

# EMCO MSI Package Builder 10

Copyright © 2001-2023 EMCO. All rights reserved.

# Table of Contents

**Chapter 1: Introduction** ... 4

**Chapter 2: Getting Started** ... 6

Getting to Know the Program Interface ... 6
Overview of MSI Creation Methods ... 6
How Installation Repackaging Works ... 6
Overview of Repackaging Best Practices ... 6
Demo: Repackage EXE to MSI Using Monitoring ... 6
Demo: Create Customized Installations Using Monitoring ... 6
Demo: Repackage Silent Installations Using Wrapping ... 6
Installation Project Editing and Customizing ... 6
Selecting the Monitoring Environment ... 6
How to Create an App-V Package ... 6
How to Create an MSIX Package ... 6
Packages Testing and Troubleshooting ... 6

**Chapter 3: Program Interface Overview** ... 53

Welcome Screen ... 53
Projects View ... 53
Project Details View ... 53
File System View ... 53
Registry View ... 53
Environment Variables View ... 53
Services View ... 53
Assemblies View ... 53
Drivers View ... 53
Printers View ... 53
Windows Firewall View ... 53
Scheduled Tasks View ... 53
Custom Actions View ... 53
Wrapped Packages View ... 53
Log View ... 53
Operations View ... 53
Graphical User Interface features ... 53

**Chapter 4: Installations Repackaging** ... 107

How to Choose a Repackaging Method ... 107
Capture Installations ... 107
Capture System Changes ... 107
Repackaging via Wrapping ... 107
Low-Level Repackaging ... 107
How should I configure the repackaged installations to support an upgrade? ... 107
Repackaging on Windows Sandbox ... 107
Repackaging on a Virtual Machine ... 107

**Chapter 5: Installation Projects** ... 140

Projects Management ... 140
Projects Preparation ... 140
File System Modifications ... 140
Registry Modifications ... 140
Environment Variables Modifications ... 140
Services Modifications ... 140
Side-by-side Assemblies Deployment ... 140
Drivers Deployment ... 140
Printers Deployment ... 140
Windows Firewall Modifications ... 140
Scheduled Tasks Deployment ... 140
Using Custom Actions ... 140
Wrapping Existing Installations ... 140
Capturing Changes ... 140
Creating an MSI Package ... 140
MSI Package Language ... 140
MSI Installation Context ... 140
Signing Packages ... 140
Exporting a Project ... 140
Importing a Project ... 140

**Chapter 6: MSIX Packaging** ... 224

Creating an MSIX/AppX Package ... 224
MSIX Applications Configuration ... 224

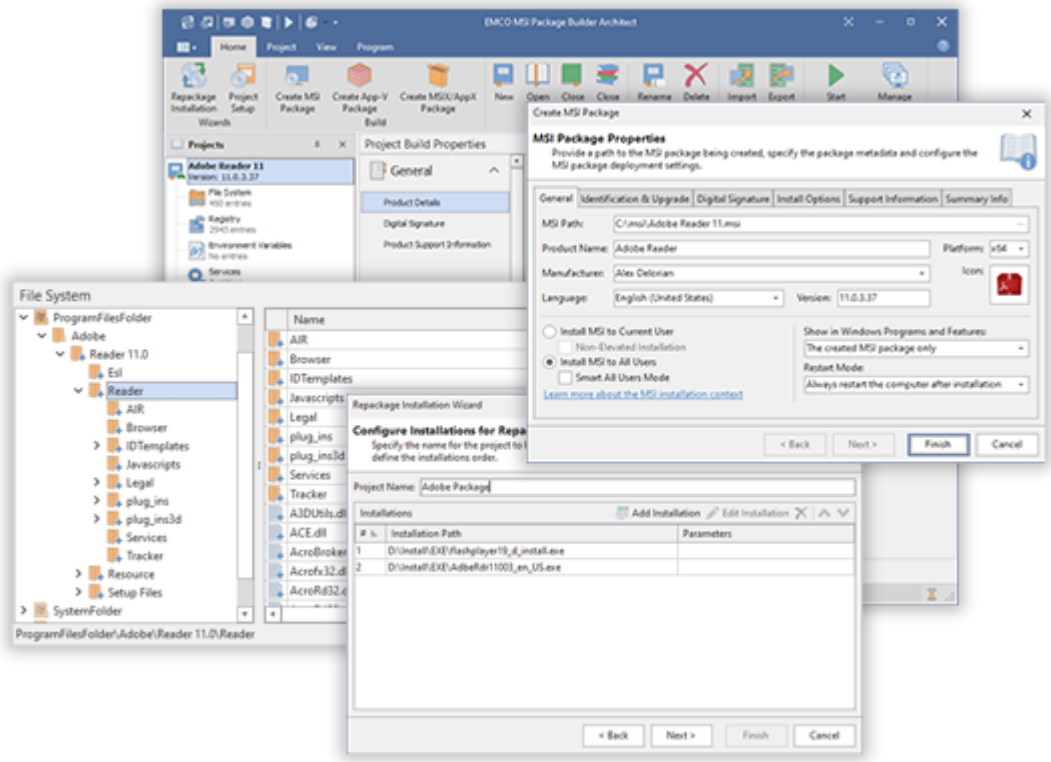
*Copyright © 2001-2023 EMCO. All rights reserved.*

