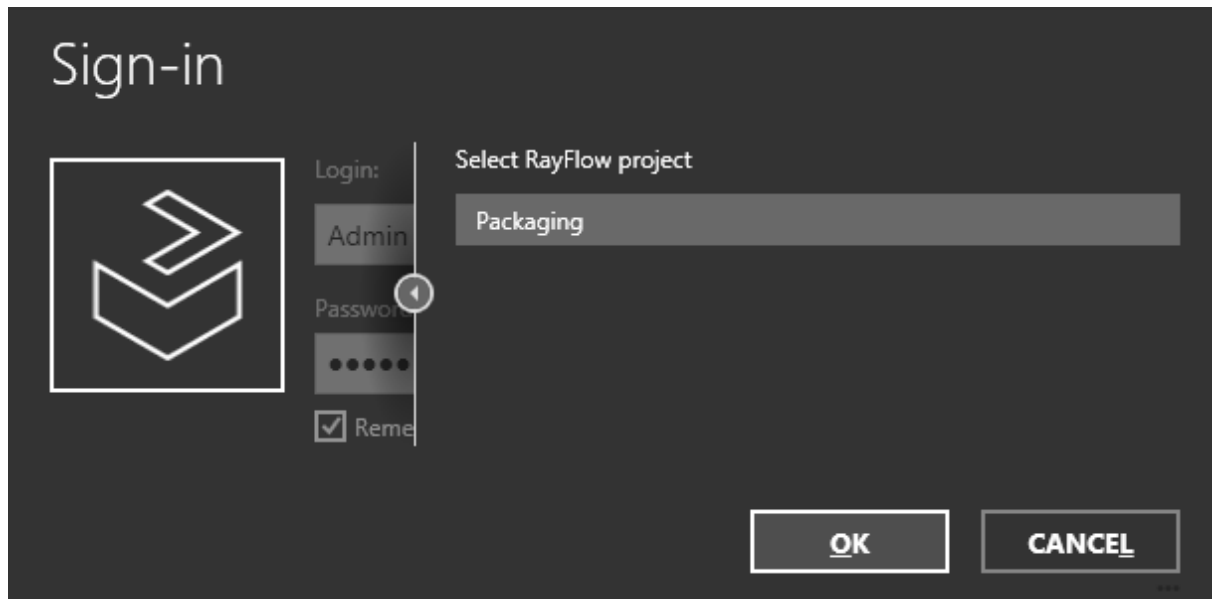
Once the project is selected, the sign-in procedure is complete and certain RayPack functions (like opening and saving files to RayFlow are available).

Once authenticated, the **RayFlow** button in the top right corner of the screen changes its color and displays the user name of the currently authenticated user. After clicking on it, additional details are shown including:

- The current project.
- A URL address of the current instance.
- A button to logout.
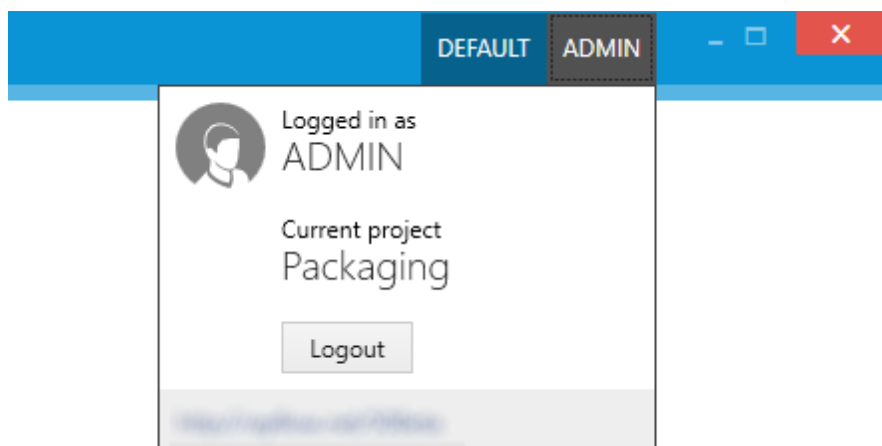
Opening Files from RayFlow

In order to open a file from RayFlow, the following prerequisites have to be carried out first.

1. *RayFlow instance has been configured in profile settings*
2. *User has signed-in and selected a RayFlow project*

In order to open a package from RayFlow

1. Press **FILE** button to show the backstage menu.

2. Click on menu item **OPEN**.

3. In the section **From RayFlow**, select **RayFlowTask**.
   (Note: when this option is selected before signing-in,it will ask for RayFlow credentials).
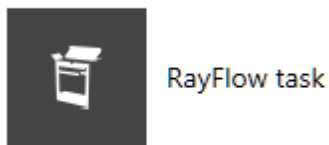
4. A new overlay window will be shown, prompting to select RayFlow phase, task, and a file belonging to a task.
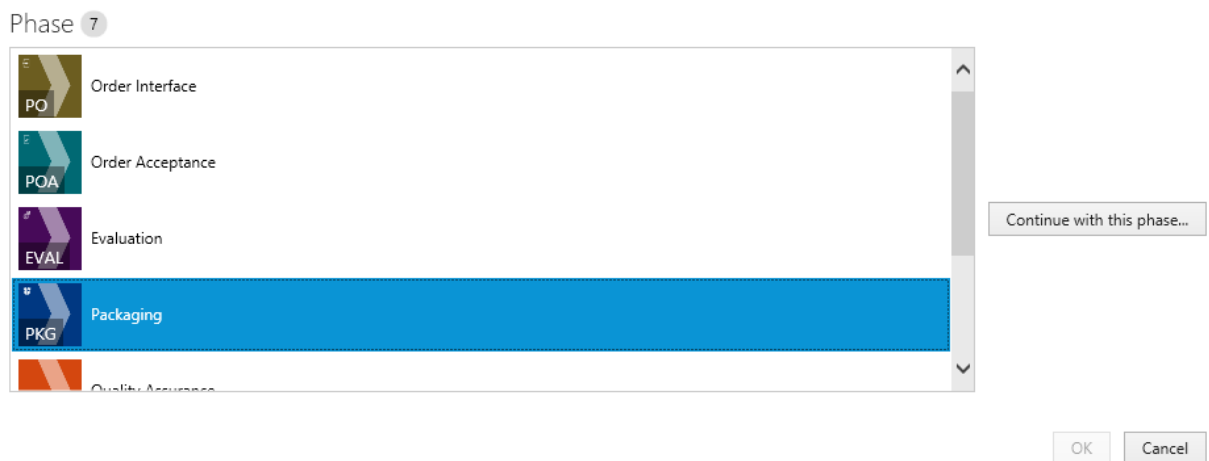
5. Select RayFlow phase which contains the task that should be opened. The counter above the list shows the total number of phases that are visible to the current RayFlow user.

6. Click **Continue with this phase...** to proceed to the task selection with the currently selected phase. Press **Cancel** to go back to the backstage menu.

7. Next, select the task that should be opened.
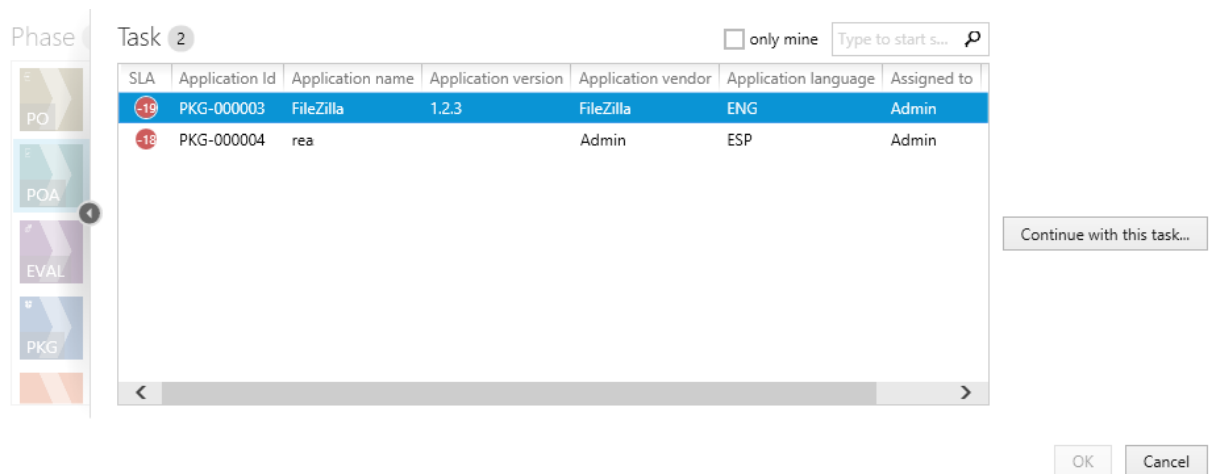
The number displayed above the list shows the total number of tasks that are visible to the current RayFlow user. The results can be narrowed by ticking the **only mine** option, which hides all packages not belonging to the current RayFlow user. The task browser displays the first 5 data fields from RayFlow to enable easier identification (refer to the RayFlow documentation about setting up the data fields).

8. The results can also be narrowed by searching, using the search box in the top right corner of the list.

9. Once the required task is selected, press **Continue with this task...** to proceed to the file selection. Press **Cancel** to go to the backstage menu. In order to go back to the phases view, click the arrow on the left side of the list or the partially transparent area containing the tiles representing available phases.

10. On the next screen, select the file to open.
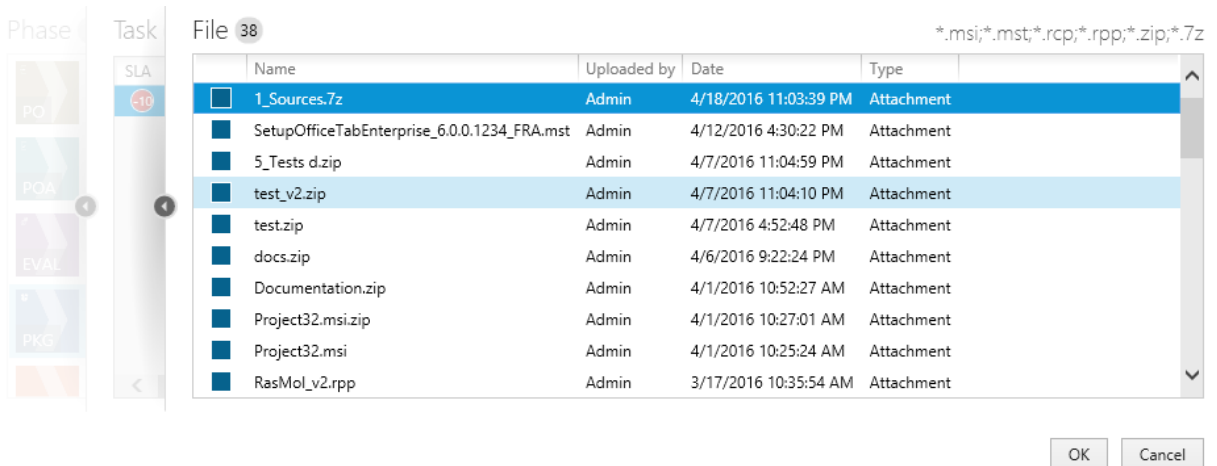
11. The number displayed in the top left corner represents the number of files that are present in

the currently selected package.

- o The type column represents the source from which the file can be accessed. Possible values are **Attachment** (file has been physically uploaded to RayFlow) and **Link** (the package is available on a shared location and only the path to the root file is saved in RayFlow).
- o **Date** and **Uploaded by** columns denote who and when has uploaded the file.
- o The list is sorted by the date, the most recent items are displayed on the top of the list.
- o Only the following file extensions are shown.
  - ▪ `.msi`
  - ▪ `.mst`
  - ▪ `.rcp`
  - ▪ `.rpp`
  - ▪ `.zip` (the file will be extracted, see section *Handling multiple and supporting files within a single package*)
  - ▪ `.7z` (the file will be extracted, see section *Handling multiple and supporting files within a single package*)

12. Press **OK** to open the selected file. Press **Cancel** to go back to the backstage. In order to change RayFlow task or phase, click the arrow on the left side of the list or the partially transparent area containing the list of tasks.

Storing RayFlow files locally

RayPack stores the downloaded RayFlow files in the following folder.

```
<RayPackProjectsFolder>\RayFlow\<TaskId>\<FileName>\
```

The four placeholders mean the following.

- The default folder for RayPack projects (configurable in **Projects** tab in RayPack Settings).
- A GUID of a task to which the package belongs to.
- A file name of the package (Keep in mind that all files in a single RayFlow task have unique names within the scope of a task).

When opening a project from RayFlow, RayPack checks this folder and if it is able to find any supporting file it will first ask whether to open a local copy or to download the file again, overriding any changes that might have been done since last download.
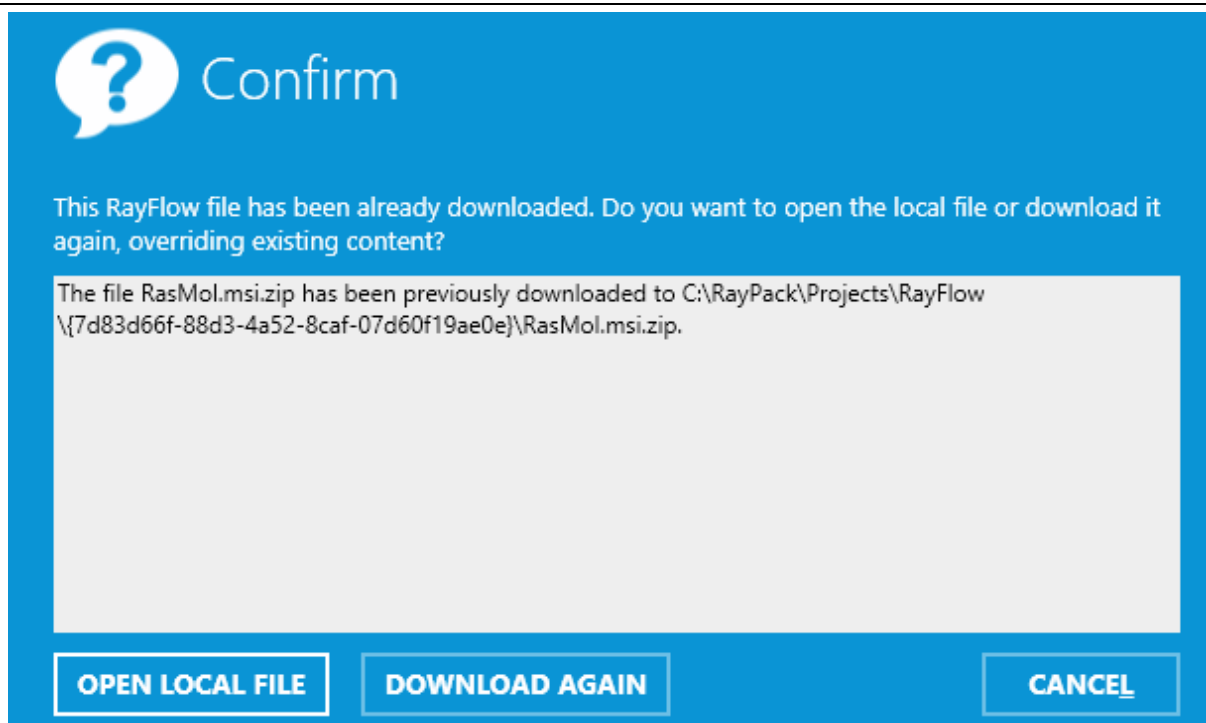
- Pressing **OPEN LOCAL FILE** opens the already existing file, preserving any changes since the last download.
- Pressing **DOWNLOAD AGAIN** downloads the file again. This option overrides all local changes.

> **Note:**
> Once uploaded to RayFlow, files never change. Each time adjustments are made and uploaded back, the files receive new names ensuring they are unique in the scope of a single task. Thus, the option **DOWNLOAD AGAIN** does not have to be used, unless the user wants to discard any changes he made locally to the package. In order to get the latest version of the package file that might have been uploaded since, simply open the newest file from RayFlow.

Handling multiple and supporting files within a single package

RayFlow stores multiple files (for example MSI and supporting files) as ZIP containers. When RayPack user requests such file, its content is extracted first. Then, the following may happen depending on the content.

1. If the compressed archive was already uploaded directly from RayPack , then the correct file will be automatically identified as an entry point and opened in PackDesigner.
2. Otherwise, RayPack tries to determine the entry point by looking for supported extension in the extracted folder. If exactly one is found, that file is opened in PackDesigner.
3. Otherwise if there is more than one supported file, RayPack looks for the file which has the same name as the name of the ZIP archive (less ZIP extension). If any is found, that file is opened in PackDesigner. Otherwise, RayPack shows a list of supported files and asks user

which one should be opened in PackDesigner.
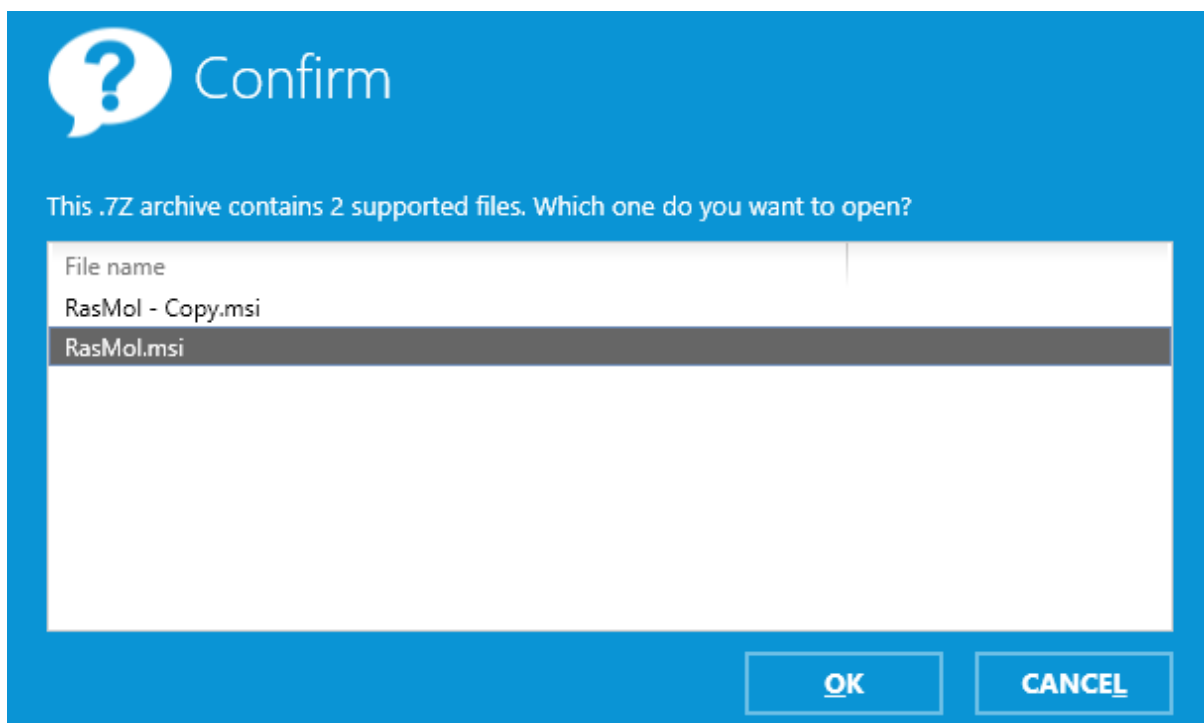
4. If no file could be found using methods described above, then the file will not be opened at all and RayPack will show a respective warning.

> **Note:**
> Only ZIP and 7Z containers are supported by this functionality. Archives in other formats (for example `.rar`) have to be extracted manually.

File depots

The files are shown within the browser irregardless of RayFlow File Depot they come from (see chapter *Using file depots* for more information and requirements of this feature). Once the file is requested, RayPack downloads it transparently for every supported type of depot (*local*, *WebDav*, *FTP*) without any further user interaction. The file browser contains an extra column informing about source **File Depot**.

Saving Files in RayFlow

In order to save or build a file from RayFlow, the following prerequisites have to be carried out first.

1. *RayFlow instance has been configured in profile settings.*
2. *User has signed-in and selected a RayFlow project.*
3. Supporting project is opened within PackDesigner or PackRecorder (supported formats are:

RPP, MSI, RCP, MST).

In Order to Save a Package from RayFlow

1. Press **FILE** button to show the backstage menu.

2. Click on menu item **SAVE AS** (to save RCP / RPP / MSI / MST files directly in RayFlow) or **BUILD** (to build MSI / RPP / RCP file to an MSI / RPP / ThinApp / SWV / App-V package and then upload it to RayFlow).

   More information about the differences and expected formats can be found in the chapter *Building packages*.

3. Select radio button **To RayFlow.**
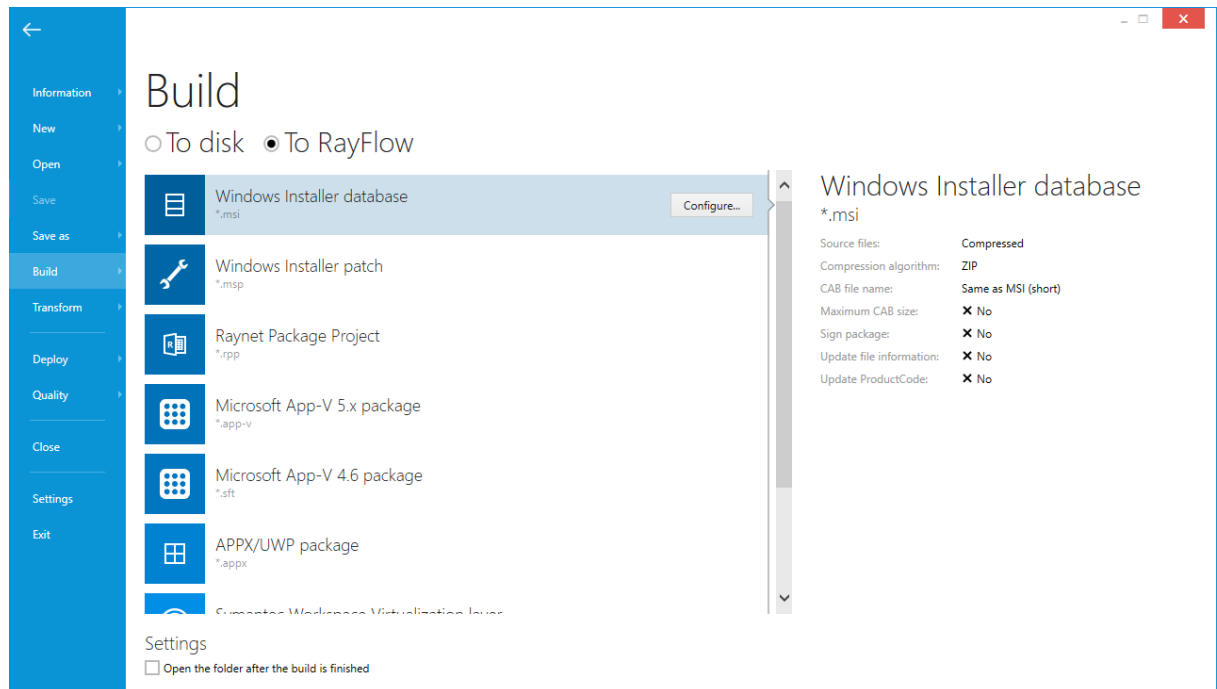
4. Select the target file format by clicking on it. A new overlay window will be shown, prompting to select RayFlow phase and task.
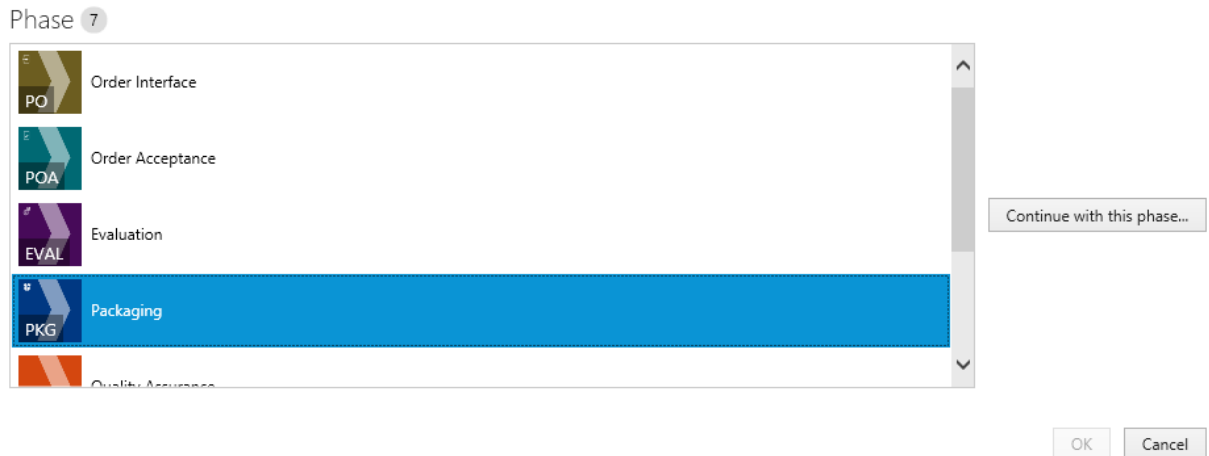
5. Select the RayFlow phase which contains the task where the file should be saved / built. The counter above the list shows the total number of phases that are visible to the current RayFlow user.
6. Click **Continue with this phase…** to proceed to the task selection with the currently selected phase. Press **Cancel** to go back to the backstage menu.
7. Next, select the target RayFlow task.
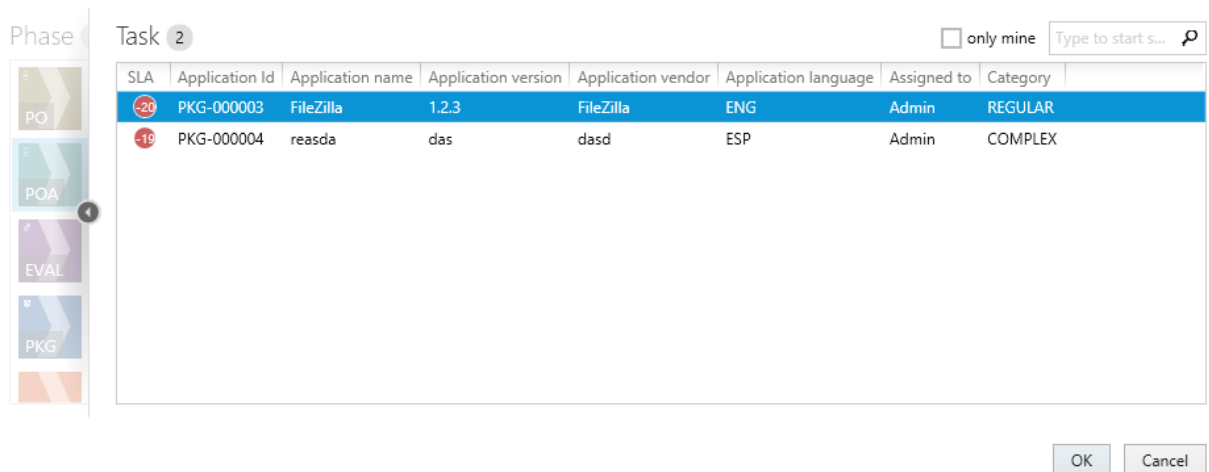
The number displayed above the list shows the total number of tasks that are visible to the current RayFlow user. The results can be narrowed by ticking the only mine option, which hides all packages not belonging to the current RayFlow user. The task browser displays first 5 data fields from RayFlow to enable easier identification (refer to the RayFlow documentation about setting up the data fields).

> **Note:**
> The screenshots above shows a default view with no RayFlow File Depots configured. The chapter Using file depots shows how to use the file depot selector once required features are available and configured on RayFlow Server side.

8. The results can be narrowed by searching, using the search box in the top right corner of the list.
9. Once the required task is selected, press **OK** to start building / saving to RayFlow. Press **Cancel** to go to the backstage menu. In order to go back to the phases view, click the arrow on the left side of

the list or the partially transparent area containing the tiles representing available phases.

Storing and Uploading RayFlow Files

The current project will be analyzed and uploaded to RayFlow once the build / save procedure is finished. The following rules apply to the name and format of the uploaded file.

- If the output is a single file (for example an `.msi` file without supporting files) then the file will be uploaded as-is to RayFlow.
- Otherwise, if the output is a main file with supporting files (for example an App-V package containing icons, XML metadata, `.osd` file etc.) then the file will be compressed to a `.zip` format and uploaded to RayFlow using the original file name plus `.zip` extension (for example `MyProject.rpp.zip`)
- Regardless of whether the first or second option is used, RayFlow ensures that the file names are unique within the scope of a single task, and the file name might get adjusted by appending an index to the file name. If any change is required, RayPack notifies the user.
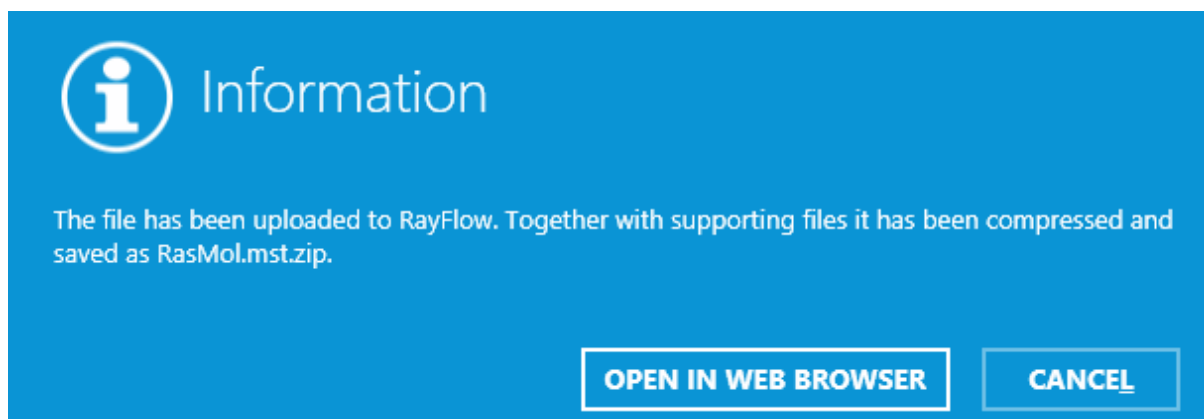
By pressing **OPEN IN WEB BROWSER**, a default Web Browser starts and the permanent link of the affected RayFlow task will be opened automatically (if the current user is also at the same time signed-in into the web version of RayFlow).

Using File Depots

RayPack 6.5 supports file depots configured via RayFlow. With **File Depots**, an option to upload and download files to and from locations different then the IIS server is available. The new target location for uploading files can be easily selected using a new dropdown box that is available when selecting to upload a file.

In order to use the new functionality, some preconditions have to be fulfilled.

- RayFlow has to be upgraded to version 4.1 or higher.
- Any file depot (other than local) must be configured.
- At least 1 non-local depot has to be available for the current user / phase combination.

If the conditions are met, when uploading a file to RayFlow the selector in the screenshot below

is shown.

By using the drop-down menu it is possible to select the preferred target upload location. The remaining steps are always the same, the upload procedure is fully transparent and requires no further interaction from the user side.

> **Note:**
> For more information about RayFlow and how to configure **File Depots**, refer to its *Product User Guide*.

Repackaging and Tailoring Files Downloaded from RayFlow

Files uploaded to RayFlow can be used as sources for repackaging (PackRecorder wizard) and tuning (PackTailor wizard). The exact procedure is describes in a respective chapter:

For PackRecorder:
 o Repackaging files from RayFlow

For PackTailor
 o Tailoring files from RayFlow

Signing out from RayFlow

In order to sign-out from RayFlow, or sign-in as another user, press the RayFlow button in the right top corner:
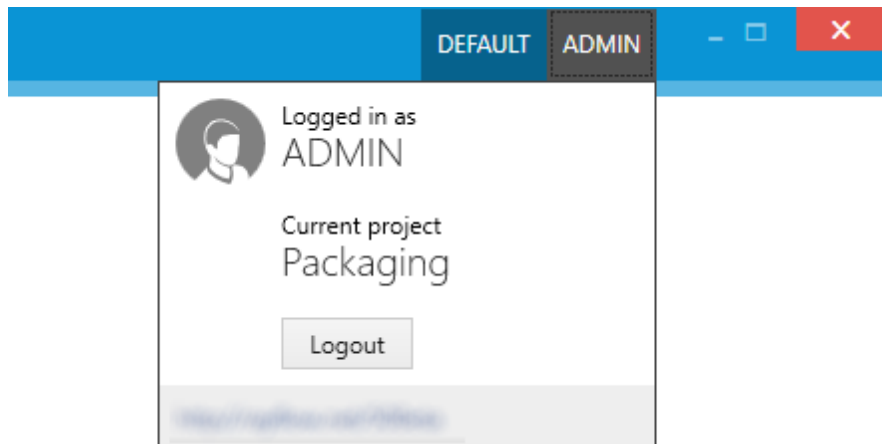
And then click the **Logout** button to sign-out from RayFlow.

Common Dialogs

Some dialogs of the RayPack interface cover generic functionality, such as managing a folder structure, or using an inline editor interface for property manipulation. They are often re-used as integral parts of several views.

The following common dialogs for standard procedures are covered within this section:

- The FILE menu
- Direct Value Editor

The FILE Menu

Clicking the **FILE** button contained within the Main Toolbar opens the file menu. This menu allows quick access to common functions.
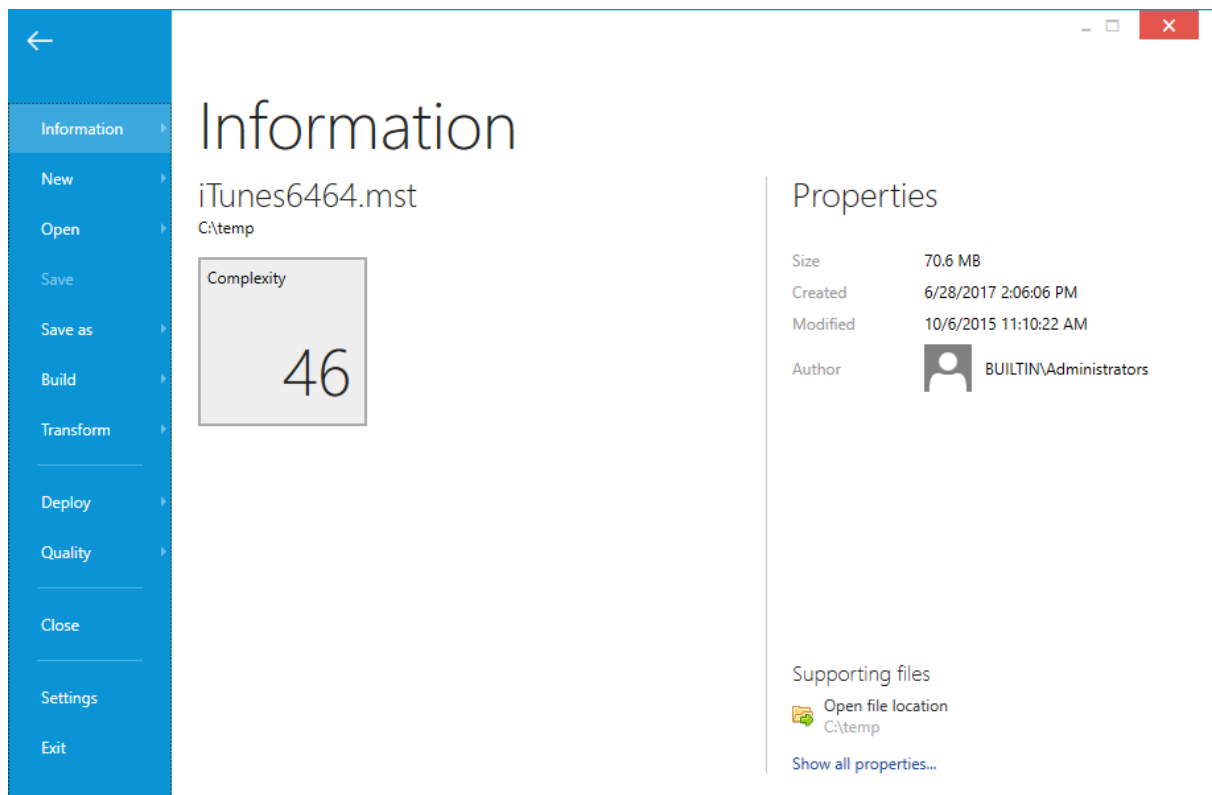
Click the items to navigate to the various sections

Information

Displays the view that is shown within the screenshot above. It presents some core properties of the currently opened packaging project or package.

New

Opens the new project view. If a project is loaded and there are outstanding changes, RayPack will ask to save any changes before continuing.

Open

Open an existing project and or package. If a project is loaded, RayPack will ask to save any changes before continuing.

Save

Saves any outstanding changes in the currently open project / package. Please note that this button is only active when pending changes are detected.

Save as

Allows to save the currently opened project under another name, or directly in RayFlow.

Build

Allows to export / build the current project as another type of project package, or directly in RayFlow.

Transform

Opens a transform menu, allowing to review, apply and save transforms, and to manage and apply transform templates.

Quality

Opens the **Quality** view.

Deploy

Starts the **Deployment** wizard.

Close

Closes the current project. If any changes are pending, RayPack will ask to save any changes before continuing.

Settings

Opens the **Settings** view.

Exit

Closes the current project and the whole RayPack application. If any changes are pending, RayPack will ask to save any changes before continuing.

Direct Value Editor

The Direct Value Editor is available wherever an edit icon is displayed next to a property or setting value. This handy data manipulation method allows property management directly within the original context, which is quicker and clearer than working with traditional interface types such as modal dialogs or edit forms.
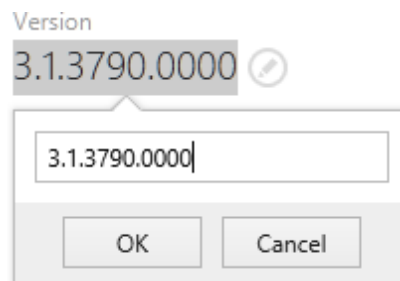
If the currently edited value is restrained by rules regarding its required formatting or domain, the Direct Value Editor supports users to comply with these rules by automatic value correction, or giving textual and visual feedback – exactly where the users attention is focused and without delay.

The illustration above shows an editor interface for plain text values. There is another data type available for direct value edition: Language settings:
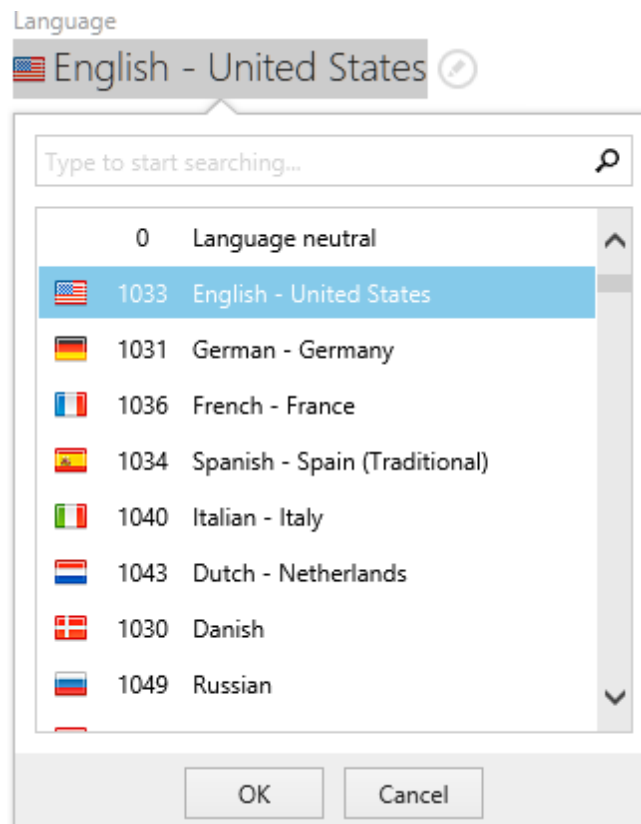
This interface is used for all package specific properties and values that have to be defined as language id value. In order to prevent users from having to know the correct id per language, RayPack provides a pre-defined set of languages that can simply be used for one-click selection.

The illustration above shows the direct value editor for a language setting that allows exactly one language values. However, there are other usages, where a column of checkboxes is provided at the left-hand side of the flag icons. Within these dialogs, it is possible to select several languages for the value of one package property.

How to use the Direct Value Editor interface

Clicking on either the property value or the **edit icon** right of it, allows to directly edit the current property value.
To **save** the changes, hitting the enter key on the keyboard or clicking on the **Save** button below the input field is due.
Discarding changes is done by hitting the escape key on the keyboard or clicking on the **Discard** button below the input field.
Moving the focus between the dialog controls is achieved by hitting tab on the keyboard.

Command Line Tools

The set of command line tools for RayPack consists of a direct command line access to the main program executable, and an additional builder tool for silent build execution. The following sections provide an overview of available options, required settings, and typical usage examples.

> **Note:**
> The command line interface is case insensitive. Therefore, it does not matter if the switch ProjectGeneration is noted as "ProjectGeneration", "projectgeneration", or "PROJECTGENERATION". Please use the diction that is recommended within your enterprises system integration style guidelines.

Error Codes

Whenever a command is sent to the RayPack Command Line Interface Tools, they respond with a specific error code:

- If **0** is returned, the command has been executed **successfully**.

- Any other return code implies that issues have occurred during the execution. Details regarding the reasons for errors have to be retrieved from the log files.

Command Line Switches

Some functions of the RayPack core executable (`RayPack.exe`) are available via command line. The following sections provide an overview of switches and parameters related to this type of command line usage.

> **Be aware:**
> Using the command line access to the main executable does not provide support for fully silent build procedures. If the use case for command line execution requires this to be given, please refer to the Command Line Builder section of this document for a detailed list of available commands and options.

Using RayPack via command line requires the application of switches. Each switch initiates a specific RayPack functionality, which may be configured by parameters added to the command line string.

Interactive mode

The following switches are available:

- **Edit**
  Opens an MSI package, MST file or an RCP/RPP project

- PackTailor
  Opens PackTailor wizard for response transform tuning

- Repackage
  Opens PackRecorder repackaging wizard

- Validate
  Starts ICE validation of a specified file

The following switches are deprecated as of RayPack 6.5, but remain in the product due to compatibility reasons. It is not recommended to use them anymore:

- Capture
- Open
- RayFlow

> **Be aware:**
> All RayPack command line switches and parameters are case insensitive. This means that parameter `Edit` is interpreted exactly the same as `eDIT`, and a switch `-PATH` is equal to `-Path`

Edit (Interactive)

Description

This switch instructs RayPack to open a project from a location specified by the `-Path` parameter.
The path specified within the `-Path` parameter may take any kind of editable project supported by one of the RayPack editors: RCP, RPP, MSI, MST.

> **Note:**
> When the path to an MST is provided as value of the `-Path` parameter, RayPack will prompt a browse dialog once it has launched, demanding the selection of the base MSI for the specified MST. In order to provide the base MSI and avoid additional prompts, use the `-mst` parameter.

Parameters

Parameter	Required	Description
-Path <path>	YES	This is the fully qualified path to the project that is about to be opened for editing in RayPack. The parameter may take any kind of editable project supported by one of the RayPack editors: RCP, RPP, MSI, MST. The path can be either a local or an UNC path.
-Profile <path>	NO	This is the full path to the profile to be used by this instance. The path can be either a local or an UNC path. When omitted, the last used profile will be used.
-MST <path>	NO	This parameter can be use more than once. It has to be a full local or UNC path of an MST transform to be applied on a loaded base MSI file.

Example

This command line will start RayPack, and open an existing project. If an RCP file is referenced by the path defined by `-Path`, the PackRecorder Editor will be launched, whilst all other supported formats launch the PackDesigner Editor interface.

```
RayPack.exe Edit
-Profile "\\raypackshare\raynet.rpprofile"
-Path "\\rayflowshare\sources\Projects\7-Zip_6.5_MUL_1.0.0\7-Zip_6.5_MUL_1.0.0.rcp"
```

Using as default switch

The **Edit** switch is a default switch in RayPack. The following syntax, using the file path as the first argument.

```
<path_to_raypack.exe> "c:\test\myMsi.msi"
```

This is a shortcut which translates to the following.

```
<path_to_raypack.exe> Edit
-Path "c:\test\myMsi.msi"
```

In order to specify transforms using the default syntax, use one of the following commands.

```
<path_to_raypack.exe> "c:\test\myMsi.msi"
-mst "c:\test\f1.mst" -mst "c:\test\f2.mst"

<path_to_raypack.exe> "c:\test\myMsi.msi" "c:\test\f1.mst" "c:\test\f2.mst"
```

These are both functionally equal to the following.

```
<path_to_raypack.exe> Edit
-Path "c:\test\myMsi.msi"
-MST "c:\test\f1.mst"
-MST "c:\test\f2.mst"
```

Deprecated parameters

The following parameters are deprecated but remain in the product due to compatibility reasons. It is not recommended to use them anymore.

Parameter	Required	Description
`-RayFlowPackageId <guid>`	NO*	When RayPack is launched from RayFlow, or operates connected to a RayFlow server,

Parameter	Required	Description
		this parameter contains the unique identifier of a specific package as populated within the RayFlow database.
`-RayFlowProjectId <guid>`	NO*	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the GUID of the project the current connection is directed to.
`-RayFlowServicePassword <password>`	NO*	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the password of the user profile the connection is assigned to. The credentials used for the RayFlow connection determine the scope available projects, packages, and methods on RayFlow data objects.
`-RayFlowServiceUser <user>`	NO*	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the name of the user profile the connection is assigned to. The credentials used for the RayFlow connection determine the scope available projects, packages, and methods on RayFlow data objects.
`-RayFlowUrl <path>`	NO*	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the address of the RayFlow server.

Note:
The five parameters starting with RayFlow are optional as long as none of them is defined. As soon as one of them is defined, all five parameters are needed.

Example

```
RayPack.exe Edit
-Path "C:\RayPack\Projects\RasMol\RasMol.msi"
-RayFlowPackageId "123-345-657-789-213"
-RayFlowServiceUser "Admin"
-RayFlowServicePassword "password"
-RayFlowUrl "http://rayflow.url"
-RayFlowProjectID "123-456-789-123-456"
```

PackTailor

Description

This switch signifies that RayPack is to start and to open the PackTailor Wizard for transform generation.
All parameters are used to fill in data that is requested by one of the PackTailor Wizard steps. If none are provided, all wizard steps have to be answered manually.

Parameters

Parameter	Required	Description
`-MSI <path>`	NO	Fully qualified path to the MSI that is about to be tailored. It can be either a local path or an UNC path. When omitted, the value will have to be entered manually in the wizard before processing to the tailoring process.
`-Transforms <path>`	NO	This parameter contains a list of paths for transform files, which are about to be applied to the base MSI before a tailoring procedure is executed. Use semicolon as a separator. The paths can be either local paths, or UNC paths, or mixed. When omitted, the value can be entered manually in the wizard.
`-IgnoreSystemState`	NO	This parameter works as a keyword trigger for the PackTailor switch: If it is present, the advanced option **Ignore current system state** within the wizard step Capturing changes is activated. When omitted, the current system state is respected.
`-IgnoreLaunchConditions`	NO	This parameter works as a keyword trigger for the PackTailor switch: If it is present, the advanced option **Ignore launch conditions** within the wizard step Capturing changes is activated. When omitted, LaunchConditions are respected.
`-MsiParams <params>`	NO	String with command line parameter definition. RayPack does not evaluate these parameters, but simply passes them as they are when the UI sequence is triggered during the tailor procedure. Please make sure to use the right format for quotes within the general parameter value as shown in the sample on the right.
`-MachineName <name>`	NO	The name of the virtual machine to use for the tailoring. The machines must be pre-configured, and the name must match one of them. If no machine is found, the tailoring does not fail but uses the local machine instead.
`-Output <path>`	NO	This is the fully qualified path to the MST that is to be created as result of a PackTailor procedure. When ommited, a default path based on profile settings will be used as a default value.

Example

This sample launches PackTailor with a specified MSI file and two transform files from different

locations. The combination of MSI and transforms will be used to specify the UI sequence behavior as well as other (invisible) MSI settings and properties.

```
RayPack.exe PackTailor
-MSI "\\rayflowshare\sources\7-Zip_6.5_MUL_1.0.0\7zip65.msi"
-Transforms "\\rayflowshare\sources\7-Zip_6.5_MUL_1.0.0\7zip65.mst;\\raypackshare\bran
```

Repackage (Interactive)

Description

Calling RayPack with this switch initiates a PackRecorder session, launching the PackRecorder Wizard in the parametrized mode.

Parameters

Parameter	Required	Description
-ProjectName <name>	YES	This is the name of the RCP project that is about to be created as result of a capture procedure with PackRecorder.
-ApplicationPath <path>	NO	Fully qualified path to the application that is about to be captured. This can be either a local path or an UNC path. If not provided, the value has to be entered manually in the wizard page.
-WizardMode <mode>	NO	When a capture procedure is triggered, RayPack needs to be informed about the required wizard mode. This parameter contains the actual name of the mode that has to be applied. Two values are supported: expert or standard. If not provided, the value will be taken from the current profile.
-ApplicationParameters <params>	NO	Parameters to pass to an application that is about to be captured. Commonly used in combination with -ApplicationPath. Please make sure to use the right format for quotes within the general parameter value as shown in the sample on the right.
-Profile <path>	NO	This is the full path to the profile to be used by this instance. The path can be either a local or an UNC path. When omitted, the last used profile will be used.
-ProjectDir <path>	NO	The full path to the RCP project that is about to be created as result of a capture

Parameter	Required	Description
		procedure with PackRecorder. The path can be either a local or an UNC path. If omitted, the default value from the profile will be used instead.

Example

This command line will start RayPack, and initiate the PackRecorder Wizard in the standard mode. The snapshots made during the wizard run will be configured according to the specified profile. The installer used for the wizard is defined within the `-ApplicationPath` parameter.

```
RayPack.exe Repackage
-ApplicationPath "\\rayflowshare\sources\7-Zip_6.5_MUL_1.0.0\7zip65.exe"
-ProjectName "7-Zip_6.5_MUL_1.0.0"
-WizardMode "standard"
```

Deprecated parameters

The following parameters are deprecated but remain in the product due to compatibility reasons. It is not recommended to use them anymore.

Parameter	Required	Description
`-RayFlowPackageId`	NO*	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the unique identifier of a specific package as populated within the RayFlow database.
`-RayFlowProjectId`	NO*	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the GUID of the project the current connection is directed to.
`-RayFlowServicePassword`	NO*	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the password of the user profile the connection is assigned to. The credentials used for the RayFlow connection determine the scope available projects, packages, and methods on RayFlow data objects.
`-RayFlowServiceUser`	NO*	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the name of the user profile the connection is assigned

Parameter	Required	Description
		to. The credentials used for the RayFlow connection determine the scope available projects, packages, and methods on RayFlow data objects.
-RayFlowUrl	NO*	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the address of the RayFlow server.

Note:
The five parameters starting with RayFlow are optional as long as none of them is defined. As soon as one of them is defined, all five parameters are needed.

Example

```
RayPack.exe Repackage
-ApplicationPath "C:\RayPack\Projects\RasMol\RasMol.msi"
-ProjectName "RasMol"
-WizardMode "standard"
-RayFlowPackageId "123-345-657-789-213"
-RayFlowServiceUser "Admin"
-RayFlowServicePassword "password"
-RayFlowUrl "http://rayflow.url" -RayFlowProjectID "123-456-789-123-456"
```

PackBot (Interactive)

Description

This switch instructs RayPack to start with a new task executed by the PackBot wizard.