MSIX Fixups ... 224
**Chapter 7: App-V Packaging** ... 245
Creating an App-V Package ... 245
**Chapter 8: Using Placeholders and Filters** ... 249
How should I correctly specify the filter value? ... 249
System Folder Definition Placeholders ... 249
Property Definition Placeholders ... 249
**Chapter 9: Command-Line Interface** ... 257
Monitoring Command ... 257
Package Building and Signing Commands ... 257
Project Import and Export Commands ... 257
**Chapter 10: Log** ... 263
Analyzing Log ... 263
Exporting Log ... 263
**Chapter 11: Program Preferences** ... 266
MSI Package Builder Part ... 266
Filters Part ... 266
Miscellaneous Part ... 266
**Chapter 12: Evaluation of the Program** ... 289
Evaluation Wizard ... 289
Where can I get my License Code? ... 289
How should I formulate the Extended License request? ... 289
**Chapter 13: Program Updates** ... 295
Live Update ... 295
Major Update ... 295
**Chapter 14: Main Program Actions** ... 298
Home Ribbon Page ... 298
Project Ribbon Page ... 298
View Ribbon Page ... 298
Program Ribbon Page ... 298
File System Contextual Ribbon Page ... 298
Registry Contextual Ribbon Page ... 298
Environment Variables Contextual Ribbon Page ... 298
Services Contextual Ribbon Page ... 298
Assemblies Contextual Ribbon Page ... 298
Drivers Contextual Ribbon Page ... 298
Printers Contextual Ribbon Page ... 298
Windows Firewall Contextual Ribbon Page ... 298
Scheduled Tasks Contextual Ribbon Page ... 298
Custom Actions Contextual Ribbon Page ... 298
Wrapped Packages Contextual Ribbon Page ... 298
**Chapter 15: Requirements** ... 323
**Chapter 16: Edition Upgrade** ... 324
**Chapter 17: How can I leave my Feedback?** ... 325
**Chapter 18: About EMCO Software** ... 327
**Chapter 19: Contact Information** ... 328

*Copyright © 2001-2023 EMCO. All rights reserved.*

# Chapter 1: Introduction

Welcome to EMCO MSI Package Builder. The program is designed to help you with repackaging of non-silent installations into silent MSI packages, virtual App-V packages and MSIX/AppX packages. This allows you to deploy repackaged installations remotely using any of software distribution tools, Microsoft Application Virtualization technology, Windows Store, sideloading or local distribution network. You can also use the program to create silent custom installations quickly and easily. This manual provides you with a detailed description of the program's features and shows you how to use these features in practice.




Copyright © 2001-2023 EMCO. All rights reserved.

## Downloading the Program

You can download the program on the *Downloads* page of the website. The download includes a free 30-days trial of the Architect edition of the program. It provides access to all the available features. You can compare the features of the different editions on the *Compare Editions* page of the website.

You can use an evaluation version during 30 days. After this period, you have to register the program to continue using it. MSI packages generated by an evaluation copy have some limitations: they can be deployed during 30 days following their generation date and are displayed with the evaluation mark in Windows **Programs and Features** after being deployed.

## Using Documentation

The program is designed to provide you with an easy and straightforward approach to repackaging third-party installations and creating customized installations in the MSI, App-V and MSIX/AppX formats. To repackage an installation, you just need to follow the steps of the repackaging wizard, and you will get an automatically generated MSI, App-V or MSIX/AppX in the end. It means that you can use the program without referring to the documentation.