Parameters

Parameter	Required	Description
-applicationPath <path>	NO	This is the fully qualified path to the setup file that is about to be repackaged or converted. The path can be either a local or an UNC path.
-applicationParameters <parameters>	NO	The list of parameters to pass to the setup file

Parameter	Required	Description
`-projectName <name>`	NO	The name of the project (used also for the name of the output directory). If not specified, then the name is read and set according to setup metadata.
`-projectDir <folder>`	NO	Root projects folder (if not specified, then the default project path is used)
`-targetFormat <format>`	NO	One of the following: MSI, RCP, RPP, APPV, MSIX, EXE, SFT. If not specified, the default value MSI is used.
`-virtualMachines <names>`	NO	The list of virtual machines to use separated by semicolons (;) to indicate a pool of machines. If not specified, then the default machines are used.

Example

This command line will start RayPack and open the **PackBot** wizard on a specific setup file.

```
RayPack.exe PackBot
-applicationPath "\\rayflowshare\sources\Projects\7-Zip_6.5_MUL_1.0.0\7-Zip_6.5_MUL_1
-projectName "7-Zip"
-virtualMachines "AME2017;App-V5 Sequencer"
```

Validate (Interactive)

Description

Calling RayPack with this switch initiates a PackDesigner session, launching the ICE validation.

Parameters

Parameter	Required	Description
`-Input <name>`	YES	Path to the MSI/RPP file to be validated.
`-Cub <path>`		Path to the CUB file with validation rules. This parameter

Parameter	Required	Description
	YES	can be defined more than once.
`-Output <mode>`	NO	Path to the file with validation results.
`-MST <params>`	NO	Path to the MST file. This parameter can be defined more than once.

Example

This command line will start RayPack, and initiate the PackDesigner ICE validation using full ICE set.

```
RayPack.exe Validate
-Input "\\rayflowshare\sources\7-Zip_6.5_MUL_1.0.0\7-Zip_6.5_MUL_1.0.0.msi"
-Cub "C:\Program Files (x86)\RayPack\Resources\Validation\darice.cub"
-Output "\\rayflowshare\sources\7-Zip_6.5_MUL_1.0.0\7-Zip_6.5_MUL_1.0.0.ICE.txt"
```

Capture (Deprecated)

> ⚠️ **WARNING:**
> This switch is deprecated, and will no longer be supported by the `RayPack.exe` provided by future RayPack releases. Use Repackage switch instead.

Description

Calling RayPack with this switch initiates a PackRecorder session, launching the PackRecorder Wizard in the parametrized mode.

Parameters

Parameter	Required	Description
`-App <path>`	YES	Fully qualified path to the application that is about to be captured. This can be either a local or an UNC path.
`-AppParams <par`	YES	Parameters to pass to an application that is about to be captured. RayPack does not evaluate these parameters, but simply passes them as they are when the application installation is triggered during the capture procedure. Please make sure to use the right format for quotes within the general parameter value as shown in the sample below.
`-Profile <path>`	YES	This is the fully qualified path and name of the configuration profile.
`-ProjectName <n`	YES	A name of the RCP project.
`-WizardMode <mc`	YES	When a capture procedure is triggered, RayPack needs to be informed about the required wizard mode. This parameter contains the actual name of the mode that has to be applied. Two values are supported: expert or standard.
`-Silent`	NO	When a switch is used with this trigger parameter, the initiated RayPack method is invoked without a visible user interface representation. This is especially useful when capture procedures are triggered from RayFlow, and the results of the procedure will directly be reused for subsequent activities.

Example

This command line will start RayPack, and initiate the PackRecorder Wizard in the standard mode. The snapshots made during the wizard run will be configured according to the specified profile. The installer used for the wizard is defined within the `-App` parameter.

```
RayPack.exe Capture
-Profile "\\raypackshare\raynet.rpprofile"
-ProjectName "7-Zip_6.5_MUL_1.0.0"
-App "\\rayflowshare\sources\7-Zip_6.5_MUL_1.0.0\7zip65.exe"
-AppParams ""
-WizardMode "standard"
```

Open (Deprecated)

> **WARNING:**
> This switch is deprecated, and will no longer be supported by the `RayPack.exe` provided by future RayPack releases.

Description

This switch signifies that RayPack is to start, and to open a project for editing.

When RayPack is used in a RaySuite context, this switch is commonly used to launch PackDesigner from a specific RayFlow package order object. It may be used from any application or script to access RayFlow package orders, and trigger resource editing in RayPack.

> **Note:**
> • This switch works only in conjunction with RayFlow. If an MSI/RPP package is to be edited, use the Edit switch.

Parameters

Parameter	Required	Description
-Path <path>	YES	This is the fully qualified path to the MSI or RPP file that is to be opened within PackDesigner. This can be either a local or an UNC path.
-Profile <path>	YES	This is the fully qualified path and name of the configuration profile.
-RayFlowPackageId <guid>	YES	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the unique identifier of a specific package as populated within the RayFlow database.
-RayFlowProjectId <guid>	YES	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the GUID of the project the current connection is directed to.

Parameter	Required	Description
-RayFlowServicePassword <password>	YES	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the password of the user profile the connection is assigned to. The credentials used for the RayFlow connection determine the scope available projects, packages, and methods on RayFlow data objects.
-RayFlowServiceUser <user>	YES	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the name of the user profile the connection is assigned to. The credentials used for the RayFlow connection determine the scope available projects, packages, and methods on RayFlow data objects.
-RayFlowUrl <path>	YES	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the address of the RayFlow server.

Example

The following example launches PackDesigner with an MSI of 7Zip opened for editing. The provided RayFlow parameters allow users to establish a connection to the RayFlow server once the result of the editing measures has to be transferred to RayFlow.

```
RayPack.exe Open
-Path "\\rayflowshare\sources\7-Zip_6.5_MUL_1.0.0\7zip65.msi"
-Profile "C:\Users\Administrator\AppData\Roaming\RayPack\Profiles\DEMO.rpprofile"
-RayFlowPackageId "47110070815"
-RayFlowServiceUser "UserName"
-RayFlowServicePassword "Password"
-RayFlowUrl "https://rayflow.raynet.de"
-RayFlowProjectId "3F2504E04F8941D39A0C0305E82C3301"
```

RayFlow (Deprecated)

> **WARNING:**
> This switch is deprecated, and will no longer be supported by the `RayPack.exe` provided by future RayPack releases.

Description

This switch signifies RayPack has been started by RayFlow. With this switch, the RayPack capture wizard is started with the user interface, using the parameters passed in the capture parameters (if no other switches are defined).

> **Note:**
> • This switch works only in conjunction with RayFlow. If a legacy setup (`.exe`) is about to be repackaged, use the Capture switch.

Parameters

Parameter	Required	Description
-App <path>	YES	Fully qualified path to the application that is about to be captured. This can be either a local or an UNC path..
-ProjectName <path>	YES	A name of the RCP project.
-Profile <path>	YES	This is the fully qualified path and name of the configuration profile.
-RayFlowPackageId <guid>	YES	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the unique identifier of a specific package as populated within the RayFlow database.
-RayFlowProjectId <guid>	YES	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the GUID of the project the current connection is directed to.
-RayFlowServicePassword <password>	YES	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the password of the user profile the connection is assigned to. The credentials used for the RayFlow connection determine the scope available projects, packages, and methods on RayFlow data objects.
-RayFlowServiceUser <user>	YES	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the name of the user profile the connection is assigned to. The credentials used for the RayFlow connection determine the scope available projects, packages, and methods on RayFlow data objects.
-RayFlowUrl <path>	YES	When RayPack is launched from RayFlow, or operates connected to a RayFlow server, this parameter contains the address of the RayFlow server.
-WizardMode <mode>		When a capture procedure is triggered,

Parameter	Required	Description
	YES	RayPack needs to be informed about the required wizard mode. This parameter contains the actual name of the mode that has to be applied. Two values are supported: expert or standard.
-AppParams <params>	NO	Parameters to pass to an application that is about to be captured. RayPack does not evaluate these parameters, but simply passes them as they are when the application installation is triggered during the capture procedure. Please make sure to use the right format for quotes within the general parameter value as shown in the sample below.
-Silent	NO	When a switch is used with this trigger parameter, the initiated RayPack method is invoked without a visible user interface representation. This is especially useful when capture procedures are triggered from RayFlow, and the results of the procedure will directly be re-used for subsequent activities.

Example

This command line initiates the PackRecorder to run in experts mode and to capture the installation of the 7Zip executable defined within the `-App` parameter.
The RayFlow connection parameters allow to export results and return them back to the media section of the specified package order object.

```
RayPack.exe RayFlow
-App "\\rayflowshare\sources\7-Zip_6.5_MUL_1.0.0\7zip65.exe"
-Profile "C:\Users\Administrator\AppData\Roaming\RayPack\Profiles\DEMO.rpprofile"
-ProjectName "\\rayflowshare\sources\Projects\7-Zip_6.5_MUL_1.0.0\7-Zip_6.5_MUL_1.0.0
-RayFlowPackageId 47110070815
-RayFlowServiceUser "UserName"
-RayFlowServicePassword "Password"
-RayFlowUrl "https://rayflow.raynet.de"
-RayFlowProjectId "3F2504E04F8941D39A0C0305E82C3301"
-WizardMode "expert"
```

Silent Command Line Switches

The thin executable `RpCmd.exe` has been added to the set of RayPack toolkit, ready to perform among others silent command line building, creating snapshots etc.. The tool is installed automatically along with the core application, and placed directly within the applications program directory, which is typically something like `C:\Program Files (x86)\RayPack\`.

Part of its functionality may require a valid license file to be executed, which makes it unavailable for usage within the scope of Standalone Repackager instances.

- New
  Creates a new blank RPP project or a blank MSI database

- Build
  Building and converting RCP/RPP projects and MSI databases

- Repackage
  Performs a fully unattended silent repackaging

- Validate
  Starts ICE validation of a specified file

- Snapshot
  Creating system snapshots (`.rcs` format)

- Edit
  Edits a given MSI, MST or RPP file by setting an MSI property or applying RPMST template.

- Patch
  Creates a patch between two MSI files

- Convert
  Converts 3rd party formats to RPP projects

- UpgradePackPoint
  Various tools to update local resources or upgrade PackPoint data

The following switches are deprecated as of RayPack 6.5, but remain in the product due to compatibility reasons. It is not recommended to use them anymore:

- Capture

Be aware:
All RayPack command line switches and parameters are case insensitive. This means that parameter `Repackage` is interpreted exactly the same as `rePACKAGE`, and a switch

---

New

Description

This switch instructs RayPack to create a new empty project (`.rpp`) or a new empty database (`.msi`).

Parameters

Parameter	Required	Description
`-Path <path>`	YES	The output path where the new project is to be created. The extension defines the project type. Currently, only `.rpp` and `.msi` extensions are supported. When an unsupported extension is provided, the `.rpp` suffix will be appended automatically.
`-Set <name>=<value>`	NO	Used to set the value of an MSI property in a new project/msi file. It can be used multiple times. Information about syntax and examples for this parameter can be found in this *section*.
`-Language <LCID>`	NO	A numeric value that represents the LCID code of the language to be used for the package. If not provided, a default value from the blank template will be used.
`-Platform <platform>`	NO	The platform that is to be used. If not provided, a default value from the blank template will be used. Possible values are Intel, Intel64 and x64. Custom templates can be defined according to the rules outlined in the section Managing default platforms and languages.
`-Schema <schema>`	NO	A numeric value that represents the MSI schema of the project. If not provided, a default value from the blank template will be used. The following values are supported: `100,110,120,200,300,301,400,405,500`

Example

```
RpCmd.exe New
-path C:\temp\newProject.rpp
-Set ProductVersion=1.2.3
-Set ProductName=NewProduct
-Set ALLUSERS=1
```

Build

Description

This switch instructs RayPack to perform a silent building of a project. Various number of input and output formats are supported. For example, you can use this command to:

- Build RPP project to an MSI format
- Rebuild an MSI package
- Build a virtual package from an MSI package
- etc.

> **Be aware:**
> In order to be able to generate virtual target package formats, the license of the parent RayPack application instance has to contain the optional Virtualization pack.

Parameters

Parameter	Required	Description
`-Format <format>`	YES	Output file format. The following values are supported: RCP, `MSI, RPP, SFT, APPV, XPF, DAT, MST` ○ RCP: RayPack Capture Project ○ RPP: RayPack Package Project ○ RPPX: RayPack MSIX Project ○ MSI: Windows Installer database ○ SFT: Microsoft App-V 4.6 ○ APPV: Microsoft App-V 5.0 ○ XPF: Symantec Workspace Virtualization package ○ DAT: ThinApp package ○ MST: Windows Installer transform ○ MSIX MSIX Package ○ VHD MSIX app attach ○ INTUNEWIN Intune Win32 Package ○ LAYPKG Citrix App Layering
`-Input <path>`	YES	Full path to the source file (supported file types: `*.msi`; `*.rpp`; and `*.rcp`). If the target format is MST, then only `*.msi` files are accepted as input paths.

Parameter	Required	Description
-Output <path>	YES	Full path to the output file. **Be aware:** If the target file (the `output` parameter) has an extension that is different than the one defined by the format type (the `format` parameter) the correct extension will be appended. Otherwise, the target file extension will be used.
-CabSplitting <size>	NO	**Only when building to *.msi format** Maximum CAB size in MB with a valid range between 1 and 2000. If the parameter is not given, the default value 2000 is used.
-Source <type>	NO	**Only when building to *.msi format** Type of resource location and compression option. The following options are supported: `CompressedExternal, CompressedInternal, UncompressedExternal`
-CabNaming <type>	NO	**Only when building to *.msi format** Defines the naming pattern for CAB files. If not specified, the value from the profile will be taken. The following values are supported: o `Data`: The names of new CAB files will be `Data1.CAB`, `Data2.CAB` etc. o `Name`: The names of new CAB files will be the same as the name of the MSI, for example `My Application.cab` o `Name83`: The names of new CAB files will be the same as the name of the MSI. If the MSI name is longer than 8 characters, the CAB name will be shortened, for example `MYAPPL12.cab` o Custom: The names of new CAB files have to be specified using the `customCabName` option.
-CustomCabName <name>	NO	**Only when building to *.msi format** Defines the custom pattern for CAB names. Only applicable if `cabNaming` is set to `Custom`
-Set <name>=<value>	NO	**Only when building to *.msi and *.rpp formats** Sets the value of an MSI property in the output file. Multiple instances are allowed to change more than one property. The name and value have to be

Parameter	Required	Description

		separated by an quality sign (=). The name of the property may not contain spaces. If the value contains spaces, the whole string should be surrounded by quotes, for example:
		
		`-set "Name=Value with spaces"`
		`-set Name=ValueWithoutSpaces`

`-PreviousVersion <path>`	NO	**Only when building to *.msi** Defines the path to a previous version of MSI, used for synchronization purposes and optimization of component structures and identifiers.
`-GenerateProductCode`	NO	**Only when building to *.msi and *.rpp format** Generates a new `ProductCode` for the output MSI file.
`-SignMsi`	NO	**Only when building to *.msi format** Signs the output MSI. The signature data is taken from the currently active RayPack settings profile.
`-TransformTemplate`	NO	**Only when building to *.mst format** Applies changes from RPMST definition to the new MST. This parameter is not required, if it is not provided then the default RPMST definition will be taken from your current profile.
`-UpdateFileInfo`	NO	**Only when building to *.msi format** Updates file information (size, version, language) of file resources available within the output file.

Example

Building an RCP project to MSI format with a set of commonly used options:

```
RpCmd.exe Build
        -Format "MSI"
        -Input "C:\RayPack\Projects\FileZilla\3.9.0.6\FileZillaClient-3_9_0_6.RCP"
        -Output "C:\RayPack\Projects\FileZilla\3.9.0.6\_MSI\FileZillaClient-3_9_0_
        -CabSplitting 1500
        -Source "CompressedExternal"
        -Set "ProductVersion=1.2.0"
        -Set "Manufacturer=The FileZilla Project"
        -GenerateProductCode
        -SignMsi
        -UpdateFileInfo
```

Building MST transform based on a selected MSI file, implementing changes from a given RPMST definition:

```
RpCmd.exe Build
          -Format "MST"
          -Input "C:\RayPack\Projects\FileZilla\3.9.0.6\_MSI\FileZillaClient-3_9_0
          -Output "C:\RayPack\Projects\FileZilla\3.9.0.6\_MST\FileZillaClient-3_9_(
          -TransformTemplate "C:\RayPack\PackPoint\Templates\default-template.rpmst
```

Parameters precedence

Build options are processed in the following order:

1. Command line build options
2. Project build options
3. Profile build options

This means, that any kind of build option passed in the command line (for example `-signMsi`) overrides signature settings present in the project or in the profile.

Repackage (Silent)

Description

This switch instructs RayPack to perform a silent repackaging of application and produces an RCP project. In order to build an MSI package from RCP project, use the Build command.

Parameters

Parameter	Required	Description
`-applicationPath <path>`	YES	Full path to the repackaged application. **Be aware:** The application has to support a silent mode in order to perform a fully silent repackaging. If additional parameters are required, they can be specify them via `applicationParameters` parameter. For example, for many legacy setups /q and /s switches are responsible for unattended installations.
`-projectName <path>`	YES	The name of the RCP project.
`-applicationParameters <params>`	NO	Command line parameters to be passed to the installer specified by the `applicationPath` parameter.
`-profile <path>`	NO	A full path to the profile that will be used during the repackaging
`-projectDir <path>`	NO	A full path to the RCP project.
`-targetDir <path>`	NO	An optional full path to the directory where the output RCP file will be saved. If the parameter is omitted, the default project folder will be used.
`-machineName`	NO	Name of virtual machine to be used for repackaging. This uses PackBot module, and the virtual machine must be already configured using Virtual Machines settings.

Example

```
RpCmd.exe Repackage
-ApplicationPath "C:\sources\install.msi"
-ProjectName "Text Editor 1.0" -ApplicationParameters "/q /e /serial=""123-456-789"""
```

Wrap

Description

This switch instructs RayPack to create a PowerShell wrapper for the given product.

Parameters

Parameter	Required	Description
`-Input <path>`	YES	The full path to MSI or EXE file.
`-Output <path>`	YES	The full path to the output directory where new files will be saved.
`-InstallArgs <arguments>`	NO	The command line arguments for installation, RayPack tries to determine it from the source file. **Be aware:** The setup must be able to install the product silently with the given command line. If this is not the case, then there is no use for the wrapper, because the user will still be prompted by original UI.
`-Language <language>`	NO	The language of the product. If not provided, RayPack tries to determine it from the source file.
`-Name <name>`	NO	The name of the product. If not provided, RayPack tries to determine it from the source file.
`-SupportingFiles <list>`	NO	Semicolon separated list of additional files or folders which are to be included. Source files / folders must be in the same folder tree as the main source file.
`-UninstallArgs <arguments>`	NO	The command line arguments for uninstallation. RayPack tries to determine it from the source file.
`-UninstallCommand <command>`	NO	Command to execute for installation (only for non-MSI setups). If left empty, the source file is called

Parameter	Required	Description
		for uninstallation command.
-Vendor	NO	The name of product vendor. If not provided, RayPack tries to determine it from the source file.
-Version	NO	The version of the product. If not provided, RayPack tries to determine it from the source file.

Example

```
RpCmd.exe Wrap
          -Input "C:\RayPack\Projects\FileZilla\3.9.0.6\FileZillaClient-3_9_0_6.EXE'
          -Output "C:\RayPack\Projects\FileZilla"
          -Name "FileZilla"
          -UninstallArgs "-s"
          -UninstallCommand "%programfiles%\FileZilla\uninst.exe"
```

Validate (Silent)

Description

Calling RayPack with this switch initiates a silent validation and puts out the results of the validation as a text file.

Parameters

Parameter	Required	Description
-Input `<name>`	YES	Path to the MSI / RPP file to be validated.
-Cub `<path>`	YES	Path to the CUB file with validation rules. This parameter can be defined more than once.
-Output `<mode>`	NO	Path to the file with validation results.
-MST `<params>`	NO	Path to the MST file. This parameter can be defined more than once.

Example

This command line will start RayPack, and initiate the silent ICE validation using full ICE set.

```
RpCmd.exe Validate
-Input "\\rayflowshare\sources\7-Zip_6.5_MUL_1.0.0\7-Zip_6.5_MUL_1.0.0.msi"
-Cub "C:\Program Files (x86)\RayPack\Resources\Validation\darice.cub"
-Output "\\rayflowshare\sources\7-Zip_6.5_MUL_1.0.0\7-Zip_6.5_MUL_1.0.0.ICE.txt"
```

Snapshot

Description

This switch instructs RayPack to create a snapshot of the current operating system.

Parameters

Parameter	Required	Description
`-Output <path>`	YES	Full path to the output snapshot file.
`-Profile <path>`	NO	Full path to the profile being used during snapshotting. If not provided, the last used profile will be used.

Example

```
RpCmd.exe Snapshot
-Output "C:\Snapshot\Snapshot.rcs"
```

Edit (Silent)

Description

This switch instructs RayPack to edit an RPP/MSI or MST file by applying a transform template (RPMST) and/or changing MSI properties. Additionally, this command can be used to create a new MST file.

Parameters

Parameter	Required	Description
`-Path <path>`	YES	Full path to the edited file. If the path has extension `.mst`, then the `-BaseMsi` parameter has to be specified as well.

Parameter	Required	Description
		Note: This parameter must be present. If the extension of the file pointed by this parameter is not `.mst` then the file is additionally checked for its presence.
`-Template <path>`	NO*	Full path to the RPMST template file. **Note:** This parameter is optional. However, when `-Path` parameter points to an `.mst` file, then this parameter can be omitted only if at least one instance of `-Set` parameter is used.
`-BaseMsi <path>`	NO*	This parameter specifies the base MSI for the MST transform. **Note:** This parameter must be used if and only if the new file extension is `.mst`.
`-Set <name>=<value>`	NO	Used to set the value of an MSI property in a new project/msi file. It can be used multiple times. Information about syntax and examples for this parameter can be found in this *section*.

Creating transform files using Template command

Because the `-Path` parameter can point to a non-existing MST file and the `-Template` parameter is optional, the command can be used to create new transforms. Below is an example of syntax:

```
RpCmd.exe Edit
-Path C:\temp\newTransform.mst
-BaseMsi "C:\temp\db.msi"
-Set "ProductName=My Product"
```

> **Be aware:**
> When `-Template` parameter is not specified, at least one `-Set` instance is required. Additionally, each Windows Installer Transform file has to contain at least one change as compared to the base MSI file.

Examples

```
RpCmd.exe Edit
-Path C:\temp\db.msi
-Template C:\templates\t1.rpmst
-Set "ProductName=My Product"

RpCmd.exe Edit
-Path C:\temp\myTransform.mst
-BaseMsi "C:\temp\db.msi"
-Template C:\templates\t1.rpmst
-Set "ProductName=My Product"
```

Patch

Description

This switch instructs RayPack to create a new MSP patch based on a difference between two MSI files.

Parameters

Parameter	Required	Description
`-MSI1 <path>`	YES	Full path to the original MSI being patched.
`-MSI2 <path>`	YES	Full path to the MSI containing the newest state.
`-Output <path>`	YES	Full path to the output file.
`-AllowRemoval`	NO	A boolean value determining whether the patch is removable. Support for this feature may be automatically disabled depending on the complexity of changes between two packages).
`-IncludeWholeFiles`	NO	A boolean value determining whether whole files are included instead of delta only.

Parameter	Required	Description

Example

```
RpCmd.exe Patch
-MSI1 <C:\temp\old.msi>
-MSI2 <c:\temp\new.msi>
-Output <c:\temp\patch.msp>
-AllowRemoval
```

Convert

Description

This switch instructs 6.5 to convert a specified project or a folder containing projects to the RPP format.

> **Note:**
> To use this command, certain prerequisite software has to be installed on the machine. Read the Converting 3rd party projects to RPP projects section for more details.

Parameters

Parameter	Required	Description
`-Source <path>`	YES	Full path to the source folder or source project file (for example `C:\SourceProjects` or `C:\SourceProjects\Project1.ism`). If the value of this parameter points to a folder, RayPack will scan its content and look for any `.ism` or `.wsi` files. If the value of this parameter points to a file, that file will be converted. To control inclusion of subfolders, use the `-recursive` switch. **Note:** If the value does not contain a valid and existing file or folder path, RayPack will convert nothing, treating the provided path as a non-existing folder.

Parameter	Required	Description
-Target <path>	YES	Full path to the output folder (for example `C:\ConvertedProjects`). RayPack will use the file name of converted project and save the converted `.rpp` file in the specified folder, using the name of the original file. If the folder does not exist, it will be created automatically.
-IgnoreErrors	NO	Determines the behavior of RayPack when any error during conversion is encountered. This parameter is optional and has no result if only one project is converted at once. Without this parameter, when two or more projects are converted at once and any errors is encountered during the conversion process, RayPack stops execution immediately, logs an error and return exit code `8`. By setting the `IgnoreErrors` flag, the process continues and always returns exit code `0`. The packages that failed are logged in the console window.
-Recursive	NO	If present, then the subfolders are also included in a recursive search for `.ism` or `.wsi` files.

Example

```
RpCmd.exe Convert
-Source C:\OldProjects
-Target c:\RppProjects
-Recursive
```

UpgradePackPoint

Description

This switch instructs RayPack to perform various upgrade tasks for PackPoint resources. Use this command to:

- Manually upgrade PackPoint created by previous version of RayPack
- Update the shared PackPoint with local resources and modifications
- Repair shared PackPoint files

Parameters

Parameter	Required	Description
`-UpgradeType <type>`	YES	Specifies the update mode. o `FromInstallation`: The shared PackPoint resources will be updated from the application installation (by default `C:\Program Files (x86)\RayPack\Resources\`). This is the default setting. o `FromLocal`: The shared PackPoint resources will be updated from the local profiles, exclusion lists, templates (by default `%appdata%\RayPack\`) o `FromPackPoint`: The local profiles, exclusion lists, templates etc. will be updated from the shared PackPoint location
`-Mode <mode>`	YES	Specifies the collision resolving mode. o `TargetWins:` In case of conflict between two values, the target value wins (nothing will be overwritten). o `SourceWins:` In case of conflict between two values, the source one wins (it will overwrite the target values) **Be aware:** This switch applies only when merging profiles and/or exclusion list. For other files, target resources are preserved, only missing ones are copied.

Example

```
RpCmd.exe UpgradePackPoint
-UpgradeType FromInstallation
```

Upgrade rules

The resources are updated according to the following rules

1. Missing files are copied from source to the target location
2. Certain XML resources are merged from source to the target location. In case of conflicts:
   o If `FromPackPoint` mode was used, then the application settings define which one from the source or the target is winning. To read how to change the PackPoint merging strategy, go to the PackPoint chapter.
   o For other types of PackPoint update, the target always wins.

Capture (Deprecated)

> **⚠ WARNING:**
> This switch is deprecated, and will no longer be supported by the `RayPack.exe` provided by future RayPack releases. Use Repackage switch instead.

Description

This deprecated switch instructs RayPack to perform a silent repackaging of application and produces an RCP project. In order to build an MSI package from RCP project, use the Build command.

Parameters

Parameter	Required	Description
`-ApplicationPath <path>`	YES	Full path to the repackaged application. **Be aware:** The application has to support a silent mode in order to perform a fully silent repackaging. If additional parameters are required, they can be specified via `applicationParameters` parameter. For example, to perform a silent installation of MSI package, the "`/qb!-`" parameter has to be specified.
`-ProjectName <path>`	YES	The name of the RCP project.
`-ApplicationParameters <params>`	NO	Command line parameters to be passed to the installer specified by the `ApplicationPath` parameter.

Parameter	Required	Description
-ProfilePath <path>	NO	A full path to the profile that will be used during the repackaging.
-TargetDir <path>	NO	An optional full path to the directory where the output RCP file will be saved. If the parameter is omitted, the default project folder will be used.

Example

```
RpCmd.exe Capture
-ApplicationPath "C:\sources\install.msi"
-ProjectName "Text Editor 1.0"
-ApplicationParameters="/qb ALLUSERS=2 SERIALNUMBER=123-456-789"
```

PowerShell Automation

Certain actions on MSI and RPP projects are exposed via the RayPack PowerShell module.

In order to get started, make sure that the module is imported. Invoke the following command in PowerShell terminal of your choice:

```
Import-Module "<RayPackInstallDir>\Libraries\Raynet.RayPack.Automation.dll"
```

You can show the available commands by invoking the following PowerShell code:

```
(Get-module Raynet.RayPack.Automation).ExportedCommands
```

Sample usage

The following script opens an RPP project, changes one of its properties and then builds MSI out of it.

```
$project = Open-Project -File "C:\demo\input.rpp";
[string]$msiPath = "C:\demo\output.msi";

try
{
    Set-Property $project -Name "ProductVersion" -Value "1.2.3";
    ConvertTo-Msi -Project $project -Target $msiPath -UpdateProductCode
$true -UpdateFileProperties $true -MediaLayout CompressedInternal | Out-
Null;
}
finally
{
```

```
        Close-Project -Project $project;
    }
```

Basic operations

Open-Project

Opens a project in MSI or RPP format.

Name	Type	Mandatory	Description
File	FileInfo	Yes	The file to open.

Returns an instance of `RayPackFile` type. You should keep the reference to the returned object in a variable,

Close-Project

Opens a project in MSI or RPP format.

Name	Type	Mandatory	Description
Project	RayPackFile	Yes	The file to open.
Commit	Switch	No	If set, all changes executed on a project will be saved. If you do not use this switch, the project is closed without saving your changes.
All	Switch	No	Instead of specifying a project (via `-Project` parameter), you can instead use this switch to close all projects in the current session.

This command let does not return any value.

Tables and properties

Remove-Row

Removes a row from RayPack project.

Name	Type	Mandatory	Description
Project	RayPackFile	Yes	The source project.
TableName	String	Yes	The name of a table to update. The row ID to be removed.
RowId	String	No	The identifier of the row to be removed. This switch is mutually exclusive with the -Row parameter.
Row	RayPackRow	No	The row to be removed. This switch is mutually exclusive with the -RowId parameter.
UpdateReferences	Switch	No	If set, referencing rows will be updated to reflect the removal of the row.
Force	Switch	No	If set, the action will continue even if the row does not exist. If unset, the removal of non-existing rows throws an exception.

This command let does not return any value.

Remove-Table

Removes a table from RayPack project.

Name	Type	Mandatory	Description
Project	RayPackFile	Yes	The source project.
TableName	String	Yes	The name of table to remove.

This command let does not return any value.

Set-Property

Sets a property in the RayPack project.

Name	Type	Mandatory	Description
Project	RayPackFile	Yes	The source project.
Name	String	Yes	The name of property to change.
Value	String	Yes	The new value of a property.

This command let does not return any value.

Remove-Property

Removes a property from RayPack project.

Name	Type	Mandatory	Description
Project	RayPackFile	Yes	The source project.
Name	String	Yes	The name of property to remove.
Force	Switch	No	If set, the action will continue even if the property does not exist. If unset, the removal of non-existing properties throws an exception.

This command let does not return any value.

Files

Import-File

Imports a file from the specified path to the RayPack project.

Name	Type	Mandatory	Description
Project	RayPackFile	Yes	The source project.
File	FileInfo	Yes	The name of a table to update. The row ID to be removed.

ComponentId	String	No*	The identifier of the component to which the file will be imported. This switch is mutually exclusive with switch – `DirectoryId`
FeatureId	String	No	The identifier of the feature to which the file will be imported. The component will be created automatically.
DirectoryId	String	No*	The identifier of the directory to which the file will be imported. The component will be created automatically. This switch is mutually exclusive with switches – `ComponentId`.
ForceOwnComponent	Switch	No	If set, the action will continue even if the row does not exist. If unset, the removal of non-existing rows throws an exception.

Returns a collection of added rows (`RayPackRow[]`)

Update-File

Updates a file within the RayPack project.

Name	Type	Mandatory	Description
Project	RayPackFile	Yes	The source project.
File	FileInfo	Yes	The name of a table to update. The row ID to be removed.

`FileID`	`String`	Yes	The file identifier of a file to be replaced.
`UpdateChecksum`	`Switch`	No	If set, the MSI checksum will be recalculated and written to the checksum table.
`UpdateAttributes`	`Switch`	No	If set, the attributes will be re-read from the source.

This command let does not return any value.

Transform and templates

Import-Template

Applies a RayPack template to the specified project.

Name	Type	Mandatory	Description
`Project`	`RayPackFile`	Yes	The source project.
`File`	`FileInfo`	Yes	The full path to the template file.

This command let does not return any value.

Import-Transform

Applies Windows Installer Transform to the specified project.

Name	Type	Mandatory	Description
`Project`	`RayPackFile`	Yes	The source project.
`File`	`FileInfo`	Yes	The full path to the transform file.
`SuppressedErrors`	`TransformErrors`	No	Optional errors to suppress: None, AddExistingRow, DelMissingRow, AddExistingTable, DelMissingTable, UpdateMissingRow, ChangeCodePage, ViewTransform.

This command let does not return any value.

Exporting and converting

ConvertTo-MSI

Converts the specified project into the MSI format. If the project is already an MSI, the file will be rebuilt using the new settings.

Name	Type	Mandatory	Description
Project	RayPackFile	Yes	The source project.
Target	FileInfo	No	The output file path where the MSI will be created.
OriginalMsi	FileInfo	No	The full path to the original MSI file. You should provide this path if your setup uses Linked folders.
Sign	Bool	No	A Boolean value indicating whether the output file should be digitally signed. If omitted, the value from the current profile will be taken.
CabNaming	CabNaming	No	The enumerable defining how the CAB file fill be named. If omitted, the value from the current profile will be taken.
MediaLayout	CompressionOptions	No	The option defining how to compress files. If omitted, the value from the current profile will be taken.

UpdateProductCode	Bool	No	A boolean value indicating whether the ProductCode should be updated. If omitted, the value from the current profile will be taken.
UpdateFileProperties	Bool	No	A boolean value indicating whether the file properties should be updated (like size, version). If omitted, the value from the current profile will be taken.
MaximumCabSize	Int	No	Defines the maximum size of a single CAB file. If omitted, the value from the current profile will be taken.
CustomCabName	String	No	Defines the custom name of a CAB file. If omitted, the value from the current profile will be taken.
BootStrapper	Bootstrapper	No	Defines the format of a bootstrapper (EXE, CMD, PowerShell). If omitted, the value from the current profile will be taken.

Returns an instance of `FileInfo` type, representing the built MSI file.

ConvertTo-MSIX

Converts the specified project into the MSIX format.

Name	Type	Mandatory	Description
Project	RayPackFile	Yes	The source project.
Target	FileInfo	No	The output file path where the MSIX will be created.
Sign	Bool	No	A Boolean value indicating whether the output file should be digitally signed. If omitted, the value from the current profile will be taken.

Returns an instance of `FileInfo` type, representing the built MSIX file.

ConvertTo-Intune

Converts the specified project into the MSIX format.

> **Tip:**
> This command let converts an MSI / RPP project to Intune. In order to convert any setup to Intune format, use New-Wrapper command let with parameter `-Format Intune`.

Name	Type	Mandatory	Description
Project	RayPackFile	Yes	The source project.
Target	FileInfo	No	The output file path where the Intune package will be created. If not provided, the package will be created in a temporary location which is returned by this command let.

Returns an instance of `FileInfo` type, representing the built Intune package.

Signing

Set-MsiSignature

Signs a Windows Installer file using a digital signature.

Name	Type	Mandatory	Description
`File`	`FileInfo`	Yes	The source file to be signed.
`Description`	`String`	No	The description of signature.
`CertificateType`	`CertificateTypes`	No	The type of certificate (default, PFX or Hardware).
`PfxFile`	`FileInfo`	No	The path to the .PFX file. This option is mutually exclusive with the `–HardwareCertificate` switch.
`CertificateFile`	`FileInfo`	No	The path to the digital certificate file.
`Password`	`SecureString`	No	The secure password for digital certificate.
`UseTimeServerUrl`	`Bool`	No	A boolean value indicating whether a timestamp sever should be used on signing.
`HardwareCertificate`	`String`	No	The hardware certificate name. This option is mutually exclusive with the `–PfxFile` switch.

Returns an instance of `FileInfo` type, representing the signed MSI file.

Wrapping

New-Wrapper

Wraps the specified project and creates a PowerShell (PSADT) based deployment package.

Name	Type	Mandatory	Description
`Format`	`WrapperFormat`	Yes	The type of the wrapper (`AppDeployToolkit` or `Intune`)
`Output`	`string`	Yes	The full path to the output file.
`Setup`	`FileInfo`	Yes	The full path to the input setup file.
`InstallArgs`	`string`	No	The arguments to be passed to the main installer. If left, auto-value will be taken from the installer.
`PackageLanguage`	`string`	No	The language of the product. If left, auto-value will be taken from the installer.
`PackageName`	`string`	No	The name of the product. If left, auto-value will be taken from the installer.
`PackageVendor`	`string`	No	The vendor of the product. If left, auto-value will be taken from the installer.
`PackageVersion`	`string`	No	The version of the product. If left, auto-value will be taken from the installer.
`RepairArgs`	`string`	No	The arguments used to repair the product. If left, auto-value will be taken from the installer (MSI).
`RepairCommand`	`string`	No	The arguments used to repair the product. If left, auto-value will be taken from the installer (MSI).
`SupportingFiles`	`string`	No	Path to a directory with extra files to be

			copied to the wrapped package.
UninstallArgs	string	No	The arguments used to uninstall the product. If left, auto-value will be taken from the installer (MSI).
UninstallCommand	string	No	The arguments used to uninstall the product. If left, auto-value will be taken from the installer (MSI).

Returns an instance of `FileInfo` type, representing the wrapped file (Intune) or directory (AppDeployToolkit)

Executable Bootstrapper Command Line Switches

Usage

```
setup.exe [/S] [/Uninstall] [/log] [/MSIPARAM=<params>]
```

The parameters can be combined. The only requirement is that the `/MSIPARAM` parameter (if used) has to always be at the end of the command string.

Parameters

/S (optional)
Unattended installation

Example:
```
        setup.exe /s
```

/Uninstall (optional)
Uninstalls the package

Example:
```
        setup.exe /Uninstall
```

> **Be aware:**
> When the product is uninstalled, dependencies are left on the system. For example, a package that installs itself and Visual C++ Redistributable package as its dependency will not uninstall the VC++ package when the uninstallation is triggered from

```
setup.exe.
```

/Log (optional)
Enables logging of setup activities. The logs are saved to `%LOCALAPPDATA%\RayPack\Setup`

Example:
```
setup.exe /Log
```

> **Be aware:**
> This switch enabled internal bootstrapper logging. Use it to see which items are installed, in which order, with which conditions and command line parameters. To enable msiexec logging for the MSI package, use the following switch:
> ```
> /MSIPARAM "/l*v "<path_to_log_file>""
> ```

/MSIPARAM "<params>"
Passes specified command line to the internal `msiexec.exe` call.

> **Note:**
> When used, this parameter has to be always the last one.

Example:
```
setup.exe /S /MSIPARAM "INSTALLDIR="C:\Program Files (x86)\MyApplication""
setup.exe /Uninstall /MSIPARAM "/L*V "C:\logs\log.txt""
```

The syntax of `<params>` value must be a valid `msiexec.exe` command line parameter. See the following link to get more information about the syntax.
https://msdn.microsoft.com/en-us/library/aa367988(v=vs.85).aspx

> **Be aware:**
> The outer quotation marks are required. The inner quotation marks do not have to be escaped.

Utilities and Standalone Tools

This chapter describes standalone tools and utilities available within RayPack installation:

- **App-V launcher**
  A tool to test App-V 4.6 and App-V 5.0 packages on a local machine

- **IIS Scanner**
  A tool to scan and record the properties of IIS Website for a further import purposes

- **ODBC Scanner**
  A tool to scan and record ODBC driver, translator, and data source entries.

App-V Launcher

This tool lets you test an App-V 4.6 or 5.0 package locally, before moving it to a deployment server. The tool can be found in the following location:

```
<RayPackinstalldir>\Tools\AppvLauncher\AppvLauncher.exe
```

for example

```
C:\Program Files (x86)\RayPack\Tools\AppvLauncher\AppvLauncher.exe
```

Prerequisites

- Microsoft .NET 4.0 must be installed on the machine where `AppvLauncher.exe` is running
- Microsoft Application Virtualization Client must be installed on the machine where `AppvLauncher.exe` is running
- File streaming must be enabled
- The tool is fully standalone, requires no other files and no RayPack license.

Usage

You can simply copy the file to any location containing App-V 4.6 or App-V 5.0 package and run it by double clicking its icon.

- If exactly one App-V 4.6 or 5.0 package could be found in the same folder where the tool is, that package will be launched
- If no package has been found in the same folder where the tool is, nothing will happen
- If more than one App-V 4.6 or App-V 5.0 package is found, then a package selector will be shown:
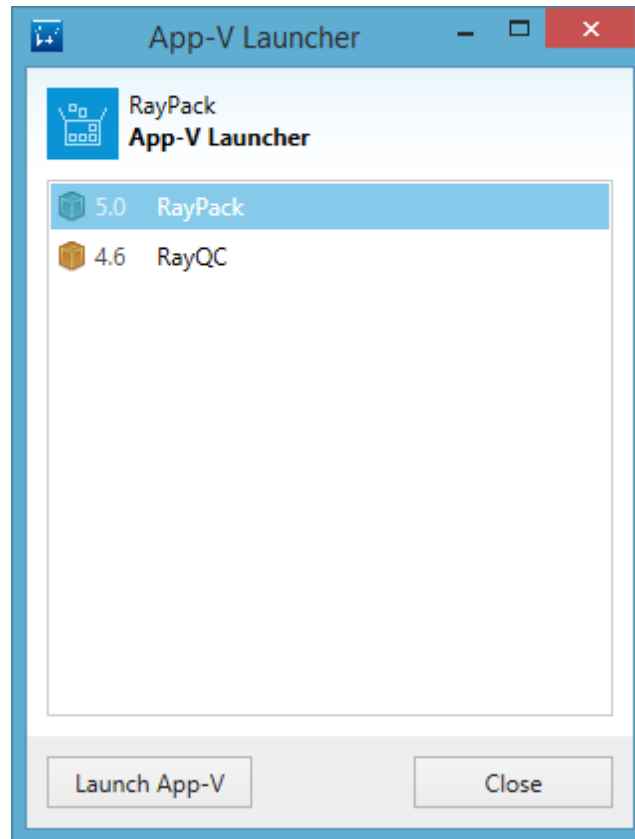
In order to start the currently selected package, click on *Launch App-V* button. The icon on the left side informs about the App-V version of package (App-V 4.6 or App-V 5.0). Press *Close* to close the App-V launcher.

Building App-V packages

App-V launcher is automatically copied to the output build folder, when *Copy App-V launcher to the output folder* option is selected in the profile settings.

IIS Scanner

This tool let you import IIS website data from a running IIS server. The tool can be found in the following location:

```
<RayPackInstalldir>\Tools\IISScanner\IISScanner.exe
```

for example

```
C:\Program Files (x86)\RayPack\Tools\IISScanner\IISScanner.exe
```

Prerequisites

- Microsoft .NET 4.0 must be installed on the machine where `IISScanner.exe` is running
- IIS Role must be enabled on the machine where `IISScanner.exe` is running
- The tool must be started as Administrator (the default manifestation requires it)
- The tool is fully standalone, requires no other files and no RayPack license.

Usage

Copy the `IISScanner.exe` file to a machine that hosts the required website(s). When the tool is started, a windows similar to the one below will be shown:
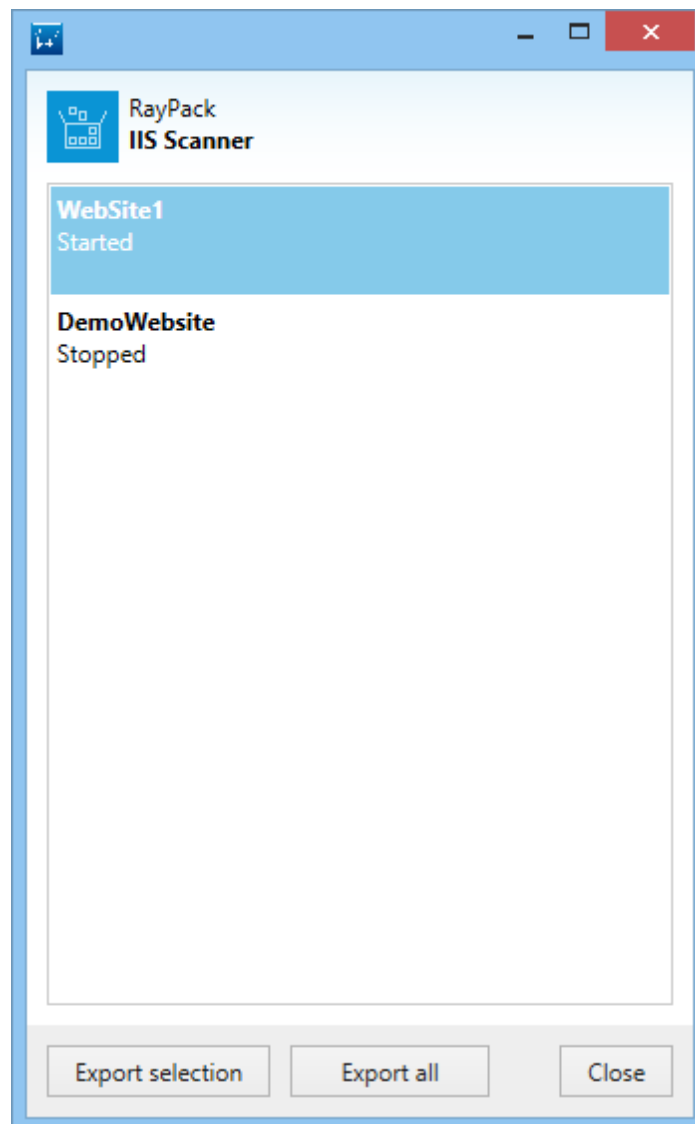
The list shows all websites found on the current IIS server. IIS 7, 7.5 and 8.5 are supported. Each entry represents a separate website, with name and status being displayed.

To export the required data, select websites to be exported, and then press *Export selection* to

select the target file where the data will be saved. You can also export all websites at once by clicking on *Export all* button. Click on *Close* to close the application.

Sample file

A following file may be produced by the IIS Scanner:

```xml
<?xml version="1.0"?>
<IISServerManager xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <Websites>
        <Website>
            <Id>1</Id>
            <Name>Default Web Site</Name>
            <Status>Stopped</Status>
            <ApplicationPoolName>DefaultAppPool</
ApplicationPoolName>
            <PhysicalPath>%SystemDrive%\inetpub\wwwroot</
PhysicalPath>
            <Bindings>
                <Binding>
                    <Ip>0.0.0.0</Ip>
                    <Port>80</Port>
                    <Host />
                    <Protocol>http</Protocol>
                    <BindingInformation>*:80:</
BindingInformation>
                </Binding>
            </Bindings>
            <LogFormat>W3c</LogFormat>
            <WebApplications>
                <WebApplication>
                    <Path>/WebApp1</Path>
                    <PhysicalPath>D:\web</PhysicalPath>
                    <ApplicationPoolName>DefaultAppPool</
ApplicationPoolName>
                    <VirtualDirectories />
                </WebApplication>
                <WebApplication>
                    <Path>/WebApp1/WebApp2</Path>
                    <PhysicalPath>D:\web\New folder</
PhysicalPath>
                    <ApplicationPoolName>DefaultAppPool</
ApplicationPoolName>
                    <VirtualDirectories />
                </WebApplication>
```

```
                        <WebApplication>
                                <Path>/WebApp1/WebApp2/WebApp3</Path>
                                <PhysicalPath>D:\web\New folder (2)</
PhysicalPath>
                                <ApplicationPoolName>DefaultAppPool</
ApplicationPoolName>
                                <VirtualDirectories />
                        </WebApplication>
                </WebApplications>
        </Website>
    </Websites>
    <ApplicationPools>
        <ApplicationPool>
                <Name>DefaultAppPool</Name>
                <ManagedRuntimeVersion>v4.0</ManagedRuntimeVersion>
                <ManagedPipelineMode>Integrated</
ManagedPipelineMode>
                <Enable32BitAppOnWin64>false</
Enable32BitAppOnWin64>
                <QueueLength>1000</QueueLength>
                <Limit>0</Limit>
                <LimitAction>NoAction</LimitAction>
                <ResetInterval />
                <IdentityType>ApplicationPoolIdentity</
IdentityType>
                <IdleTimeout />
                <MaxProcesses>1</MaxProcesses>
        </ApplicationPool>
    </ApplicationPools>
</IISServerManager>
```

ODBC Scanner

With the ODBC scanner the ODBC entries of a target computer can be scanned. The tool can be found in the following location:

```
<RayPackInstalldir>\Tools\OdbcScanner\OdbcScanner.exe
```

for example

```
C:\Program Files (x86)\RayPack\Tools\OdbcScanner\OdbcScanner.exe
```

Prerequisites

- Microsoft .NET 4.0 must be installed on the machine where `OdbcScanner.exe` is running
- The tool is fully standalone, requires no other files, and no RayPack license.