If you wish to get an overview of the program features, to understand how the program works and to see how to use the main features in practice, you can read the **Getting Started** guide. To get more information about the program features, you can refer to the corresponding chapters of this manual.

## Getting Help

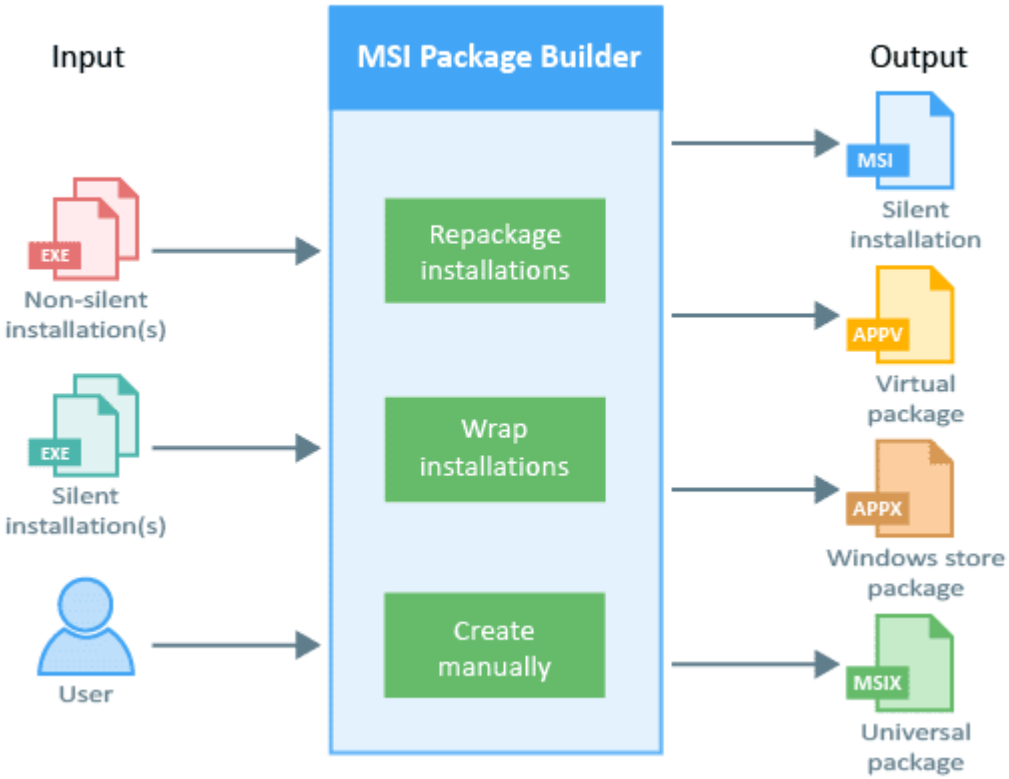
If you see that a generated MSI, App-V or MSIX/AppX package doesn't work as required, first you need to check if the repackaging best practices were followed during the repackaging since the repackaging results depend on the compliance with the repackaging requirements and other aspects. You can learn more about this in the Overview of the Repackaging Best Practices chapter. You can also refer to MSI Packages Testing and Troubleshooting to learn about the general troubleshooting recommendations.

To get help on a problem, you can also *contact support*. Please send us the problem details including the name and the version of the installation you try to repackage, the OS name and the platform (x86 or x64) of your repackaging environment, and the problem description. This information should help the support team to reproduce the problem and provide you with troubleshooting instructions.

*Copyright © 2001-2023 EMCO. All rights reserved.*

# Chapter 2: Getting Started

EMCO MSI Package Builder is a packaging tool that allows you to repackage non-silent installations into silent MSI packages and create custom silent installations quickly and easily. The program uses a unique repackaging technology that allows converting almost any non-silent installation into a silent MSI to enable its automatic deployment using Group Policy or other software distribution tools. The same technology can be used for automatic creation of customized MSI packages. If you need to repackage applications into App-V or MSIX/AppX, the program can cope with this task as well.




Using EMCO MSI Package Builder, you can benefit from the following features that are demonstrated in the course of this tutorial.