Usage

Copy the `OdbcScanner.exe` file to the target system. When the tool is started, a windows similar to the one below will be shown:
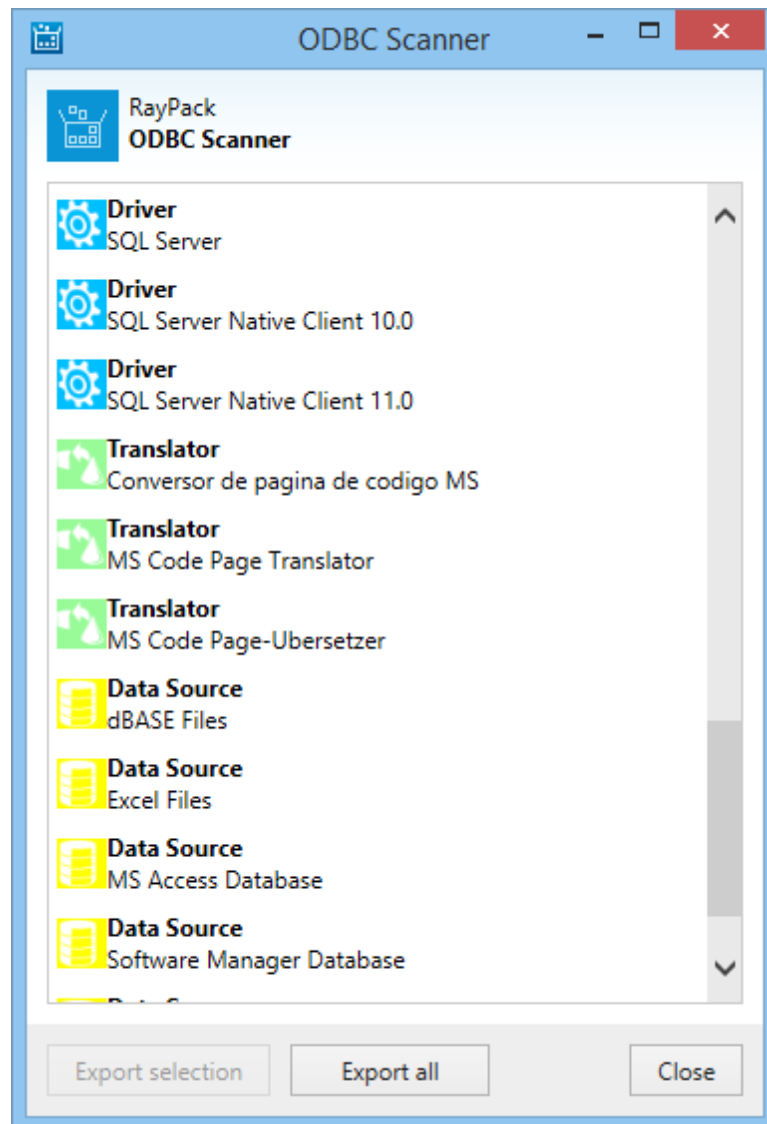
The list shows all entries that can be found on the target machine under `Control Panel\All Control Panel Items\Administrative Tools\ODBC Data Sources (32-bit)` and `Control Panel\All Control Panel Items\Administrative Tools\ODBC Data Sources (64-bit)`.
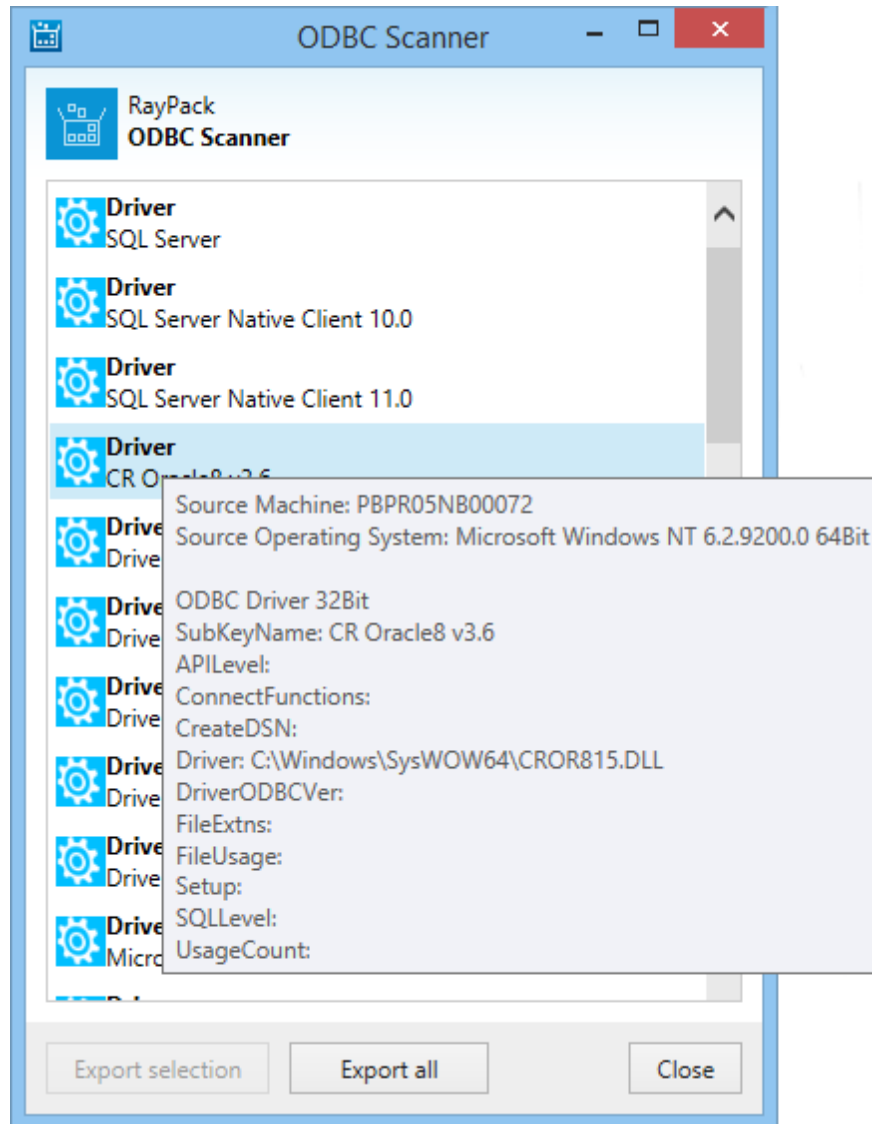
Each entry represents a separate driver, translator, or data source. More information on an entry will be shown when the mouse pointer is hovering on the respective entry. See screenshot above.

To export all entries, press the **Export selection** button and select the target destination to where the data will be saved. It is also possible to export only a selection of entries. To export a selection of entries, select the entries by clicking on them. An entry can be deselected by clicking on it again. After all entries that are meant to be exported are marked, press the **Export selection** button. Select the folder to where the data will be saved.

The data will be saved as RayPack ODBC file (`.rpodbc`).

Portable Repackager

The portable repackager is a general-purpose, standalone, and license-free executable which can be used to:

- Snapshot the current state of the machine.
- Compare two snapshots to produce RCP files.

The tool `rps.exe` can be found in subfolder `Tools/Repackaging`.

Creating Snapshots with RPS.EXE

Syntax:
```
rps.exe snapshot -snap <snap_file> [DRIVE1: [DRIVE2: ....]] [HKLM|HKCU|HKCR] [-permiss
```

Parameters:
`-snap <snap_file>`
(required) The full path to a snapshot file, for example `C:\snapshots\snapshot.rcs`.

`DRIVE1: [DRIVE2: ....]`
(optional) A list of drives to scan separated by space, for example `C: D: E:`. By default nothing is scanned.

`HKLM|HKCU|HKCR`
(optional) A list of registry root nodes to scan, separated by space, for example `HKLM HKCU`. By default nothing is scanned.

`-permissions <True|False>`
(optional) A boolean value determining whether permissions are scanned. Default: `False`.

`-services <True|False>`
(optional) A boolean value determining whether services are scanned. Default: `False`.

`-ini <True|False>`
(optional) A boolean value determining whether INI files are scanned. Default: `False`.

Examples
- `rps.exe snapshot -snap C:\temp\snapshot1.rcs C: HKLM`
- `rps.exe snapshot -snap C:\temp\snapshot1.rcs C: D: HKLM HKCU -permissions True`
- `rps.exe snapshot -snap C:\temp\snapshot1.rcs C: -permissions True -services True -in`

Comparing Snapshots

Syntax:
```
rps.exe compare <snapshot1> <snapshot2> <output_file> [-copyFiles <True|False>] [-disc
```

Parameters:

`<snapshot1>`
(required) The full path to the first snapshot file, for example `C:\snapshots\snapshot1.rcs`.

`<snapshot2>`
(required) The full path to the second snapshot file, for example `C:\snapshots\snapshot2.rcs`.

`<output_file>`
(required) The full path to the output file, for example `C:\snapshots\comparisonResult.rcp`.

`-discardExcluded <True|False>`
(optional) Determines whether to discard excluded items. Default: `<False>`.

`-copyFiles <True|False>`
(optional) Determines whether new or changed files are copied. Default: `<False>`.

Examples
- `rps.exe compare "C:\temp\snapshot1.rcs" "C:\temp\snapshot2.rcs" "C:\temp\result.rcp"`
- `rps.exe compare "C:\temp\snapshot1.rcs" "C:\temp\snapshot2.rcs" "C:\temp\result.rcp"`

MSI Diff Tool

This is a standalone version of visual comparer, normaly available from the FILE menu for opened projects (see Comparing Projects and MSI Files).

To start the tool, navigate to the installation directory, and double click `raypack-diff.exe`. If started without any command line, it will start with a blank screen, allowing the user to select old and new version for the comparison.

You can automatically load a comparison of two projects by passing two command line arguments - paths to the old and to the new package, for example:

```
raypack-diff.exe c:\src\version1.msi c:\src\version2.msi
```

The tool supports transform files too. To configure them, append one or more paths to MST files after the MSI file, for example (note: line-breaks added for a better readibility):

```
raypack-diff.exe
   c:\src\version1.msi
   c:\src\version1a.mst
   c:\src\version1b.mst
   c:\src\version2.msi
   c:\src\version2a.mst
   c:\src\version2b.mst
```

Advanced Topics

Ignoring Certain Resources When Repackaging

Certain folders and registries are not scanned at all to improve the performance of PackRecorder scanning engine. RayPack contains an out-of-the-box list of such places, which should never be scanned regardless of what exclusion lists are defining. Most importantly, these resources are not part of the RCP project (unlike excluded ignored resources, which - although being excluded - are still copied to the project to provide an easy way to correct mistakes at later point of time).

The ignored resource list is also a way for advanced packagers to instruct RayPack which registry keys and folders should be ignored, should the exclusion of the full node/drive be not an option. To edit these resources, use any text editor and open the `RayPackIgnoredResources.rpx` file from the Resources folder which is located in the main installation folder (for example `C:\Program Files (x86)\RayPack Studio\RayPack\Resources\RayPackIgnoredResources.rpx`). The file has a following syntax (sample):

```xml
<?xml version="1.0"?>
<IgnoredResources Version="1.0.1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <RegistryEntries>
        <!-- Registry Hives or complete keys are supported -->
        <!-- Use the short form for defining the registry hives...
        HKLM, HKCU,HKCR etc... -->
        <Registry><![CDATA["HKLM\COMPONENTS"]]></Registry>
        <Registry><![CDATA["HKLM\Schema"]]></Registry>
        ...
        ...
        ...
    </RegistryEntries>
    <!-- Only directories are supported. Any sub-directories will also be ignored...
    USE WITH CAUTION! -->
    <FileSystemEntries>
        <FileSystem><![CDATA["%LOCALAPPDATA%\RayPack"]]></FileSystem>
        <FileSystem><![CDATA["%LOCALAPPDATA%\RayNet"]]></FileSystem>
        ...
        ...
        ...
    </FileSystemEntries>
</IgnoredResources>
```

Environment variables are supported to avoid hardcoded paths.

For example, if a folder `C:\Sources` is known to contain a data which should never be scanned, a

following entry may exclude it from scanning:

```
<FileSystem><![CDATA["%HOMEDRIVE%\Sources"]]></FileSystem>
```

Authoring Large Packages

Due to Windows Installer standard limitations, the maximum number of files in a single Windows Installer package is limited to 32767 files. If a package deployed by a Windows Installer database has more files, a different (non-standard) table schema is required.

⚠️ **WARNING**
Changing the table schema manually is generally not recommended for standard tables, as the resulting package may break the functionality or not work at all.

RayPack can automatically handle the schema updates when working with RCP or RPP projects. During the build process of an MSI-based output file (RPP/MSI), depending on number of files in the project a custom schema may be applied, enabling authoring large MSI packages. Specifically, the following changes may be introduced:

- The type of the column `Sequence` in the `File` table changed from i2 (short) to i4 (long int)
- The type of the column `LastSequence` in the `Media` table changed from i2 (short) to i4 (long int)
- The maximum values for columns `File` > `Sequence` and `Media` > `LastSequence` in the validation table changed from 32767 to 2147483647

The process is fully automated and requires no further action. If the number of files in the output MSI would be less than 32768, no change to the schema will be attempted.

⚠️ **WARNING**
The produced MSI will have a non-standard definition of the crucial tables `File` and `Media`. Patching or transforming MSI/MST having different schemata may fail due to the Windows Installer limitations. To produce a patch or transform between two MSI packages, the schema of all affected tables must be the same.

📎 **Note:**
This schema change lets you add more than 32767 files to a single MSI project by setting a new limit to a value of 2'147'483'647. No single package will ever reach this limit and remain usable and operational.

Editing existing MSI packages with standard schema

Due to compatibility reasons, RayPack does not change the schema of existing MSI packages when using SAVE/SAVE AS functionality. If you open a package suited for a small number of files (under 32768) and add new files resulting in overflow of the standard limit, RayPack shows a warning that the limit is reached. You will be able to ignore the warning and continue the

process. However, after the files are imported, saving such an MSI may fail. In order to persist your changes, use the BUILD option instead of SAVE AS, since building recreates the schema and ensures that the number of files can exceed the standard limit of 32767.

> **Note:**
> It is not possible to save the MST transform with a different schema than the underlying base MSI. Applying such transform will fail due to the Windows Installer limitations.

To force RayPack to always use the large package schema and bypass the original MSI limitations

1. Launch RayPack

2. Click on FILE and select the Options item from the menu column on the left.

3. Open the general tab and look-up the Windows Installer Template section

4. Use the browse button and select the `Blank_large.msi` template file

5. Save the settings by clicking OK

The procedure outlined above has to be repeated for each RayPack settings profile that requires adjustment.

After saving the changes, all subsequent projects and databases created by RayPack will be able to handle more than 32767 files without additional warnings.

> **Note:**
> Opening existing packages with an unadjusted file limit will still show a warning when more than 32767 files are attempted to be imported.

To revert to the original template

In order to revert to the original standard schema, simply repeat the steps described above, but point to a previous template file (for example `Blank.msi`). After saving the changes, all subsequent projects and databases created by RayPack will not be able to handle more than 32767 files.

Patching, transforming and merging of adjusted databases

The following functions of the Windows Installer technology heavily rely on the installer schema:
- MST transforms
- MSP patches
- Merge modules

Due to Windows Installer limitations, changing the schema may leave the following functions not working for adjusted packages:

- Patching between two installer databases having different schemat
- Creating MST transforms between two installer databases having different schemat
- Merging a merge module having a different schema than the target installer database

> **Note:**
> Projects and databases created before changing the template will remain their original schema, even if they are opened and saved again.

Customizing Predefined Folders

> **Be aware:**
> This is an advanced topic requiring advanced knowledge of Windows Installer. Make a backup of a configuration before you make any changes to any file.

The default choice of predefined folders can be customized by editing an XML file. In order to do it, navigate to the file `<INSTALLDIR>\Resources\Folders\Default.xml` where `<INSTALLDIR>` is the folder where RayPack has been installed, for example `C:\Program Files (x86)\RayPack\`.

The file has an XML-based syntax:

```xml
<?xml version="1.0" encoding="utf-8" ?>

<Default>
    <Folder Id="AdminToolsFolder" TargetId="TARGETDIR"
DefaultDir=".:ADMINT~1|AdminTools" />
    <Folder Id="AppDataFolder" TargetId="TARGETDIR"
DefaultDir=".:APPLIC~1|ApplicationData" />
    <Folder Id="CommonAppDataFolder" TargetId="TARGETDIR"
DefaultDir=".:COMMON~1|CommonAppData" />
    <Folder Id="CommonFiles64Folder"
TargetId="ProgramFiles64Folder" DefaultDir=".:Common64" />
    <Folder Id="CommonFilesFolder" TargetId="ProgramFilesFolder"
DefaultDir=".:Common" />
    <Folder Id="DesktopFolder" TargetId="TARGETDIR"
DefaultDir=".:Desktop" />
    <Folder Id="MyFolder" TargetId="DesktopFolder"
DefaultDir="Subfolder" />
    ...
</Default>
```

Each entry contains an identifier (`Id`, corresponding to the `Directory` column in the `Directory` table), parent identifier (`TargetId`, corresponding to the `Directory_Parent` column in the `Directory` table) and the displayed name (`DefaultDir`, corresponding to the `DefaultDir` column in the `Directory` table).

In order to define a new custom directory that points to `%ProgramFiles%\Folder1 \Folder with a long name`, the following entries are necessary:

```
        <Folder Id="Folder" TargetId="ProgramFilesFolder"
DefaultDir="Folder" />
        <Folder Id="Folder1" TargetId="Folder"
DefaultDir="FOLDERWI~1|Folder with a long name" />
```

Note: this structure must resolve to a valid MSI Directory structure. When customizing these fields, it is recommended to create necessary folders in a blank PackDesigner project manually, and then recreate the structure in the XML file.

More information about the `Directory` table and `DefaultDir` column can be found here: https://msdn.microsoft.com/pl-pl/library/aa368295(v=vs.85).aspx

Adjusting Condition Snippets

Snippets displayed in the **Condition builder** are fully customizable. To add new ones or edit/remove existing ones, open the XML file in any XML editor:

```
<PackPointDir>\Conditions\Snippets.xml
```

By default, this location is `C:\RayPack\PackPoint\Conditions\Snippets.xml`
Below is a short excerpt of the file content:

```
<?xml version="1.0" encoding="utf-8" ?>
<Snippets>
  <Snippet DisplayName="Product is already installed?">
    <Condition>Installed</Condition>
  </Snippet>
  <Snippet DisplayName="Product is not yet installed?">
    <Condition>NOT Installed</Condition>
  </Snippet>
  <Snippet DisplayName="Product is being removed?">
    <Condition>REMOVE ~= "ALL"</Condition>
  </Snippet>
  <Snippet DisplayName="Windows 10 or newer?">
    <Condition>VersionNT &gt;= 603</Condition>
  </Snippet>
  <!-- etc. -->
</Snippets>
```

Using the template, add new elements to the Snippets collection. RayPack has to be restarted in order to apply the changes.

Note:
Some characters are not allowed in XML files. Escape them using hexadecimal code (for example > instead of >, & instead of &) or use the CDATA syntax to mark the

unescaped areas.

Adjusting the Default Template

An MSI template is nothing more than an empty MSI file. This template can be used to create a MSI based project that contains pre-defined information that will be applied during the creation of all MSI based files created with RayPack. This ensures a uniform (base) for all transforms created.

As an example, the MSI template can contain predefined properties and predefined Summary Information. The templates themselves can be edited with RayPack. Only one template can be chosen at any one time, although it is possible to create more than one template. The location of the template to be used is set using the profile editor here.

The standard template applied to the process of new project creation in PackDesigner (or for the export of package information from rcp to rpp) is documented within the Default MSI template tables help topic.

> **Tip:**
> Please refer to the Authoring large packages section for further details regarding the default MSI templates that are delivered along with the RayPack application resources.

Using GUID placeholders in templates

When adding components to the template, it is important to ensure that the package-specific components get new unique identifiers so that packages created from the same template can coexist on the same machine. For example, a component containing the registry branding should always have an unique GUID, because the branding information has to be installed for each package separately.

In order to control the automatic GUID generation, RayPack recognizes special character "*" (asterisk) present in `Component` table, column `ComponentId`. Asterisks are are replaced with unique identifiers when a new project is created from a template.

> **Be aware:**
> RayPack does not generate the identifiers for all components automatically, because in some situations leaving hardcoded GUIDs may be desired. For example this may be a case if the component present in the package template belongs to a Merge Module or is by any other mean common and has to be shared accross packages.

Default MSI Template Tables

The default profile for RayPack users applies a standard template for MSI and RPP generation. The following overview provides a list of automatically prepared, added, or added and pre-populated

tables, named in alphabetical order and linked to their specific MSI Standard Table Reference pages.

Depending on user activity in the wizard guided views of Visual Designer and Advanced Mode, PackDesgner automatically adds prepared tables to the active database tables stock when required. For example, if a user adds service manipulation items to a project, RayPack automatically checks the availability of the required database tables (e. g. ServiceInstall or ServiceControl) and adds them to the pool of active tables if necessary.

Prepared tables

- AppId
- BBControl
- Billboard
- BindImage
- CCPSearch
- Class
- CompLocator
- Complus
- Condition
- CreateFolder
- DrLocator
- DuplicateFile
- Environment
- Extension
- Font
- IniFile
- IniLocator
- IsolatedComponent
- LaunchCondition
- LockPermissions
- MIME
- ModuleSignature
- MoveFile
- MsiAssembly
- MsiAssemblyName
- MsiDigitalCertificate
- MsiDigitalSignature
- MsiFileHash
- ODBCAttribute
- ODBCDataSource
- ODBCDriver

- ODBCSourceAttribute
- ODBCTranslator
- PatchPackage
- ProgId
- PublishComponent
- RemoveIniFile
- RemoveRegistry
- ReserveCost
- SelfReg
- ServiceControl
- ServiceInstall
- Shortcut
- TypeLib
- Verb

Added tables

- AdvtUISequence
- AppSearch
- CheckBox
- ComboBox
- Component
- CustomAction
- Feature
- FeatureComponents
- File
- ListBox
- ListView
- Media
- Patch
- Registry
- RegLocator
- RemoveFile
- Signature
- Upgrade

Pre-populated tables

- _Validation
- ActionText
- AdminExecuteSequence

- AdminUISequence
- AdvtExecuteSequence
- Binary
- Control
- ControlCondition
- ControlEvent
- Dialog
- Directory
- Error
- EventMapping
- Icon
- InstallExecuteSequence
- InstallUISequence
- Property
- RadioButton
- TextStyle
- UIText

RPMST Templates

A Transform Template is nothing more than a simple XML file, with the file extension `.rpmst`. This template can be used to create a transform file that contains predefined information which will be applied during the creation of all transform files created with RayPack. This ensures a uniform base for all transforms created.

Transform Template structure

Below is an abbreviated example of a transform template. Basically using this template, every transform file that uses it will contain:

A top level feature named "`Demo_Feature`", this feature will have a component named "`Demo_Branding`" assigned to it. This component in turn will contain three resources of the type registry. The properties `ALLUSERS, ARPNOMODIFY, ARPNOREMOVE, ARPNOREPAIR, REBOOT, RebootYesNo, REBOOTPROMPT, ROOTDRIVE` will be assigned default values. The feature and its component will only be installed in the default installation directory if the feature is not present on the system or a reinstall (repair) is initiated. The summary information stream properties `Author, Comments, Subject, Title` and `Keywords` are also set to default / dynamic values. Notice that any special characters **must be** escaped. For example, see the text **<PackagerInitials>** and how it has been escaped using the less-than and greater-than characters.

```
<?xml version="1.0" encoding="UTF-8"?>
<Settings>
  <!-- This is demonstration of transform template configuration.
  Any tables can be added - just make sure that any tables referenced here a
  also present in source(vendor) MSI  -->
```

```
    <SIS Author="&lt;PackagerInitials&gt;" Comments="Created: $TIME$ - $DATE$"
         Subject="" Title="$PROJECTNAME$" Keywords="MST Template 1.0.0" />
  <Tables>
    <Table Name="Feature">
      <Row Feature="Demo_Feature" Feature_Parent="" Title="Demo_Branding"
           Description="Branding Feature" Display="0" Level="1"
           Directory_="INSTALLDIR" Attributes="16"/>
    </Table>
    <Table Name="Component">
      <Row Component="Demo_Branding" ComponentId="$GUID$"
           Directory_="INSTALLDIR" Attributes="4"
           Condition="NOT Installed OR REINSTALL"
           KeyPath="Demo_registry1"/>
    </Table>
    <Table Name="FeatureComponents">
      <Row Feature_="Demo_Branding" Component_="Demo_Branding"/>
    </Table>
    <Table Name="Registry">
        <Row Registry="Demo_registry1" Root="2"
             Key="Software\Demo\Packages\[ProductName]\[ProductVersion]"
             Name="*" Value=""
             Component_="Demo_Branding"/>
        <Row Registry="Demo_registry2" Root="2"
             Key="Software\Demo\Packages\[ProductName]\[ProductVersion]"
             Name="-" Value=""
             Component_="Demo_Branding"/>
        <Row Registry="Demo_registry3" Root="2"
             Key="Software\Demo\Packages\[ProductName]\[ProductVersion]"
             Name="Application Name" Value="[ProductName]"
             Component_="Demo_Branding"/>
    </Table>
    <Table Name="Property">
      <Row Property="ALLUSERS" Value="1"/>
      <Row Property="ARPNOMODIFY" Value="1"/>
      <Row Property="ARPNOREMOVE" Value="1"/>
      <Row Property="ARPNOREPAIR" Value="1"/>
      <Row Property="REBOOT" Value="ReallySuppress"/>
      <Row Property="RebootYesNo" Value="No"/>
      <Row Property="REBOOTPROMPT" Value="S"/>
      <Row Property="ROOTDRIVE" Value="C:\"/>
    </Table>
  </Tables>
</Settings>
```

Settings element

The `<Settings>` element has no attributes and has two child elements, the `<SIS>` and the `<Tables>` elements. While theoretically possible, both child elements could be empty. This would make the use of a template nonsensical.

SIS element

The `<SIS>` element can contain attributes that are/can be present in the Summary Information

Stream, provided that they can be represented as a string value. It is recommend that only the SIS properties shown in the example be set using the transform template.

The following attributes of the `SIS` element are supported and have direct mapping to one of the supported summary properties (see https://msdn.microsoft.com/en-us/library/aa372049(v=vs.85).aspx for a complete list):

- `Author`
- `CharacterCount`
- `Codepage`
- `Comments`
- `CreateTimeDate`
- `LastPrinted`
- `LastSavedTimeDate`
- `CreatingApplication`
- `Keywords`
- `LastSavedby`
- `PageCount`
- `RevisionNumber`
- `Security`
- `Subject`
- `Template`
- `Title`
- `WordCount`

The following extra attributes have no direct translation to SIS attributes:

- `DatabaseCodepage`
  A numeric value representing the code page of the whole database (not to be confused with SIS code page and with `ProductLanguage`).

- `Languages`
  A list of comma separated numeric values representing supported languages (in LCID format)

- `Platform`
  A value representing the package target platform.

> **Note:**
> `Languages` and `Platform` values can be combined together and represented as a single value of `Template` attribute separated by a semicolon, for example `<SIS Template="Intel64;0,1031"` …

Tables element

The `<Tables>` element has no attributes and 1 to n child `<Table>` elements.

Table element

The `<Table>` element has one attribute and 1 to n child `<Row>` elements. The Name attribute is

case sensitive and must be unique.

The following example would signify that the Feature table is to be edited.

```
<Table Name="Feature">
...
...
</Table>
```

Row element

The `<Row>` element may contain multiple attributes. The attributes correspond with the column names (case sensitive) of the MSI table that is the Name attribute of the parent `<Table>` element.

```
<Table Name="Feature">
    <Row Feature="Demo_Feature"
        Feature_Parent="" Title="Demo_Branding"
        Description="Branding Feature"
        Display="0"
        Level="1"
        Directory_="INSTALLDIR"
        Attributes="16"/>
...
...
</Table>
```

Please ensure that all columns are set with values. If a column entry is to be set to empty, use an empty string value `=""`.

- If the row template is used to add a new row, it is recommended to specify all cell values. At minimum, the columns required by the MSI schema have to be specified.
- If the row template is used to edit an existing row, at minimum the columns forming the Primary Key have to be specified plus the columns to be updated.

RowUpdate element

The `<RowUpdate>` element has a similar behavior to the `<Row>` element.
The only difference is that the values specified by this node will be inserted only if the row with a given ID already exist. For example, the following syntax:

```
<Table Name="Feature">
    <RowUpdate Feature="Demo_Feature" Description="Updated branding Feature" /
</Table>
```

updates the description of a feature `Demo_Feature` to "Updated branding Feature". If the feature does not exist, no action is done.

Please ensure that all required columns including columns forming table's Primary Key are set with values.

Note:

As one can see, theoretically any MSI table can be populated in the transform file using the transform template. The only exception to this are streams in the Binary and Icon tables which have to be represented using a special RPMST syntax.

Replaceable Tokens

The transform template also supports tokens that can be replaced during the creation of a transform file. Generally a token can be identified as a string entity contained within dollar characters. The following tokens are supported at present:

- **`$TIME$`**
  The current time (`HH:mm:ss`)

- **`$DATE$`**
  The current date (`dd/MM/yy`)

- **`$GUID$`**
  A new GUID (uppercased, enclosed in curly braces)

- **`$MACHINENAME$`**
  The current machine name

- **`$PROJECTNAME$`**
  The current project name

- **`$USERNAME$`**
  The current user name

For example, to add a custom properties `ProjectCreatedOn` and `ProjectCreatedBy` which should by default be populated with the current date, time and creator of the project, the following snippet can be used:

```
<Table Name="Property">
      <Row
          Property="ProjectCreatedOn"
          Value="$DATE$ $TIME$" />
      <Row
          Property="ProjectCreatedBy"
          Value="$USERNAME$ on $MACHINENAME$" />
</Table>
```

Binary streams

RPMST syntax supports stream cells. The stream has to be represented as a base64-encoded string, representing the binary content of the stream. It is recommended to simply add a stream using RayPack and then export the RPMST template rather than creating binary values manually.

Below is an example of how to define a stream in the template:

```
<Settings>
  <Tables>
    <Table Name="Binary">
      <RowUpdate Name="Binary1" Data="MzNERC0zQjgyLTAxQjEtMzNFMC02MEQy" />
    </Table>
  </Tables>
</Settings>
```

> **Be aware:**
> Stream cells do not support replaceable tokens.

Real-life sample: ActiveSetup

The snippet below can be used to automatically add required Active Setup entries:

```xml
<Settings>
  <Tables>
    <Table Name="Feature">
      <Row Feature="ActiveSetup" Title="Active Setup" Description="A feature for Activ
      components" Display="0" Level="1" Attributes="16" />
    </Table>
    <Table Name="Registry">
      <Row
        Registry="ActiveSetup_Default"
        Root="2"
        Key="SOFTWARE\Microsoft\Active Setup\Installed Components\[ProductCode]"
        Value="[ProductName] [ProductVersion]"
        Component_="ActiveSetup" />
      <Row
        Registry="ActiveSetup_Version"
        Root="2"
        Key="SOFTWARE\Microsoft\Active Setup\Installed Components\[ProductCode]"
        Name="Version"
        Value="1,0,0,0"
        Component_="ActiveSetup" />
      <Row
        Registry="ActiveSetup_StubPath"
        Root="2"
        Key="SOFTWARE\Microsoft\Active Setup\Installed Components\[ProductCode]"
        Name="StubPath"
        Value="msiexec.exe /fpums [ProductCode] /qn "
        Component_="ActiveSetup" />
    </Table>
    <Table Name="Component">
      <Row
        Component="ActiveSetup"
        ComponentId="*"
        Directory_="TARGETDIR"
        Attributes="4"
        KeyPath="ActiveSetup_StubPath" />
    </Table>
    <Table Name="FeatureComponents">
      <Row
        Feature_="ActiveSetup"
        Component_="ActiveSetup" />
    </Table>
  </Tables>
</Settings>
```

> **Note:**
> This and a few more samples are available by default in the following folder:
> `<PackPoint>\Templates\Custom`.

Custom PackWrapper Templates

It is possible to define custom templates for the PackWrapper module, for example to:

- Add new custom functions
- Change the banner (logo)
- Define custom actions
- And for other purposes

To customize the default toolkit, perform the following steps:

1. Make a copy of the following folder (so that the original template remains unchanged and it is always possible to revert back to it):
   `<PackPointFolder>\Resources\Wrappers\PSAppDeploymentToolkit`

2. Go to Project settings and change the folder path to the new location

Once saved, all wrapper created with the currently selected profile will use the new folder as the template. In the new folder, it is possible to freely customize the wrapper, specifically:

- Replace banner image (`.png`) with a custom image
- Add any custom files with custom functions (they will be added automatically to resulting wrappers)
- Edit the content of existing files, add new functions, or extend the existing ones
- Change the XML configuration

The following tokens can be used inside `Deploy.ps1` file - they will be replaced by real values when a wrapper is created:
- Application vendor: `{@appVendo}`
- Application name: `{@appName}`
- Application version: `{@appVersion}`
- Application architecture: `{@appArch}`
- Application language: `{@appLang}`
- Application revision: `{@appRevision}`
- Script version: `{@appScriptVersion}`
- Script date: `{@appScriptDate}`
- Script author: `{@appScriptAuthor}`

> **Note:**
> Adjusting the toolkit is not a trivial task and requires an extensive knowledge of PowerShell and the toolkit itself. Refer to the official documentation http://psappdeploytoolkit.com for a guide on how to use the functions and modules.

Managing Default Platforms and Languages

Application vendor: Default language and platform templates for new projects are using RPMST syntax (read chapter RPMST templates for more information about the syntax and supported features). The template are stored in PackPoint folder in the following directory:

- Language templates in `<PackPoint>\Templates\Languages`
- Platform templates in `<PackPoint>\Templates\Platforms`

New templates can be added by simply placing new files in one of these two locations. Bundled definitions can be adjusted using a text editor, to further customize the modifications (using RPMST syntax).

The template will be available in the UI under the same name as its file name (for platforms) or under localized version of a language having the same LCID (Language Code Identifier) as the name of the file. For example, bundled files `1031.rpmst`, `1033.rpmst` and `1045.rpmst` represent German, English, and Polish language, respectively.

> **Be aware:**
> Certain platforms have special meaning in RayPack. The following names: x64, Intel86, Intel have special display names in the RayPack UI.

For example, this is a default RPMST template for `Intel64` platform:

```
<Settings>
  <SIS Platform="Intel64" />
  <Tables>
    <Table Name="Directory">
      <Row Directory="ProgramFiles64Folder" Directory_Parent="TARGETDIR" DefaultDir="
       PROGRA~2|ProgramFiles64" />
      <Row Directory="System64Folder" Directory_Parent="TARGETDIR" DefaultDir=".:
       SYSTEM~2|System64" />
    </Table>
  </Tables>
</Settings>
```

The template changes the platform to `Intel64`, and adds two directory entries required for x64 support: 64-bit Program Files and 64-bit system folder.

Creating Universal Transforms

By default, Windows Installer engine validates MST transforms before applying them to a base MSI image. More specifically, the minimum criteria listed below are checked.

- The transform does not add an already existing table.

- The transform does not remove non-existing tables.
- The transform does not add an already existing row.
- The transform does not remove non-existing rows.
- The transform does not edit non-existing rows.
- The transform does not change the code page.
- Additional validation rules, for example matching product name, language or version can be also enforced.

The default validation rules make it difficult to create a transform that can be applied to any MSI. For example, a transform that adds a property `ALLUSERS = 2` works with packages that do not have it, but will fail if the package already contains it. The solution to that problem is to create a universal transform, which involves the following.

- The content of the transform needs to be planned carefully and changes that may behave incorrectly and / or are dependent on any existing package structure should be avoided. For example, the new content should be added to a new hidden feature instead of an existing one, existing components should not be reused etc.
- All possible errors should be suppressed and product validation should be disabled before the transform is generated.

Be aware:
A universal transform is a quick solution to a content which must be present in all packages, also those coming from vendors. RayPack offers more flexibility by utilizing a concept of RPMST templates, which are even more flexible and provide a basic parametrization. Read more about how to use them in section *Transform templates*.

In order to create an universal transform, make sure that RayPack configuration is correctly set up. In *PackDesigner MST settings*, disable the following MST errors:

MST errors + validation

Error conditions that should be suppressed when the transform is applied:

- ☑ Adding existing rows
- ☑ Deleting missing rows
- ☑ Adding existing tables
- ☑ Deleting missing tables
- ☑ Modifying missing rows
- ☑ Changing code page

Database must meet the following criteria before the transform is applied:

- ☐ Same language
- ☐ Validate ProductVersion
- ☐ Same product
- ☐ Same UpgradeCode

- Adding existing row (checked)
- Deleting missing rows (checked)
- Adding existing tables (checked)

- Deleting missing tables (checked)
- Modifying missing rows (checked)
- Changing code page (checked)

Also, make sure the following validation rules are unchecked:

- Same language (unchecked)
- Same product (unchecked)
- Same `UpgradeCode` (unchecked)
- Validate `ProductVersion` (unchecked)

Save the configuration after making any changes. From now on, MST transforms produced by RayPack will be "universal", that means they will be applicable to any valid Windows Installer database.

Customizing Bootstrappers

Defining Prerequisites for Bootstrapper

This chapter describes how to prepare own definitions of prerequisites.

Standard prerequisite definition

The prerequisite definitions pre-installed with RayPack are stored in the PackPoint folder in the `Prerequisite` subfolder, for example:

```
C:\RayPack\PackPoint\Prerequisites
```

A single prerequisite is defined in an XML file, which has the following syntax:

```
<?xml version="1.0"?>
<Prerequisite
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">

    <MetaData>
        <ProductName>Visual C++ 2010 Runtime Libraries (32-bit)</ProductName>
        <ProductVersion>10.0.30319.1</ProductVersion>
        <Manufacturer>Microsoft Corporation</Manufacturer>
        <SchemaVersion>1.0</SchemaVersion>
    </MetaData>

    <Setup>
        <Command>vcredist_x86.exe</Command>
        <Arguments>/q</Arguments>
        <ExitCodes>1641,3010</ExitCodes>
        <Files>
```

```
                        <File FileHash="B88228D5FEF4B6DC019D69D4471F23EC" Size="5073240">
                                <Path>($PackPointDir)\Prerequisites\($DepName)\vcredist_x86.ex
                                <DownloadPath>http://download.microsoft.com/download/5/B/C/
                                 5BC5DBB3-652D-4DCE-B14A-475AB85EEF6E/
                                 vcredist_x86.exe</DownloadPath>
                                <DownloadPage>https://www.microsoft.com/en-us/download/details
                                 id=40784</DownloadPage>
                        </File>
                        ...
                        ...
                        ...
                </Files>
        </Setup>

        <Conditions>
                <Condition Type="Registry" Property="None" Check="NotExist"
                 Architecture="SystemDefault">
                        <Path>HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\
                         Uninstall\{196BB40D-1578-3D01-B289-BEFC77A11A1E}</Path>
                </Condition>
                <Condition Type="OperatingSystem" Property="Version" Check="Equals"
                 Architecture="X86">
                        <ServicePack>2</ServicePack>
                        <OsVersion>51</OsVersion>
                </Condition>
                <Condition Type="OperatingSystem" Property="Version" Check="Equals"
                 Architecture="SystemDefault">
                        <OsVersion>60</OsVersion>
                </Condition>
                ...
                ...
                ...
        </Conditions>

        <Dependencies>
                <Dependency>
                        <Path>($PackPointDir)\Prerequisites\WindowsInstaller_3.1_x86.rpdep</
                </Dependency>
                ...
                ...
                ...
        </Dependencies>

</Prerequisite>
```

MetaData
Contains a definition of basic details like name and version, that are displayed in PackDesigner interface.

- **ProductName**
  The name to be displayed in prerequisites browse.r

Example:

```
Visual C++ 2010 Runtime Libraries (32-bit)
```

- **ProductVersion**:
  The version to be displayed in the prerequisites browser.

  Example:
  ```
  10.0.30319.1
  ```

- **Manufacturer**
  The manufacturer to be displayed in the prerequisites browser.

  Example
  ```
  Microsoft Corporation
  ```

- **SchemaVersion**
  A reserved value denoting the version of the schema. Must be set to 1.0.

Setup

Contains a definition of setup routine - which files are to be included, what command is used to install the package etc.

- **Command**
  The command to be executed to install the package.

  Example:
  ```
  vcredist_x86.exe
  ```

- **Arguments** (optional)
  The additional arguments to be passed to the command. In many scenarios, any kind of silent switches should be used here.

  Example:
  ```
  /q
  ```

- **ExitCodes** (optional)
  A list of exit codes returned by the installing applications considered as "positive". If the only valid exit code is "0", then this value can be left empty. For example, if the installation has an exit code meaning that the product is already installed, then the respective exit code should be used here. The exit code 0 is always considered to be correct exit code, even if not provided in the list.

  Example:
  ```
  1641,3010
  ```

- **Files**
A list of files required for the installation. See the next section for the definition of a single file.

File

Contains a definition of setup routine - which files are to be included, what command is used to install the package etc.

- **FileHash** (optional)
The hash of the file used to confirm if the downloaded file is not corrupted or incomplete. If omitted, the CRC hash of the actual source file will not be verified.

  Example:
  ```
  B88228D5FEF4B6DC019D69D4471F23EC
  ```

- **Size** (optional)
The size of the file, in bytes. If omitted, the size of the actual source file will not be verified.

  Example:
  ```
  5073240
  ```

- **Path**
The local path where the resource should be present. Placeholders can be used to resolve the PackPoint directory and the dependency name.
`($PackPointDir)` resolves to the full PackPoint directory, for example `C:\RayPack\PackPoint`
`($DepName)` resolves to the name of `.rp`dep file, for example `VCRedistr_2010_x86`.

  Example:
  ```
  ($PackPointDir)\Prerequisites\($DepName)\vcredist_x86.exe
  ```

  This value is required and must point to a valid file location.

- **DownloadPath** (optional)
The source URL used to download the file. The value may be empty if there is no direct download link.

  Example:
  ```
  http://download.microsoft.com/download/5/B/C/5BC5DBB3-652D-4DCE-B14A-
  475AB85EEF6E/vcredist_x86.exe
  ```

- **DownloadPage** (optional)
The source URL used to download the file. The value may be empty if there is no download page.

  Example:
  ```
  https://www.microsoft.com/en-us/download/details.aspx?id=40784
  ```

Dependency

Contains a definition of setup routine - which files are to be included, what command is used to install the package etc.

- **Path**
  A path to a `.rpdep` file containing a dependency for a given prerequisite.

  example:
      `($PackPointDir)\Prerequisites\WindowsInstaller_3.1_x86.rpdep`

  This value is required and must point to a valid file location.

Command Line Bootstrapper

Customizing bootstrapper templates

The bootstrapper templates can be customized by changing the following resources:

- **($PackPointDir)\Wrappers\install.cmd**
  (where `($PackPointDir)` is the PackPoint location)

  This template is used for installation wrappers.

- **($PackPointDir)\Wrappers\uninstall.cmd**
  (where `($PackPointDir)` is the PackPoint location)

  This template is used for uninstall wrappers.

In each template, custom pre- and post-commands can be adjusted. RayPack inserts the content of installation and uninstallation routine by replacing the following placeholders:

- `($InstallationRoutine)`
- `($UninstallationRoutine)`

with appropriate content. For example, a template can be defined like below:

```
@echo off
echo Installing ($ProductName)
($InstallationRoutine)
echo The installation is finished
```

Advanced users can specify different default templates per-profile. To do so, open a profile with any XML editor, locate the following XML elements

```
ProfileConfiguration/Wrapping/CmdTemplates
```

and define them like in the example below:

```
<Wrapping>
        <CreateWrapper>false</CreateWrapper>
        <OutputType>Compressed</OutputType>
        <CmdTemplates>
                <CmdTemplates TargetName="($ProductName)
_($ProductVersion)_install.cmd">
                        <Path>($PackPointDir)\Wrappers\install.cmd</Path>
                </CmdTemplates>
                <CmdTemplates TargetName="($ProductName)
_($ProductVersion)_uninstall.cmd">
                        <Path>($PackPointDir)\Wrappers\uninstall.cmd</Path>
                </CmdTemplates>
                ...
                ...
                ...
        </CmdTemplates>
</Wrapping>
```

Special variables `$(PackPointDir)`, `$(ProductName)` and `$(ProductVersion)` can be used to avoid hardcoding the PackPoint path, product name and product version respectively.

> **Be aware:**
> The default RayPack 6.5 profile does not define any custom template so it might be required to manually create the XML structure described above.

Executable Bootstrapper

Customizing bootstrapper appearance

The bootstrapper images can be customized by changing the following resources:

- **($PackPointDir)\Wrappers\raypack.ico**
  (where `$(PackPointDir)` is the PackPoint location)

  This image represents the icon being displayed in taskbar and in the title bar of the bootstrapper.

- **($PackPointDir)\Wrappers\banner.bmp**
  (where `$(PackPointDir)` is the PackPoint location)

  This image represents the background image of the bootstrapper.

Advanced users can specify different default images per-profile. To do so, open a profile with any XML editor, locate the following XML elements

```
ProfileConfiguration/Wrapping/ExeIcon
ProfileConfiguration/Wrapping/ExeWrapper
```

and override them with own values, for example:

```
<Wrapping>
     <CreateWrapper>false</CreateWrapper>
     <OutputType>Compressed</OutputType>
     <ExeIcon>($PackPointDir)\Wrappers\raypack.ico </ExeIcon>
     <ExeHeader>($PackPointDir)\Wrappers\banner.bmp</ExeHeader>
     <CmdTemplates />
</Wrapping>
```

Special variable `$(PackPointDir)` can be used to avoid hardcoding the PackPoint path.

> **Be aware:**
> The default RayPack 6.5 profile does not define any custom images so it might be required to manually create the XML structure described above.

Advanced Logging Options

Besides the product configuration options that are available from the Settings section of the application UI, there are some additional configuration settings users may adjust to tailor RayPack towards their individual requirements.

Logging RayPack activity

The program data directory (`%AppData%\Roaming\RayPack\Logs`) is used by default to store the application activity log file (`yyyy-MM-dd HH-mm-ss.log`). If the default settings remain unchanged, RayPack adds a new line to this log file for every system DEBUG level message that is generated during application use.

The log is by default to be used as in a rolling appender manner, which means that one global log file is permanently extended until a certain file size is reached. As soon as it is reached, the oldest lines are automatically transferred into an archive of log files, which is by default limited to 10 files. When this second limit is reached, the oldest archive is removed to free a slot for the newest archive file.

In order to change the default settings for the log file behavior, users have to manually edit the `log4net.config` file, which resides in the root of the installation folder of RayPack (usually something like `C:\Program Files (x86)\RayPack\`). The settings which are most likely to be of interest for adjustments are:

Log file storage location

The log file is by default "`%appdata%\RayPack\Logs\dd-MM-yyyy-HH-mm-ss.log.log`". However, it is possible to define any other absolute local paths as well as shared network locations for logging resource storage.

```
<file type="log4net.Util.PatternString"
      value="%env{AppData}\\RayPack\\Logs\\%date{yyyy-MM-dd HH-mm-ss}.log" />
```

> **Be aware:**
> The user that runs RayPack must have write permissions in the log file location in order to initiate and maintain the message flow to the log file. If the user does not have sufficient access rights, there will be no error message, or actual product usage cutback, but simply a loss of system activity documentation. Please refer to the Troubleshooting section for additional instructions in case of missing logs.

Max. log file size

The max. log file size may be defined as "`KB`", "`MB`" or "`GB`". The default setting for newly installed RayPack instances is "`2048KB`"

```
<maximumFileSize value="2048KB" />
```

Log level

The most frequently used log level settings are `DEBUG`, `INFO`, `WARN`, `ERROR`, `FATAL`, `OFF`, whilst `OFF` prevents logging at all, `FATAL` is the most restrictive but still writing setting, and `DEBUG` the most talkative option.

The recommendation is to use the `DEBUG` level for newly setup systems, since a lot of the information logged in this mode may help to adjust settings regarding access rights, and the like. As soon as the application and system are up and running productively, setting the log level to `WARNING` should be sufficient for permanent maintenance.

```
<level value="DEBUG" />
```

Default logging configuration

The default configuration file is given below as a review and backup support:

```
<?xml version="1.0" encoding="utf-8" ?>
<log4net xmlns="urn:log4net">
  <appender name="FileAppender" type="log4net.Appender.FileAppender">
```

```xml
    <file type="log4net.Util.PatternString" value="%property{PathToLog}%
property{LogFileName}" />
    <appendToFile value="true" />
    <layout type="log4net.Layout.PatternLayout">
      <header value="/***************************** LOG HEADER
*****************************************/&#13;&#10;"/>
      <footer value="/**************************** LOG FOOTER
****************************************/&#13;&#10;"/>
      <conversionPattern value="%date [%thread] [%property{ProcessTime}] %-5level %
logger - %message%newline" />
    </layout>
  </appender>


    <appender name="TraceAppender" type="log4net.Appender.TraceAppender">
        <layout type="log4net.Layout.PatternLayout">
            <conversionPattern value="%date %message%newline" />
        </layout>
    </appender>


    <appender name="FormAppender" type="raypack.msi.WPF.BAL.FormAppender,
raypack.msi.WPF">
        <layout type="log4net.Layout.PatternLayout">
            <conversionPattern value="%message" />
        </layout>

    <!-- Do not show debug messages in this logger -->
    <threshold value="INFO" />

    <!-- Filtering by namespace (instead of hardcoded checks) -->
    <filter type="log4net.Filter.LoggerMatchFilter">
      <loggerToMatch
value="raypack.msi.BusinessLayer.Snapshots.Snapshots.SnapshotManager" />
    </filter>
    <filter type="log4net.Filter.LoggerMatchFilter">
      <loggerToMatch
value
="raypack.msi.BusinessLayer.Snapshots.ExternalTools.SnapshotTools.RpMakeToolMonitor
" />
    </filter>
    <filter type="log4net.Filter.LoggerMatchFilter">
      <loggerToMatch
value="raypack.msi.BusinessLayer.Snapshots.Snapshots.Filtering.FilterManager" />
    </filter>
    <filter type="log4net.Filter.LoggerMatchFilter">
      <loggerToMatch value="raypack.msi.WPF.BAL.MsiMaker" />
    </filter>
    <filter type="log4net.Filter.LoggerMatchFilter">
      <loggerToMatch value="raypack.msi.WPF.BAL.MstMaker" />
    </filter>
    <filter type="log4net.Filter.LoggerMatchFilter">
      <loggerToMatch value="raypack.msi.BusinessLayer.Tables" />
    </filter>
    <filter type="log4net.Filter.LoggerMatchFilter">
      <loggerToMatch value="raypack.msi.BusinessLayer.Utils.CABHelper" />
    </filter>
    <filter type="log4net.Filter.LoggerMatchFilter">
      <loggerToMatch value="raypack.msi.BusinessLayer.MsmPool" />
    </filter>
```

```xml
    <filter type="log4net.Filter.DenyAllFilter" />
</appender>

    <root>
    <level value="ALL"/>
    <appender-ref ref="FileAppender"/>
            <appender-ref ref="TraceAppender"/>
            <appender-ref ref="FormAppender"/>
    </root>
</log4net>
```

Further Information

RayPack uses an external library to provide logging functionality. Please refer to the online-documentation provided for the log4net project (http://logging.apache.org/log4net/) in order to get further details regarding available configuration and usage options. log4net can be adjusted to connect directly with databases or event-loggers. There are numerous options for layout and behavior manipulations. RayPack system administrators with a slight affection for perfection are highly welcome to configure their very own logger version.

Configuring the Complexity Index

The **Complexity index** is a number estimating the internal complexity of an MSI / RPP / MST setup.
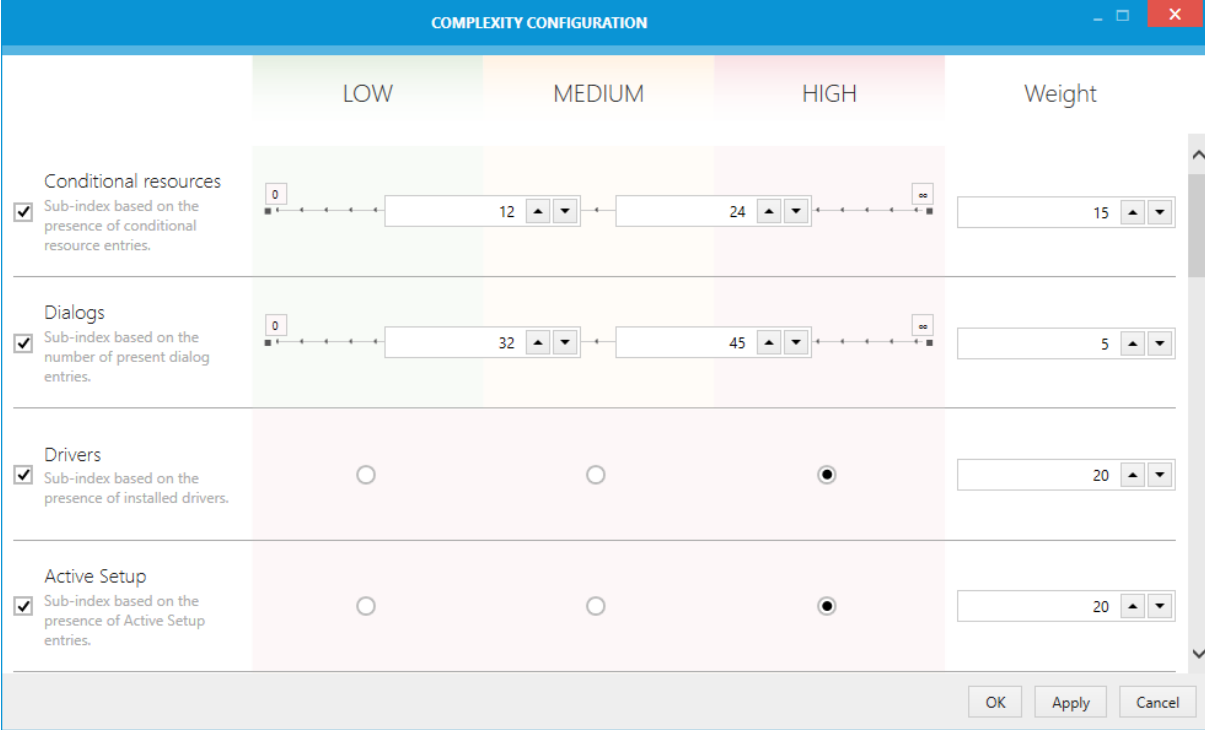
RayPack 5.0 has a default out-of-the-box algorithm to determine the complexity based on a number of factors. In some cases, it may be required to change the way this number is calculated. An example on how this can be done is by excluding certain factors or changing their importance. The configuration of the complexity can be accessed from the PackDesigner view by going to the **Plugins** view and focusing the **Complexity Index** item.