- **Multiple MSI packaging methods**. Depending on the project, you can use one of the provided packaging methods that are suitable for that project. You can convert non-silent installations into a silent MSI automatically using Live Monitoring. In the specific cases, when Live Monitoring cannot be used, you can wrap installations into an MSI package if those installations can be deployed silently.
- **Best-in-class repackaging technology**. The program allows you to repackage a non-silent installation of any complexity into a silent MSI package. The program supports repackaging of installations that perform complex changes such as modifying file system and registry permissions, installing Windows services and drivers, changing Windows environment variables, etc.
- **Automatic MSI creation**. The program allows you to repackage EXE to MSI automatically: you just need to go through the deployment steps of the original EXE installation, and the program will capture all the performed changes on the fly and generate an MSI for you. You can also use

*Copyright © 2001-2023 EMCO. All rights reserved.*

the Live Monitoring technology to monitor any activity on a PC and generate an MSI reproducing the captured changes.

- **Easy MSI customization**. Having created an MSI package, you can edit it, if required. You can have an access to all captured changes to modify them or add additional changes. You can also configure the MSI package to execute any scripts or executable files before and after the MSI installation.
- **Repackaging on virtual machines**. You can use Windows Sandbox running on the local machine for repackaging installations in a clean environment. Windows Sandbox is available on the modern Windows versions, it doesn't require any setup and allows you to repackage installations in an isolated, clean OS environment.
- **Repackaging on Windows Sandbox**. You can use Windows Sandbox running on the local machine for repackaging installations in a clean environment. Windows Sandbox is available on the modern Windows versions, it doesn't require any setup and allows you to repackage installations in an isolated, clean OS environment.
- **App-V packaging**. In addition to generating MSI packages, the program can also create App-V packages. The procedure of repackaging into virtual packages is the same as repackaging into MSI, so you only need to change the output format. Moreover, if you have a previously stored project, you can open it and generate an App-V package without repeating the repackaging process.
- **MSIX/AppX packaging**. Together with generating classic MSI packages and App-V virtualization packages, the program can also create MSIX (targeting Windows 10 October 2018 Update or later) or AppX (earlier versions of Windows 10) packages, that can be distributed through Windows Store, installed by sideloading or distributed in local networks.

In the following chapters, you will learn about different packaging methods offered by the program and will see how to use them in practice. In addition, you will get a brief overview of the installation repackaging internals and best practices. Finally, you will learn how to test generated packages and how to troubleshoot packaging problems.

## What's Inside

Getting to Know the Program Interface
Overview of MSI Creation Methods
How Installation Repackaging Works
Overview of Repackaging Best Practices
Demo: Repackage EXE to MSI Using Monitoring
Demo: Create Customized Installations Using Monitoring
Demo: Repackage Silent Installations Using Wrapping
Installation Project Editing and Customizing
Selecting the Monitoring Environment
How to Create an App-V Package
How to Create an MSIX Package
Packages Testing and Troubleshooting

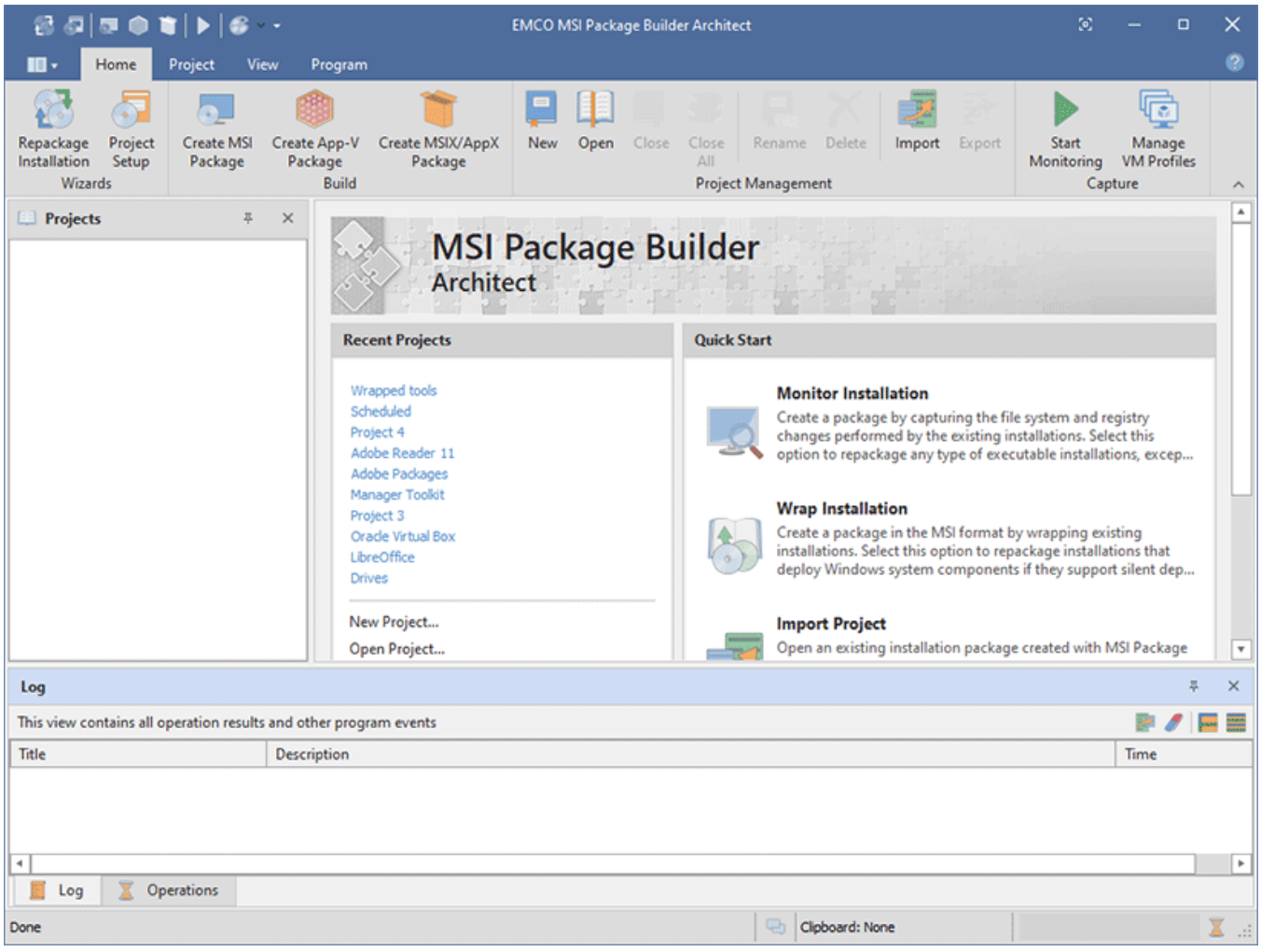
*Copyright © 2001-2023 EMCO. All rights reserved.*

## Getting to Know the Program Interface

When you start EMCO MSI Package Builder the first time, you can see the main screen of the program and the **Repackage Installation** wizard displayed at the top of it. The main goal of the program is installation repackaging, so the wizard is designed to guide you through the repackaging process. In the wizard, you can select the required repackaging method, choose the installations to be repackaged, configure the deployment package settings, etc. In the next chapter, you will learn how to select the right repackaging method in the wizard and you will see how to use the wizard in practice in the following chapters.

When the **Repackage Installation** wizard is closed and no installation projects are opened, you can see the starting main program screen Pic 1. The program has a Ribbon user interface, so the Ribbon menu is displayed at the top of the screen providing you with access to the main actions of the program. Different types of the actions are located on different Ribbon tabs, so you can switch them, if required. Additional Ribbon tabs are displayed automatically when specific elements are selected and provide you access to the contextual actions.