Press **Configure the way complexity is calculated...** to show a new window with a grid that allows adjustments to the algorithm.

The rows represent different categories and elements RayPack can include when calculating the complexity. If the checkbox next to an item is checked, the item affects the complexity. If the checkbox is unchecked, the item does not affect the complexity.

The columns represent different values and their thresholds. Their meaning may differ based on the type of an item.

- Boolean items (for example whether Active Setup is present or not) have three radio buttons. The impact of the presence is the same, but when an item is marked as **Medium** or **High** it may get extra points at the end of the calculation. If the check returns true (for example, ActiveSetup is present) then the overall index is increased by the amount shown in the **Weight** column.

- Discrete items (for example file counters) have two numeric textboxes. They define thresholds between **Low**, **Medium**, and **High** complexity. The actual value that is added to the overall index is calculated by using linear approximation of selected point ranges. For example, if the value of files lies exactly in the middle between 0 and the lower number, the package gets half of points for **Low** complexity. Additionally, if the amount hits **Medium** or **High** complexity, the package may get some extra points at the end of the calculation.

Weights denote the importance of every check. The more weight an item has, the more important its result is. To calculate the result, sum of all active sub-index values is divided by sum of weights of all active sub-indexes, and then extra points may be assigned depending on the presence of any **Medium** or **High** complexity hit.

In order to save the changes, press **OK**. Changes are saved to the current profile.

Getting started with the configuration

- **Question:** I do not want to include some specific category in my calculation.
  Answer: Uncheck the checkbox shown next to the category to exclude it from calculations.

- **Questions:** I want to increase importance of a category.
  Answer: Change its weight to a higher value.

- **Questions:** I want to decrease importance of a category.
  Answer: Change its weight to a lower value.

- **Questions:** I do not want extra points for a presence of a boolean check (for example Active Setup).
  Answer: Change its complexity to Low.

- **Questions:** I want that packages having more than 1000 files get higher complexity indexes aside of points from a regular calculation.
  Answer: Change the upper range to 1000. Set the lower range to any value from 0-999.

Creating PackDesigner Scripts

PackDesigner provides an easy way to implement scripting-based logic to all MSI / RPP / MST projects. The scripts (macros) are shown in the aggregated **Plugins** view. Scripts are simple VBS or PowerShell files that should be present in the following location.

```
<PackPoint>\Scripts
```

Here `<PackPoint>` is the location where the PackPoint resources are installed. Default installation of RayPack already contains two sample scripts for GUID manipulations (both equal from a functional point of view; one written in VBS, the other one in PowerShell). Scripts are shown in the tree, and file names are used as node names.

Accessing the temporary database

Regardless of the technology chosen, the script is started with a single argument - a full path to a temporary database that the script should update. Below is an example of a script that simply starts an Orca tool and opens the current package in Orca.

```
const DontWaitUntilFinished = false, ShowWindow = 1, DontShowWindow = 0,
WaitUntilFinished = true

'Standard VBS object, used to operate with windows shell (explorer)
Dim objShell
Set objShell = WScript.CreateObject( "WScript.Shell" )

'Finding correct path to the ProgramFiles directory
```

```
programFilesPath = objShell.ExpandEnvironmentStrings("%
ProgramFiles(x86)%")
arguments = ""

'Checking, if we have at least one command line argument. It's expected
to be MSI file path.
if WScript.Arguments.Count > 0 Then
arguments = """" & WScript.Arguments(0) & """"
end if

'Creating final executable path with arguments
exePath = """" & programFilesPath & "\Orca\Orca.exe"" " & arguments
'Running the application and waiting for it to finish
objShell.Run exePath, ShowWindow, WaitUntilFinished

Set objShell = Nothing
```

Updating the temporary database

Below is a default script (also VBS) that resets all package GUIDs and saves the temporary database. After the scripts is finished, RayPack collects the changes and applies them in the current project.

In order to open, edit and save an MSI database, MSI COM interface is used. The documentation of its methods can be found in MSDN website: https://msdn.microsoft.com/en-us/library/windows/desktop/aa367810(v=vs.85).aspx.

```
Option Explicit

Function CreateGUID
  Dim TypeLib
  Set TypeLib = CreateObject("Scriptlet.TypeLib")
  dim guid
  guid = TypeLib.Guid

  CreateGUID = Left(UCase(guid), 38)
End Function

Dim installer, database, view, record, viewUpdate

Set installer = CreateObject("WindowsInstaller.Installer")
Set database = installer.OpenDatabase (WScript.Arguments(0), 1)

Set view = database.OpenView ("SELECT Component FROM Component")
view.execute

set record = view.Fetch

Do While Not record Is Nothing

   dim componentName, componentGuid
   componentName = record.StringData(1)
   componentGuid = CreateGUID
```

```
        set viewUpdate = database.OpenView ("UPDATE Component SET
        ComponentId='" & componentGuid & "' WHERE Component='" &
        componentName & "'")
        viewUpdate.Execute

        set record = view.Fetch
loop

set viewUpdate = database.OpenView ("UPDATE Property SET Value='" &
CreateGUID & "' WHERE Property='ProductCode'")
viewUpdate.Execute

set viewUpdate = database.OpenView ("UPDATE Property SET Value='" &
CreateGUID & "' WHERE Property='UpgradeCode'")
viewUpdate.Execute

database.Commit
Set database = Nothing
Set installer = Nothing
Set view = Nothing
```

A similar approach can be executed with PowerShell. Please note that the parameter `$Path` is defined, and the value of that parameter will be set externally by RayPack before calling the PowerShell script engine. In this example MSI COM interface is used to update the MSI, and its methods are called using Reflection.

```
param(
    [parameter(Mandatory=$true)]
    [ValidateNotNullOrEmpty()]
    [String]$Path
)
Process {
    try {
        # Read property from MSI database
        $OpenReadOnly = 0
        $OpenTransact = 1

        $WindowsInstaller = New-Object -ComObject
        WindowsInstaller.Installer

        $MSIDatabase =
        $WindowsInstaller.GetType().InvokeMember("OpenDatabase",
        "InvokeMethod", $null, $WindowsInstaller, @($Path,
        $OpenTransact))

        $Query = "SELECT Component FROM Component"
        $View = $MSIDatabase.GetType().InvokeMember("OpenView",
        "InvokeMethod", $null, $MSIDatabase, ($Query))
        $View.GetType().InvokeMember("Execute", "InvokeMethod", $null,
        $View, $null)
        $Record = $View.GetType().InvokeMember("Fetch", "InvokeMethod",
        $null, $View, $null)
```

```
            while ($Record -ne $null)
            {
                $componentName = $Record.GetType().InvokeMember("StringData",
                "GetProperty", $null, $Record, 1)
                $newGuid = [guid]::NewGuid().ToString("B").ToUpper()

                $updateQuery = "UPDATE Component SET ComponentId='" +
                $newGuid + "' WHERE Component='" + $componentName + "'"
                $viewUpdate = $MSIDatabase.GetType().InvokeMember("OpenView",
                "InvokeMethod", $null, $MSIDatabase, ($updateQuery))
                $viewUpdate.GetType().InvokeMember("Execute", "InvokeMethod",
                $null, $viewUpdate, $null)

                $Record = $View.GetType().InvokeMember("Fetch",
                "InvokeMethod", $null, $View, $null)
                $viewUpdate.GetType().InvokeMember("Close", "InvokeMethod",
                $null, $viewUpdate, $null)
            }

        $viewUpdate = $MSIDatabase.GetType().InvokeMember("OpenView",
        "InvokeMethod", $null, $MSIDatabase, ("UPDATE Property SET
        Value='" + [guid]::NewGuid().ToString("B").ToUpper() + "' WHERE
        Property='ProductCode'"))
        $viewUpdate.GetType().InvokeMember("Execute", "InvokeMethod",
        $null, $viewUpdate, $null)

        $viewUpdate = $MSIDatabase.GetType().InvokeMember("OpenView",
        "InvokeMethod", $null, $MSIDatabase, ("UPDATE Property SET
        Value='" + [guid]::NewGuid().ToString("B").ToUpper() + "' WHERE
        Property='UpgradeCode'"))
        $viewUpdate.GetType().InvokeMember("Execute", "InvokeMethod",
        $null, $viewUpdate, $null)




        # Commit database and close view
        $MSIDatabase.GetType().InvokeMember("Commit", "InvokeMethod",
        $null, $MSIDatabase, $null)
        $View.GetType().InvokeMember("Close", "InvokeMethod", $null,
        $View, $null)
        $viewUpdate.GetType().InvokeMember("Close", "InvokeMethod",
        $null, $viewUpdate, $null)
        $MSIDatabase = $null
        $View = $null
        $viewUpdate = $null
    }
    catch {
        Write-Warning -Message $_.Exception.Message ; break
    }
}
End {
    # Run garbage collection and release ComObject
    [System.Runtime.Interopservices.Marshal]::ReleaseComObject($WindowsI
    nstaller)
```

```
            [System.GC]::Collect()
      }
```

Troubleshooting PackBot

Troubleshooting Problems with Hyper-V Connectivity

The following checklist helps to find and fix any possible issues when working with Hyper-V machines:

1. Is PowerShell 3.0 installed (both on Guest and Host Machine)?
   a. Check `$PSVersionTable.PSVersion` in PowerShell
2. Is the machine properly configured in the **Settings** > **Virtual Machines** screen (pay attention to hardcoded IP addresses which may be dynamically assigned by DHCP)
3. Is RayPack Studio Tools for Hyper-V installed on the Guest machine? Is the process `vm-proxy.exe` from RayPack Studio Tools for Hyper-V running?
4. Is WINRM configured?
   a. Check `winrm qc`
5. Does WINRM have proper `TrustedHosts` entries on both VM and server?
   a. `winrm s winrm/config/client '@{TrustedHosts="RemoteComputer"}'`
   b. `winrm g winrm/config/client` - shows the current `TrustedHosts` lists
   c. More information: https://technet.microsoft.com/en-us/library/ff700227.aspx
6. Does WINRM have a connection to the VM and vice-versa?
   a.- `Test-WSMan -ComputerName IP`
7. Are all necessary ports unblocked on the physical machine?
   a. The default port range is `48654-48999`.

Changing TCP / IP Configuration

In some cases it may be required to use custom port ranges, timeouts, etc. for PackBot related functionality.

The following table summarizes the available options:

Setting name	Default value	Description
`TcpIpDefaultPort`	48654-48999	Port range used for TCP/IP communication. Use minus (-) and comma (,) to indicate which ports are valid for incoming communication. Make sure that these ports are not blocked by your firewall. PackBot tries to find first valid free port and listen on it from lower to higher numbers.
`TcpIpMaxRetry`	3	Maximum number of retries before asserting the machine is not available..

Setting name	Default value	Description
`TcpIpDefaultReceiveTi`	240000	Reverts to default value if Windows does not define its own timeouts.
`TcpIpDefaultSendTimeout`	240000	Reverts to default value if Windows does not define its own timeouts.

Reference and Cheat Sheets

Active Setup

Active Setup is a mechanism for executing commands once per user early during logon. Active Setup is used by some operating system components like Internet Explorer to set up an initial configuration for new users logging on for the first time. Active Setup is also used in some corporation's software distribution systems to create an initial customized user environment. To understand why such a mechanism is necessary we need to take a step back.

Application programs use two different types of data: machine-specific data like the files in the installation directory, and user-specific data like the information which document a user last edited with the application. Installing machine-specific data is simple. Just copy it to `C:\Program Files` and `HKEY_LOCAL_MACHINE` and you are done. But how to get an initial user configuration into a user profile? Writing into the profile of the user doing the install does not help, because we need the initial configuration for all users logging on to the system.

One solution to this problem is Active Setup. It uses both machine-specific data and user-specific data. The machine part consists of a list of components each identified by a GUID. The user part is basically a mirror of the machine data, but, and this is the key point, it does not exist in new user profiles. Whenever a user logs on, Active Setup checks if each machine part component GUID is present in the user part. If not, the component's command is executed and the component's GUID is stored in the user part. If it is, the current user profile has already been initialized and no further action is required.

Active Setup runs before the shell is started, i.e. before the Desktop appears. Commands started by Active Setup run synchronously, blocking the logon while they are executing. Active Setup is executed before any Run or RunOnce registry entries are evaluated because Run(Once) is handled by the shell, `Explorer.exe`, which is started only after Active Setup is finished.

> **Tip:**
> RayPack contains a *ready-to-use sample of a template* that can be used to automatically set up all necessary ActiveSetup entries. This advanced section describes details about values and mechanisms being used.

Registry Entries Related to Active Setup

`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Active Setup\Installed Components`
This is the root key containing all things Active Setup. The keys and values mentioned below are all located below this root key.
A duplicate of this machine key exists in the user profile.
`HKEY_CURRENT_USER\SOFTWARE\Microsoft\Active Setup\Installed Components`

This is what is known as the "user part" Active Setup key.

GUID

- Type: registry key
- For each component, there is a GUID key below the root key. Technically, this need not be a GUID, but GUIDs have the advantage of being unique.
- Each GUID represents one component to be managed by Active Setup. The number of components is not limited and there can be zero, one or multiple components per application. The number of components is dependent on the number of commands that are to run – only one command per component is possible (see StubPath below). Typically, the `ProductCode` property of the msi package is used as the GUID, as this is guaranteed to be unique.

Default Value

- Type: `REG_SZ`
- Optional name of the component. If a name is stored here, it will be shown in the Active Setup user interface dialog when the component's command is run. It is advisable to use the `ProductName` property here, as this is easy to recognize.

IsInstalled

- Type: `REG_DWORD`
- Possible values:

    0: The component's command will not run.

    1: The component's command will be run once per user. This is the default (if the IsInstalled value does not exist).

Locale

- Type: `REG_SZ`
- An arbitrary string specifying the installation language of the component. If this string is either not found in the user part or different from the machine part, the component is run. Please note that Active Setup does not impose any restrictions on the nature of this string, you could use "abc" just as well as "de" or "en". Once run, the version number is copied to the user part. It is advisable to use an "asterisk" here (*).

StubPath

Type: `REG_SZ` or `REG_EXPAND_SZ`

- Format: Any valid command line, e.g. "notepad"
- This is the command that is executed if Active Setup determines this component needs to run during logon. Normally for use in msi packages, the `StubPath` contains the msi engine executable, the `ProductCode` property of the msi package, and any command lines that the msi engine requires. For example, `msiexec.exe /Fpums [ProductCode] /Qn`

> **Note:**
> The `[ProductCode]` property would be replaced by the msi engine on installation with the correct `ProductCode` property of the msi package.

Version

- Type: `REG_SZ`
- Format: Four numbers separated by commas, e.g.: 1,2,3,4 (periods do not work)
- If a version value is present, the component's command will run only if the corresponding version in the user part is smaller or not present. Once run, the version number is copied to the user part. If you want a component's command to run again, you need to increase the version number. That is what Windows updates do, e.g. when you install Internet Explorer 8 on a machine that only had IE7 before. By increasing the version number, the initial user configuration is guaranteed to run again.

Things You Should Know

Active Setup employs neither a timeout nor any other mechanism to determine if a `StubPath` process it started is still alive. That means it is very easy to shoot yourself in the foot: if a process started by Active Setup hangs, Active Setup hangs, too, and there is typically no easy way to remedy the situation, except flipping the big red power switch. Therefore it is advisable to thoroughly test the implementation of any packages that use Active Setup to avoid problems when deploying the package.

Original text from Helge Klein

MSI ICE Reference

A common problem when carrying out the quality of MSI based packages are the so called ICE errors and warnings. What they are and what checks are executed against the MSI (and MST) packages are listed and explained here.

Regular Expressions Guide and Cheat Sheet

Regular expressions (abbreviated to RegExp) are sequences of characters forming a search pattern that can be used to identify a textual material of a given pattern. The syntax of regular expression is standardized and used along many different products and operating systems, particularly for find and replace functionalities.

RayPack uses regular expressions to provide a flexible way of defining compact text conditions, which can be further customize to exclude and include particular resources based on complicated rules and logic.

The following cheat sheet presents some commonly used regular expression patterns and example patterns with explanation.

Phrase	Description	Examples
Letters or digits	Literal meaning, case sensitive	`Abc` matching `Abc` but not `abc`
Pipe (`|`)	Alternative	`A|b` matching `A`, `b`, `Ab` but not `BC`
Asterisk (`*`)	Zero or more instances of the previous set	`A*b` matching `Aaaaab`, `Ab`, `bcd` but not `Ac` `A|b*` matching `a`, `b`, `bb`, `bbb` but not `CDE`

Plus (`+`)	One or more instances of the previous set	`A+b` matching `Ab`, `AAAb` but not `b`
		 `(a|b)+cd` matching `acd`, `bcd`, `aaacd`, `abcd`, `bacd` but not `cd`
		 `Folder[0-9]+` matching `Folder1`, `Folder2`, `Folder21` but not `Folder` or `FolderA`

Phrase	Description	Examples
Question mark (`?`)	Zero or one instance of the previous set	`Ab?(c|d)` matching `Ac`, `Ade`, `Abcd` but not `Abb`
Round brackets `(` and `)`	Grouping of sets	`(abc|def)ghi` matching `abcghi`, `defghi`, `abcdefghi` but not `abcdef` `(a|b|c)?def` matching `adef`, `cdefghi`, `def` but not `abc`
Caret (`^`)	On the beginning of the string marks that no characters are allowed before	`^abc` matching abc, abcdef but not defabc `^(folder1|folder2|folder3)\\test` matching `folder1\test`, `folder2\test\test2` but not `C:\folder1\test`
Dollar (`$`)	At the end of the string marks that no characters are allowed after	`:\\test\\folder$` matching `C:\test\folder`, `D:\test\folder` but not `C:\test\folder1`

Square brackets `[` and `]`	Allowed set of characters	`[a-z]` matching `abc`, `def` but not `123`
		`^[A-D]:\\Test\\` matching `C:\Test\` and `D:\Test\` but not `E:\Test\`
		`folder[a-zA-Z0-9]` matching `folderA`, `folder1` but not `folder\test`

Caret in square brackets `[^]`	In square brackets – negation of character set	`C:\\folder[^\\]*\\test` will match `C:\folder\test` but not `C:\folder\subfolder\test`
Dot (`.`)	Any character	`Folder.\\test` will match `FolderA\`, `Folder4\`, `Folder$\` but not `Folder\`

Backslash (`\`)	Escape character	`C:\\test\\test2` matches `C:\test\test2` but not `C:\\test\\test2`
		`Test\*` matches `Test*`
		`Test\(a\)` matches `Test(a)` but not `Testa`

`(?i)`	Used at the beginning on the expression – do not match case	`(?i)abc` will match `abc`, `Abc`, `ABC123` but not `123`

Examples

```
^(?i)(%windir%)\\Installer$
```

Matches	Does not match
%windir%\Installer %windir%\INSTALLER	%windir%\Installer\123-123.msi

```
^(?i)%ProgramFiles\(x86\)%\\Common Files\\(InstallShield|Wise Installation)$
```

Matches	Does not match
%ProgramFiles(x86)%\Common Files \InstallShield %ProgramFiles(x86)%\COMMON FILES \Wise Installation	%ProgramFiles(x86)%\Common Files\Micr %ProgramFiles%\Common Files\InstallSh %ProgramFiles(x86)%\Common Files\ %ProgramFiles(x86)%\Common Files\ Wise Installation\Subfolder

```
^unins(|t|tall)\d*\.(cif|cfg|dat|dll|ini|exe|xml|lnk)$
```

Matches	Does not match
uninst.exe uninst_myApp.lnk uninstallapp.cfg uninst.ini	MyProgram_uninstall.cfg uninstall.txt unins.txt

```
^_isreg32\.dll$
```

Matches	Does not match
_isreg32.dll	_ISREG32.dll isreg32.dll _isreg.dll _isreg32.dll.backup

```
^(?i)(HKEY_LOCAL_MACHINE|HKEY_CURRENT_USER)\\(Software\\Wow6432Node|Software)
\\InstallShield
```

Matches	Does not match
HKEY_LOCAL_MACHINE\Software\InstallSh HKEY_LOCAL_MACHINE\Software\Wow6432No \InstallShield HKEY_CURRENT_USER\SOFTWARE\Wow6432Nod \InstallShield HKEY_CURRENT_USER\Software\INSTALLSHI]	HKEY_CLASSES_ROOT\Software\InstallShi HKEY_CURRENT_USER\Software \Progrems\InstallShield

```
^%USERPROFILE%\\(.+\\)?Temp$
```

Matches	Does not match
%USERPROFILE%\Temp	%USERPROFILE%\Temp2
%USERPROFILE%\test\Temp	"%USERPROFILE%\Temp"
%USERPROFILE%\test\test2\Temp	%userprofile%\Temp

User Interface Shortcuts

The RayPack user interface offers a set of shortcuts for those packagers, who are used to leave the mouse aside and swiftly work their way through task sequences by keyboard.

Scope	Hotkeys	Action
All editor views	CTRL + N	Creates a new project
All editor views	CTRL + O	Opens existing project
All editor views	CTRL + S	Saves existing project
All editor views	CTRL + F4	Closes current project
All editor views	F7	Builds the current project
PackDesigner	CTRL+F7	Quick build of the current package
PackDesigner	F8	Test current package
PackDesigner	CTRL + Q	Shows the Quality backstage
PackDesigner	CTRL + T	Shows the Transform backstage
PackDesigner	CTRL + SHIFT + T	Switch to Advanced mode > TABLES
PackDesigner	CTRL + SHIFT + F	Switch to Advanced mode > FEATURES
PackDesigner	CTRL + SHIFT + C	Switch to Advanced mode > COMPONENTS
PackDesigner	CTRL + SHIFT + A	Switch to Advanced mode > CUSTOM ACTIONS
PackDesigner	CTRL + SHIFT + S	Switch to Advanced mode > SEQUENCES
PackDesigner	CTRL + SHIFT + U	Switch to Advanced mode > UPGRADES
PackDesigner	CTRL + SHIFT + I	Switch to Advanced mode > USER INTERFACE
PackDesigner > Visual Mode	CTRL + 2	Switch to Advanced mode
PackDesigner > Advanced Mode	CTRL + 1	Switch to Visual mode
PackDesigner > Advanced Mode > TABLES	CTRL + C	Copy a row
PackDesigner > Advanced	CTRL + X	Cut a row

Scope	Hotkeys	Action
Mode > TABLES		
PackDesigner > Advanced Mode > TABLES	CTRL + V	Paste a row
PackDesigner > Advanced Mode > TABLES	Delete	Delete a row
PackDesigner > Advanced Mode > TABLES	Insert	Insert a new row
All trees (directory browser, registry browser etc.)	*	Expand the currently selected node and all of its children
All trees (directory browser, registry browser etc.)	-	Collapse the currently selected node

Custom RayPack Installer Database Tables

Whilst most of the tables stored within a packaging project database belong to the standard defined for MSI packages, some additional custom tables are added by RayPack. All custom tables start with the prefix "RP" and are designed to allow users to benefit from built-in Visual Designer logic that extends the possibilities of the MSI standard.

Usually, these custom tables are not present in MSI packages that have been created with other packaging tools, or in newly created RPP projects. They are automatically added to the packaging project's database when they are required. They are required when the packaging engineer uses a RayPack convenience functionality, such as predefined Visual Designer interfaces for TXT change tasks, SQL database and SQL script support, etc..

The following help topics tend to describe the structure of custom RayPack database tables, as well as their relevance for PackDesigner views and functions.

> **Note:**
> Please refer to the MSDN documentation for details regarding the specifications and restrictions of standard column types mentioned below.

RPBuild

The RPBuild table contains the columns shown in the following table.

Column	Type	Key	Nullable
BuildId	Identifier (s72)	Y	N
Key	Identifier (s72)	Y	N

Column	Type	Key	Nullable
Value	Text (s0)	N	Y

Columns

BuildId
The identifier of a build. (Valid values: MSI, THINAPP, APPV, COMMON)

Key
The name of build property.

Value
The value of the build property.

RPIIsAppPool

The RPIIsAppPool table contains the columns shown in the following table.

Column	Type	Key	Nullable
RPIIsAppPool	Identifier	Y	N
Name	Formatted (s72)	N	N
Component_	Identifier (s72)	N	N
Attributes	Integer (i2)	N	N
RPUser_	Identifier (S72)	N	Y
RecycleMinutes	Integer (I2)	N	Y
RecycleRequests	Integer (I2)	N	Y
RecycleTimes	Text (S72)	N	Y
IdleTimeout	Integer (I2)	N	Y
QueueLimit	Integer (I2)	N	Y
CPUMon	Text (S72)	N	Y
MaxProc	Integer (I2)	N	Y
VirtualMemory	Integer (I4)	N	Y
PrivateMemory	Integer (I4)	N	Y
ManagedRuntimeVersion	Text (S72)	N	Y
ManagedPipelineMode	Text (S72)	N	Y
CreateOrOverwriteExisting	Integer (I2)	N	Y

Columns

RPIIsAppPool
Primary key, non-localized token for AppPool.

Name
Name to be used for the IIs AppPool.

Component_
Foreign key into the Component table denoting the component whose selection gates the AppPool creation/deletion.
If the Component marked as Win64 = yes, the application pool will be created with Enable 32 bit flag set to false.
Otherwise (that is, by default), it will be created with Enable32bit set to true.

Attributes
Configure the application pool to run as
networkService,localService,localSystem,other,applicationPoolIdentity.

RPUser_
User account to run the AppPool as. To use this, you must set the Identity attribute to other.

RecycleMinutes
Number of minutes between recycling app pool.

RecycleRequests
Number of requests between recycling app pool.

RecycleTimes
Times to recycle app pool comma delimited - i.e. 1:45,13:30

IdleTimeout
Amount of idle time before shutting down.

QueueLimit
Reject requests after queue gets how large.

CPUMon
A comma delimited list of the following format: percent CPU usage, refresh minutes, Action.
Possible values for Action are
0 = No Action
1 = KillW3wp
2 = Throttle
3 = ThrottleUnderLoad

MaxProc

Maximum number of processes to use.

VirtualMemory

Amount of virtual memory (in KB) that a worker process can use before the worker process recycles. The maximum value supported for this field is 4,294,967 KB.