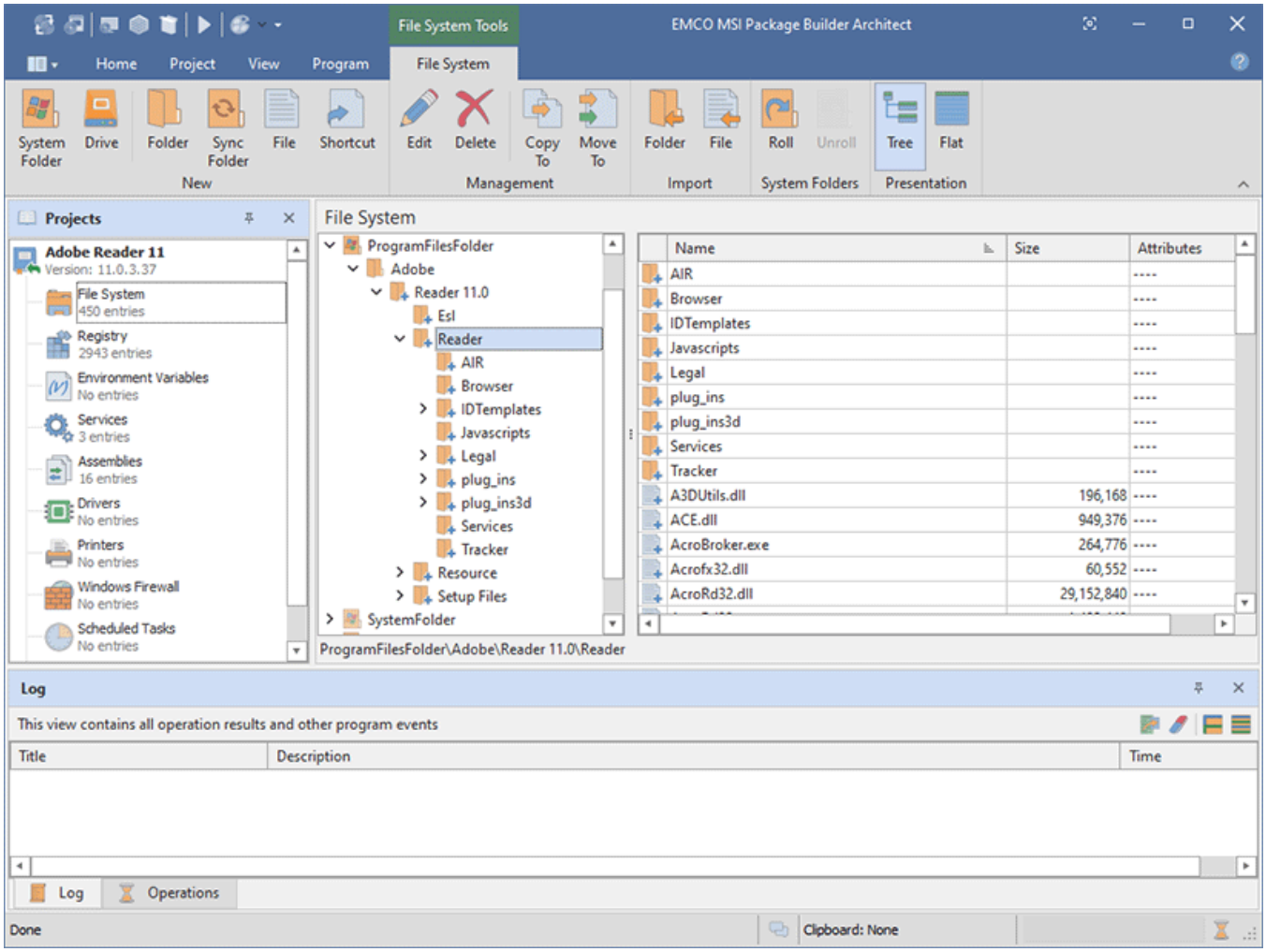
Under the Ribbon, you can see opened projects on the left and the main area on the right. When you start the program, the **Projects** view is empty and the main area displays a list of the recent projects and the main actions of the program. You can click the main actions to initiate the corresponding repackaging wizards. You can open the **Repackage Installation** wizard to access all repackaging options by clicking **Repackage Installation** on the Ribbon.




**Pic 1. The starting main screen**

*Copyright © 2001-2023 EMCO. All rights reserved.*

If you have completed repackaging or opened previously created installation projects, you can see the current projects in the **Projects** view. You can have more than one project opened to be able to copy data between the projects. A project consists of different types of resources. If you select a project node in the **Projects** view, you can see and edit the project settings. If you select, for example, the **File System** node, you can review and modify files and folders in an installation project using the file system editor Pic 2. There are different editors for managing different types of resources, so you need to select the required item in the **Projects** tree to review and modify its content using the corresponding editor.




**Pic 2. The program screen with an opened project**

At the bottom of the main screen, you can see the **Log** and **Operations** views. The **Log** view reports status information for deployment packages generation and other operations. You can use this view to find error messages and troubleshooting information in case of a problem. The **Operations** view displays the currently running operations and allows you to manage them.

You can learn more about the available views and editors in the **Program Interface Overview** chapter where you can find detailed explanations of features provided by every view.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Overview of MSI Creation Methods

EMCO MSI Package Builder is designed to create silent MSI packages, i.e. packages that can be deployed without interacting with a user. Such packages can be deployed manually on a local PC or remotely across a network using any remote software distribution tool.

The primary feature of EMCO MSI Package Builder is repackaging of non-silent installations into silent MSI packages. This allows you to convert non-silent installations, which cannot be deployed remotely, into MSI packages that are ready for remote deployment. You can use the program in many other cases, for example, to create custom silent installations or to combine multiple installations into a single MSI package. The program offers different approaches to creating an MSI to be used in different cases, so you need to know all available options and their features to select the right approach for every case.

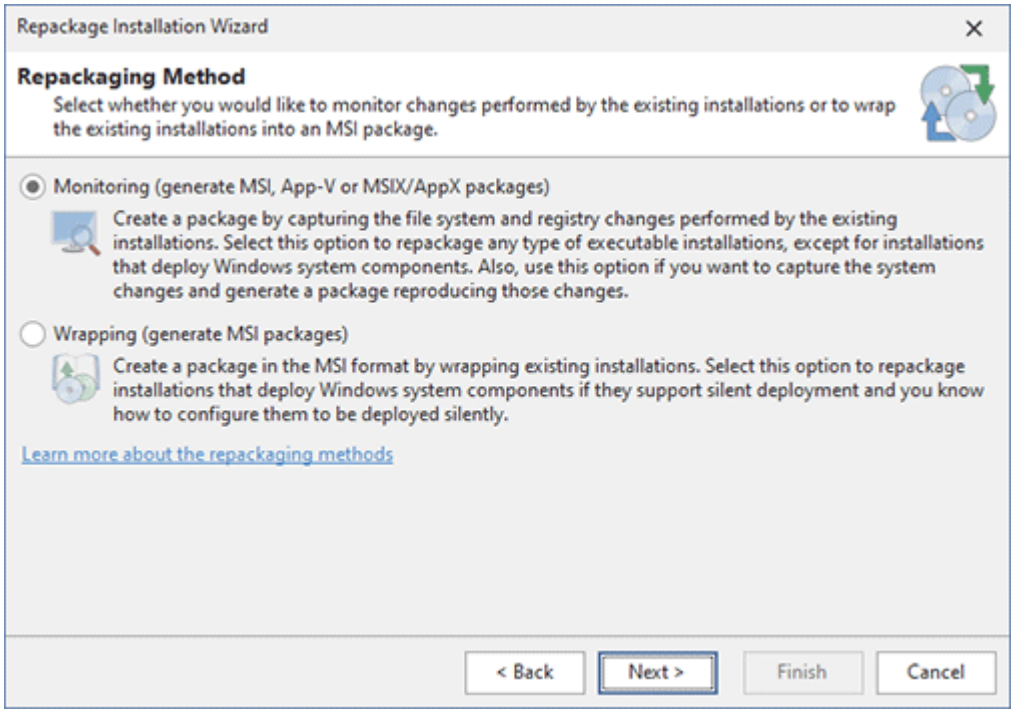
## What is Monitoring, Wrapping and Manual Creation?

The primary method of MSI creation is monitoring, which is available in all editions of the program. Using this method, you can repackage non-silent installations into silent MSI packages and create custom silent installations according to your needs. It uses the Live Monitoring technology to capture all changes performed under Windows to create an MSI reproducing such changes. You can learn how it works in the How Installation Repackaging Works chapter.

The other MSI creation methods are wrapping and manual creation. Wrapping is available in the Enterprise and Architect editions of the program. It can be used to repackage Windows system installations that cannot be repackaged using monitoring. This method allows including one or multiple silent installations in the EXE or MSI format into an MSI package, so the included installations are deployed silently when you deploy the wrap MSI. Besides, you can create silent MSI packages manually through visual editors by specifying the changes to be included into an MSI.

How to use these MSI creation methods in the program? When you start the program for the first time, the **Repackage Installation** wizard is displayed automatically Pic 1. In the Enterprise and Architect editions of the program, as the first step, you need to select if you would like to use monitoring or wrapping. In the Professional edition, you cannot use wrapping, so you need to select the capturing type that will be described later in this chapter. If you wish to create an MSI manually though editors, you can close the wizard and create a new project using the Ribbon options.

Copyright © 2001-2023 EMCO. All rights reserved.




**Pic 1. Selecting a repackaging method**

It is recommended that monitoring be used in most cases since it is a universal method, and wrapping or a manual MSI creation should be considered only if you have good reasons to do so. Below, you can find some examples demonstrating when using different types of the MSI creation methods makes sense.