PrivateMemory

Amount of private memory (in KB) that a worker process can use before the worker process recycles. The maximum value supported for this field is 4,294,967 KB.

ManagedRuntimeVersion

Specifies the .NET Framework version to be used by the application pool.

ManagedPipelineMode

Specifies the request-processing mode that is used to process requests for managed content.

CreateOrOverwriteExisting

1 = Yes
If the application pool already exists on the target system, the installation overwrites its settings with the values that you have configured for it in your project. If the application pool does not exist on the target system, the installation creates it.
0 = No
If the application pool already exists on the target system, the installation does not.

RPIIsWebAddress

The RPIIsWebAddress table contains the columns shown in the following table.

Column	Type	Key	Nullable
RPIIsWebAddress	Identifier (s72)	Y	N
RPIIsWebSite_	Identifier (s72)	N	N
IP	Formatted (S255)	N	Y
Port	Integer (I2)	N	Y
Header	Text (S255)	N	Y
Secure	Integer (I2)	N	Y

Columns

RPIIsWebAddress

Primary key, non-localized token.

RPIIsWebSite_
Foreign key referencing Web that uses the address.

IP
String representing IP address (#.#.#.#) or NT machine name (fooserver)

Port
Port web site listens on.

Header
Special header information for the web site.

Secure
Specifies whether SSL is used to communicate with web site. Valid values: 0, 1

RPIIsWebApplication

The RPIIsWebApplication table contains the columns shown in the following table.

Column	Type	Key	Nullable
RPIIsWebApplication	Identifier (s72)	Y	N
Name	Formatted (s255)	N	N
Isolation	Integer (i2)	N	N
AllowSessions	Integer (I2)	N	Y
SessionTimeout	Integer (I2)	N	Y
Buffer	Integer (I2)	N	Y
ParentPaths	Integer (I2)	N	Y
DefaultScript	Text (S26)	N	Y
ScriptTimeout	Integer (I2)	N	Y
ServerDebugging	Integer (I2)	N	Y
ClientDebugging	Integer (I2)	N	Y
RPIIsAppPool_	Integer (I2)	N	Y

Columns

RPIIsWebApplication

Primary key, non-localized token for ASP Application.

Name
Name of application in IIS MMC applet.

Isolation
Isolation level for ASP Application:
0 = Low
1 = High
2 = Medium

AllowSessions
Specifies whether application may maintain session state.

SessionTimeout
Time session state is maintained without user interaction.

Buffer
Specifies whether application buffers its output.

ParentPaths
Specifies whether to allow ASP client-side script debugging.

DefaultScript
Default scripting language for ASP applications.

ScriptTimeout
Time ASP application page is permitted to process.

ServerDebugging
Specifies whether to allow ASP server-side script debugging.

ClientDebugging
Specifies whether to allow ASP client-side script debugging.

RPIIsAppPool_
App Pool this application should run under.

RPIIsWebApplicationExtension

The RPIIsWebApplication table contains the columns shown in the following table.

Column	Type	Key	Nullable
RPIIsWebApplication_	Identifier (s72)	Y	N

Column	Type	Key	Nullable
Extension	Formatted (s255)	Y	N
Verbs	Text (S255)	N	Y
Executable	Text (s255)	N	Y
Attributes	Integer (I2)	N	Y
Description	Text (s255)	N	N

Columns

RPIIsWebApplication_
Foreign key, referencing possible ASP application for the web site.

Extension
Primary key, extension that should be registered for this ASP application.

Verbs
Specifies the HTTP verbs for which the handler mapping applies. (e. g. POST, GET, HEAD, TRACE)

Executable
Reference to the executable file.

Attributes
Attributes for the extension
1 = Allows processing for files of this type in a directory with Script permission
4 = Allow the server to check that the requested script exists before mapping an extension to an application.

Description
The displayed name of the extension

RPIIsWebDirProperties

The RPIIsWebDirProperties table contains the columns shown in the following table.

Column	Type	Key	Nullable
RPIIsWebDirProperties	Identifier (s72)	Y	N
Access	Integer (I2)	N	Y
Authorization	Integer (I2)	N	Y
RPUser_	Identifier (S72)	N	Y

Column	Type	Key	Nullable
IIsControlledPassword	Integer (I2)	N	Y
LogVisits	Integer (I2)	N	Y
Index	Integer (I2)	N	Y
DefaultDoc	Text (S255)	N	Y
AspDetailedError	Integer (I2)	N	Y
HttpExpires	Text (S255)	N	Y
CacheControlMaxAge	Integer (I4)	N	Y
CacheControlCustom	Text (S255)	N	Y
NoCustomError	Integer (I2)	N	Y
AccessSSLFlags	Integer (I2)	N	Y
AuthenticationProviders	Text (S255)	N	Y

Columns

RPIIsWebDirProperties
Primary key, non-localized token for Web Properties.

Access
Access rights to the web server.

Authorization
Authorization policy to web server (anonymous access, NTLM, etc)

RPUser_
Foreign key, User used to log into database.

> **Be aware:**
> The definition of this column has undergone slight changes between RayPack 1.3 and 1.4: Whilst the type was s72 in older versions, it is S72 in the current one. This modification may cause issues during the edition of RPP projects, which have originally been created with prior RayPack versions.
> If you encounter troubles, please contact our support team, we will gladly assist on upgrading your project files.

IIsControlledPassword
Specifies whether IIs is allowed to set the AnonymousUser_ password.

LogVisits

Specifies whether IIs tracks all access to the directory.

Index
Specifies whether IIs searches the directory.

DefaultDoc
Comma delimited list of file names to act as a default document.

AspDetailedError
Specifies whether detailed ASP errors are sent to browser.

HttpExpires
Value to set the HttpExpires attribute to for a Web Dir in the metabase.

CacheControlMaxAge
Integer value specifying the cache control maximum age value.

CacheControlCustom
Custom HTTP 1.1 cache control directives.

NoCustomError
Specifies whether IIs will return custom errors for this directory.

AccessSSLFlags
Specifies AccessSSLFlags IIS metabase property.

AuthenticationProviders
Comma delimited list, in order of precedence, of Windows authentication providers that IIS will attempt to use: NTLM, Kerberos, Negotiate, and others.

RPIIsWebLog

The RPIIsWebLog table contains the columns shown in the following table.

Column	Type	Key	Nullable
RPIIsWebLog	Identifier	Y	N
Format	Text (s255)	N	N

Columns

RPIIsWebLog
Primary key, non-localized token.

RPIIsWebServiceExtension

The RPIIsWebServiceExtension table contains the columns shown in the following table.

Column	Type	Key	Nullable
RPIIsWebServiceExtension	Identifier (s72)	Y	N
Component_	Identifier (s72)	N	N
File	Text (s255)	N	N
Description	Text (S255)	N	Y
Group	Text (S255)	N	Y
Attributes	Integer (i2)	N	N

Columns

RPIIsWebServiceExtension
Primary key, non-localized token.

Component_
Foreign key referencing Component that controls the WebServiceExtension.

File
File for the web service extension.

Description
Description about the web service extension.

Group
Extension group.

Attributes
Attributes for the extension file.

RPIIsWebSite

The RPIIsWebSite table contains the columns shown in the following table.

Column	Type	Key	Nullable
RPIIsWebSite	Identifier (s72)	Y	N

Column	Type	Key	Nullable
Component_	Identifier (s72)	N	N
Description	Formatted (S255)	N	Y
ConnectionTimeout	Integer (I2)	N	Y
Directory_	Identifier (S72)	N	Y
State	Integer (I2)	N	Y
Attributes	Integer (I2)	N	Y
RPIIsWebAddress_	Identifier (s72)	N	N
RPIIsWebDirProperties_	Identifier (S72)	N	Y
RPIIsWebApplication_	Identifier (S72)	N	Y
Sequence	Integer (I2)	N	Y
RPIIsWebLog_	Identifier (S72)	N	Y
Id	Formatted (S74)	N	Y

Columns

RPIIsWebSite
Primary key, non-localized token for IIsWebSite.

Component_
Foreign key referencing Component that controls the web site.

Description
Description displayed in IIS MMC applet.

ConnectionTimeout
Time connection is maintained without activity (in seconds).

Directory_
Foreign key referencing directory that the web site points to.

State
Sets intial state of web site.

Attributes
Control the install behavior of web site.

RPIIsWebAddress_
Foreign key referencing primary address for the web site.

RPIIsWebDirProperties_
Foreign key referencing possible security information for the web site.

RPIIsWebApplication_
Foreign key referencing possible ASP application for the web site.

Sequence
Allows ordering of web site install.

RPIIsWebLog_
Reject requests after queue gets how large.

Id
Optional number or formatted value that resolves to number that acts as the WebSite Id.

RPIIsWebVirtualDir

The RPIIsWebVirtualDir table contains the columns shown in the following table.

Column	Type	Key	Nullable
RPIIsWebVirtualDir	Identifier (s72)	Y	N
Component_	Identifier (s72)	N	N
RPIIsWebSite_	Identifier (s72)	N	N
Alias	Text (s255)	N	N
Directory_	Identifier (s72)	N	N
RPIIsWebDirProperties_	Identifier (S72)	N	Y
RPIIsWebApplication_	Identifier (S72)	N	Y

Columns

RPIIsWebVirtualDir
Primary key, non-localized token.

Component_
Foreign key referencing the component that controls the virtual directory.

RPIIsWebSite_
Foreign key referencing the web site that controls the virtual directory.

Alias

Name of virtual directory displayed in IIS MMC applet.

Directory_
Foreign key referencing the directory that the virtual directory points at.

RPIIsWebDirProperties_
Foreign key referencing possible security information for the virtual directory.

RPIIsWebApplication_
Foreign key referencing possible ASP application for the virtual directory.

RPLinkedFolders

The RPLinkedFolders table contains the columns shown in the following table.

Column	Type	Key	Nullable
Directory_	Identifier (s72)	Y	N
Feature_	Text (s72)	N	N
SourcePath	Text (S0)	N	N
Wildcard	Text (S0)	N	Y
Attributes	Integer (I2)	N	Y

Columns

Directory_
Directory identifier.

Feature_
The identifier of a feature to which the linked content should be attached

SourcePath
A full or relative source path to the folder on a local drive. The base folder of a relative path is the folder of the current package. The following special syntax is supported:

- $(ProfileName) – the name of the current profile
- (%VariableName) – the value of environment variable `VariableName`

Refer to the reference of table RPSourcePath for more available formatting options.

Wildcard
A wildcard pattern to determine which files should be imported (for example *.`exe`). Null or empty value matches all files.

Attributes
Additional options:
0 = Search only top folder
1 = Search of subfolders recursively

RPModuleSignature

The RPModuleSignature table contains the columns shown in the following table.

Column	Type	Key	Nullable
ModuleID	Identifier (s72)	Y	N
Language	Integer (i2)	Y	N
Feature_	Identifier (S72)	N	Y
ModulePath	Text (S0)	N	Y
ModuleName	Text (S72)	N	Y
Manufacturer	Text (S72)	N	Y
Version	Version (S32)	N	Y

Columns

ModuleID
Module identifier (String.GUID).

Language
Default decimal language of module.

Feature_
The feature to which a given Merge Module has to be merged.

ModulePath
The path to the .msm file containing this MergeModule.

ModuleName
The name of the MergeModule.

Manufacturer
The manufacturer of the MergeModule.

Version
Version of the module.

RPPermissions

The RPPermissions table contains the columns shown in the following table.

Column	Type	Key	Nullable
LockObject	Identifier (s72)	Y	N
Table	Identifier (s72)	Y	N
Sddl	Formatted (S0)	N	Y
Attributes	Integer (I2)	N	N
TemplateProperty	Formatted (S255)	N	Y
InfFilePath	Formatted (S255)	N	Y

Columns

LockObject
Identifier of the object to be locked. References may relate to tables File, Registry, CreateFolder, ServiceInstall

Table
The name of the table to take object from. Valid values: File, Registry, CreateFolder, ServiceInstall

Sddl
The formatted SDDL string defining the set of permissions.

Attributes
The attributes of permissions set.

TemplateProperty
The template for newly created security templates.

InfFilePath
The path where .inf file will be stored.

RPPrerequisite

The RPPrerequisite table contains the columns shown in the following table.

Column	Type	Key	Nullable
Path	Identifier (s0)	Y	N

Column	Type	Key	Nullable
Order	Integer (I2)	N	Y
Content	Text (S0)	N	Y

Columns

Path
A path to the prerequisite descriptor. Certain formatted variables are allowed in this field:

- ($PackPointDir) resolves to the full path of the PackPoint folder
- ($ProfileName) - the name of the current profile
- (%VariableName) - the value of environment variable `VariableName`

Order
The numeric order in which prerequisites are installed. The lower the number the sooner the prerequisite is installed

Content
This field is reserved for a future use.

RPScheduledTasks

The RPScheduledTasks table contains the columns shown in the following table.

Column	Type	Key	Nullable
RPScheduledTasks	Identifier (s72)	Y	N
Component_	Identifier (s72)	N	N
TaskName	Formatted (l255)	N	N
TaskRun	Formatted (l255)	N	N
Arguments	Text (S255)	N	Y
WorkingDirectory	Formatted (L72)	N	Y
RPUser_	Identifier (S72)	N	Y
StartTime	Integer (I4)	N	Y
Schedule	Integer (i2)	N	N
Modifier	Text (S255)	N	Y
Days	Integer (I4)	N	Y
Months	Integer (I4)	N	Y

Column	Type	Key	Nullable
IdleTime	Integer (I4)	N	Y
StartDate	Integer (I4)	N	Y
EndDate	Integer (I4)	N	Y
Attributes	Integer (I4)	N	Y
Description	Text (S255)	N	Y

Columns

RPScheduledTasks
Primary key, name of the scheduled task.

Component_
Foreign key, Component used to determine the task install state.

TaskName
A value that specifies a name which uniquely identifies the scheduled task.

TaskRun
A value that specifies the path and file name of the task to be run at the scheduled time.

Arguments
A value that specifies the command line arguments for the program specified with the TaskRun column.

WorkingDirectory
An optional value that specifies the working directory for the program specified with the TaskRun column.

RPUser_
Foreign key to the RPUser table, defining the user context under which the task runs.

StartTime
A value that specifies the start time to run the scheduled task.

Schedule
A value that specifies the schedule frequency.

Modifier
A value that refines the schedule type to allow for more detailed control over the schedule recurrence.

Days
A value that specifies the day of the week to run the task.

Months
A value that specifies months of the year.

IdleTime
A value that specifies the amount of idle time to wait before running a scheduled ONIDLE task.
The valid range is 1 - 999 minutes.

StartDate
A value that specifies the first date on which to run the task.

EndDate
Specifies the last date that the task is scheduled to run.

Attributes
This column contains a flag for the scheduled task.

Description
The optional comment about the task.

RPShortcutExtended
The RPLinkedFolders table contains the columns shown in the following table.

Column	Type	Key	Nullable
Shortcut_	Identifier (s72)	Y	N
Attributes	Integer (I2)	N	Y

Columns

Shortcut_
Shortcut identifier.

Attributes
Additional options:
0 = No action
1 = Pin to the Taskbar
2 = Pin to the Start Menu
3 = Pin to the Taskbar and Start Menu

RPSourcePath

The RPSourcePath table contains the columns shown in the following table.

Column	Type	Key	Nullable
File_	Identifier (s72)	Y	N
Path	Text (S0)	N	Y

Columns

File_
Name of the file to which a given source path applies.

Path
The path to the source file.

The value in the Path column can contain special syntax that will be resolved to actual values on build. During building, RayPack collects the files pointed by the value from the Path column. All files must be existing in order to successfully build the package.

The value from the Path column should contain the full absolute path to the resource, for example:
`C:\Resources\file.txt`

In order to avoid hardcoding values and provide interoperability of projects, several formatted syntaxes are available. When importing any file, RayPack will automatically try to avoid hardcoding the value by looking up the location of the project and the current environment variables for a best match. If nothing is found, a full absolute path will be used.

The following syntax is available in the Path column:

String	Example & Resolving

`<ProjectFile>`	Example:
	`<ProjectFile>.Sources\fileA.txt`
	
	Given that the project is saved as
	`C:\RayPack\Projects\ProjectA.rpp`
	
	the value will be resolved to:
	`C:\RayPack\Projects\ProjectA.rpp.Sources\file.txt`

`<ProjectFolde`	Example: `<ProjectFolder>\Sources\fileA.txt`

String	Example & Resolving
	Given that the project is saved as `C:\RayPack\Projects\ProjectA.rpp` the value will be resolved to: `C:\RayPack\Projects\Sources\file.txt`

`<%Environment`	Example:
	 `<%WINDIR%>notepad.exe`
	
	 The environment variables are resolved according to the system configuration:
	 `C:\Windows\notepad.exe`

`.\`
 (at the beginning of the path)
 (deprecated)
	Example:
	 `.\fileA.txt`
	
	 Given that the project is saved as
	 `C:\RayPack\Projects\ProjectA.rpp`
	
	 the value will be resolved to:
	 `C:\RayPack\Projects\ProjectA.rpp.Sources\Files\file.txt`
	
	 Note:
	 The support for this syntax is available to keep the compatibility between old projects created in RayPack 1.2 or earlier. It is not recommended to use this syntax anymore.

`($VariableName)`	Uses the path of variable with a given name as defined in RPVariable table.

RPSqlDatabase

The RPSqlDatabase table contains the columns shown in the following table.

Column	Type	Key	Nullable
RPSqlDatabase	Identifier (s72)	Y	N
Server	Text (s72)	N	N
Instance	Text (S255)	N	Y
Database	Text (s255)	N	N
Component_	Text (s72)	N	N
RPUser_	Text (S72)	N	Y
RPSqlFileSpec_	Text (S72)	N	Y
RPSqlFileSpec_Log	Text (S72)	N	Y

Column	Type	Key	Nullable
Attributes	Integer (I2)	N	Y

Columns

RPSqlDatabase
Display name of the Sql Database.

Server
Address of the database server.

Instance
Name of the database Instance.

Database
Name of the Sql Database.

Component_
Required foreign key into the Component Table, specifying the component controlling the processing of sql database.

RPUser_
User used to log into database.

RPSqlFileSpec_
Foreign key referencing RPSqlFileSpec.

RPSqlFileSpec_Log
Foreign key referencing RPSqlFileSpec.

Attributes
1 = create on install
2 = drop on uninstall
4 = continue on error
8 = drop on install
16 = create on uninstall
32 = confirm update existing table
64 = create on reinstall
128 = drop on reinstall

RPSqlFileSpec

The RPSqlFileSpec table contains the columns shown in the following table.

Column	Type	Key	Nullable
RPSqlFileSpec	Text (s72)	Y	N
Name	Text (s255)	N	N
Filename	Text (s255)	N	N
Size	Text (S72)	N	Y
MaxSize	Text (S72)	N	Y
GrowthSize	Text (S72)	N	Y

Columns

RPSqlFileSpec
ID of File specification.

Name
Name of File specification.

Filename
Name of file.

Size
Size of file.

MaxSize
Maximum size of file.

GrowthSize
Growth size of file.

RPSqlReplace

The RPSqlReplace table contains the columns shown in the following table.

Column	Type	Key	Nullable
RPSqlReplace	Text (s72)	Y	N
RPSqlScript_	Text (s72)	N	N

Column	Type	Key	Nullable
Search	Text (S0)	N	Y
Replace	Text (S0)	N	Y
Attributes	Integer (i4)	N	N
Sequence	Integer (i2)	N	N

Columns

RPSqlReplace
ID of Sql Replace object.

RPSqlScript_
Foreign key to the RPSqlScript table

Search
String to search for

Replace
String that replaces the match on Search.

Attributes
1 = Match whole word only
2 = Match case
4 = Replace once only

Sequence
Order in which the text replacements are processed.

RPSqlScript

The RPSqlScript table contains the columns shown in the following table.

Column	Type	Key	Nullable
RPSqlScript	Text (s72)	Y	N
RPSqlDatabase_	Text (s72)	N	N
Component_	Text (s72)	N	N
Source_	Text (S0)	N	Y
RPUser_	Text (S72)	N	Y
Attributes	Integer (i2)	N	N

Column	Type	Key	Nullable
Sequence	Integer (i2)	N	N
SourceType	Integer (i2)	N	N
Target	Text (S0)	N	Y
Description	Text (S0)	Y	N

Columns

RPSqlScript
ID of Sql Script.

RPSqlDatabase_
Foreign key related to RPSqldatabase table used to determine SQL database to execute the script on.

Component_
Foreign key to Component used to determine install state.

Source_
If SourceType = 1 : Foreign key related to Binary table
If SourceType = 2 : Foreign key related to Property table

RPUser_
User used to log into database.

Attributes
1 = execute on install
2 = execute on uninstall
4 = continue on error
8 = rollback on install
16 = rollback on uninstall

Sequence
Order to execute SQL scripts.

SourceType
1 = Binary
2 = Property
3 = Text

Target
SQLScript source if SourceType = 3

Description
The description of the sql script.

RPStreamFiles

The RPStreamFiles table contains the columns shown in the following table.

Column	Type	Key	Nullable
Name	Identifier (s72)	Y	N
Table	Identifier (s32)	Y	N
FilePath	Text (s0)	N	N
Date	Integer (I4)	N	Y
Size	Integer (I4)	N	Y
Attributes	Integer (I2)	N	Y

Columns

Name
ID of stream file.

Table
Stream file resource table type. Valid values: Binary, Icon

FilePath
Path to the file.

> **Note:**
> This field accepts formatted values. Refer to the reference of table RPSourcePath for more information.

Date
Date of last modification.

Size
Size of file.

Attributes
Attribute from refresh action:
0 = refresh inactive
1 = refresh active

RPTextReplacements

The RPTextReplacements table contains information about manipulation tasks on text based files. Data rows within this table are managed by the interfaces of the Visual Designer view TXT changes.

The RPTextReplacements table contains the columns shown in the following table.

Column	Type	Key	Nullable
TextReplacement	Identifier (s72)	Y	N
Sequence	Integer (i2)	N	N
FindValue	Text (S0)	N	Y
ReplaceValue	Text (S0)	N	Y
IncludeFiles	Text (S0)	N	Y
FindValueType	Integer (i2)	N	N
ExcludeFiles	Text (S0)	N	Y
TargetFolder	Text (S72)	N	Y
IncludeSubfolders	Integer (i2)	N	N
Component_	Identifier (s72)	N	Y
RunType	Integer (i2)	N	N
MatchType	Integer (i2)	N	N
ReplaceOnce	Integer (i2)	N	N
Description	Text (s0)	N	N
ReplaceType	Integer (i2)	N	N

Columns

TextReplacement
Name of the text replacement action.

Sequence
Order in which the text replacements are processed.

FindValue
Text specifying the value to be replaced.

ReplaceValue
Text specifying the value to replace the old value.

IncludeFiles
Text specifying the included files (wild cards supported).

FindValueType
Integer number specifying the type of searched value (text, property or regular expression).

ExcludeFiles
Text specifying the excluded files (wild cards supported) or empty to include all.

TargetFolder
Localizable text specifying the path to the folder in which value will be searched.

IncludeSubfolders
Integer number specifying whether sub-folders are included (1) or excluded (0).

Component_
Required foreign key into the Component table, specifying the component controlling the processing of text replacements action.

RunType
Integer number specifying the timing of the text replacement.

MatchType
Integer number specifying the matching of case and whole words.

ReplaceOnce
Integer number specifying whether the replacement will be done only once (1) or as many times as the searched value is found (0).

Description
Is the description of the text replacement action.

ReplaceType
Integer number 0-Replace 1-Append

RPUser

The RPUser table contains the columns shown in the following table.

Column	Type	Key	Nullable
RPUser	Text (s72)	Y	N
Name	Text (s255)	N	N

Column	Type	Key	Nullable
Domain	Text (S255)	N	Y
Password	Text (S255)	N	Y
Attributes	Integer (I2)	N	Y

Columns

RPUser
ID of the user object.

Name
Actual user name.

Domain
Domain of the user profile.

Password
Password of the user profile

Attributes
Attribute options to apply to the user profile.

RPVariable

The RPLinkedFolders table contains the columns shown in the following table.

Column	Type	Key	Nullable
RPVariable	Identifier (s72)	Y	N
Value	Text (S0)	N	Y
Attributes	Integer (i4)	N	N

Columns

RPVariable
Variable identifier.

Value
Variable path.

The value can be referenced in a certain subset of tables (for example RPStreamFiles or RPSourcePath) using the following syntax:

```
($VariableName)
```

Nested variables are supported, for example the following composition is allowed:

RPVariable	Value	Attributes
Variable1	C:\Path1	0
Variable2	($Variable1)\SubFolder	0

Attributes
Reserved for future use.

Help and Support

Request RayPack support

Our Raynet support team gladly assists you on any questions or issues you encounter regarding RayPack. Feel free to sign in and open incidents via our Raynet Support Panel.

Join the RayPack community

The RayPack community resides within our Knowledge Base: http://knowledgebase.raypack.net . Once you have signed up for access to the Raynet support panel, you automatically have access to the Knowledge Base, too.

You will surely come to a point where you would love to suggest a new feature for the future development of RayPack. Maybe you need to find some tips & tricks to hit your target right. The RayPack community is your place for discussing such topics, for sharing and expanding your own experience.

Step in contact with Raynet's professional packaging teams and learn how to polish your packaging activities to a level of highest quality standards. Since Raynet has years and years of experience in the packaging business, we know what to do, and how to do it. Don't row your boat alone when you have the chance to join our RayPack community for free.

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 RayPack training? Ask for dates and locations to find the fitting training occasion. You are highly welcome to step in contact via sales@raynet.de.

Additional Information

Visit www.raypack.net for further information regarding the product and current community incentives. It's also recommended taking a look at additional resources available at the Knowledge Base for Raynet products:
http://knowledgebase.raypack.net.

Raynet is looking forward to receiving your feedback from your RayPack experience. Please contact your Raynet service partner or use our Support Panel to add your ideas or requirements to the RayPack development roadmap!

•rayNET

Raynet GmbH

Technologiepark 20
33100 Paderborn, Germany
T +49 5251 54009-0
F +49 5251 54009-29
info@raynet.de
support@raynet.de

www.raynet.de