**Monitoring usage examples**:

- Repackage one or multiple non-silent installations into a silent MSI.
- Create a silent MSI package that includes a repackaged non-silent installation and its customization (the installed product registration, configuration, etc.).
- Create a custom silent MSI that includes any captured changes.

**Wrapping usage examples:**

- Repackage an installation of a Windows component, such as .NET Framework, for example, that cannot be repackaged using monitoring. The repackaged installation should support silent deployment.
- Wrap one or multiple EXE/MSI installations that can be deployed silently into a silent MSI.

**Custom creation examples**:

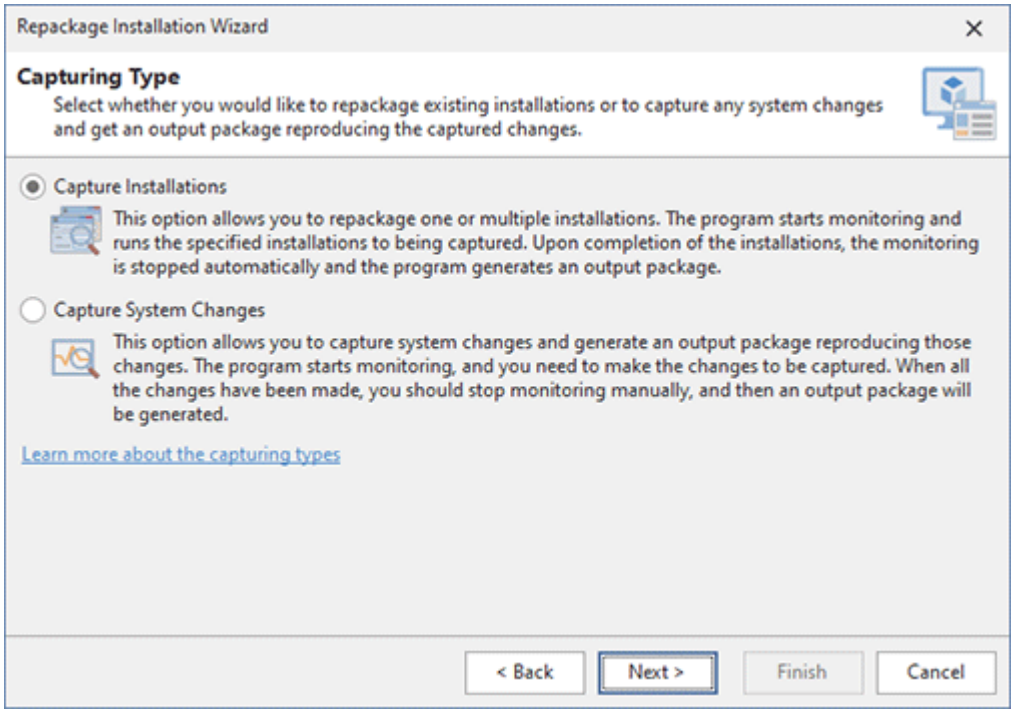
- Create a simple MSI that deploys only a few resources.

Note that you can use monitoring not only for repackaging but also for creating custom installations by capturing the changes performed manually. It is much easier to use that approach than to create an MSI manually using visual editors. You can see how to use monitoring in practice in the demos included in the Repackage EXE to MSI Using Monitoring and Create Customized Installations Using Monitoring chapters. Wrapping is demonstrated in the demo contained in the Repackage Silent Installations Using Wrapping chapter.

Copyright © 2001-2023 EMCO. All rights reserved.

## How to Select the Capturing Type

You can use monitoring in a number of cases, for instance, to repackage an installation or to repackage an installation together with its post-install customization. To support all those cases, monitoring can work in two options, which are **Capture Installations** and **Capture System Changes** that you can select in the **Repackage Installation** wizard Pic 2.




**Pic 2. Selecting the capturing type**

When you select any of these options, EMCO MSI Package Builder uses the Live Monitoring technology to capture the file system and registry changes and create an MSI reproducing such changes. You will learn how it works in the next chapter. To understand which method you need to select, you can review their comparison below.

| | Capture Installations | Capture System Changes |
|---|---|---|
| How repackaging works | In the wizard, you specify a non-silent installation that you need to repackage, and EMCO MSI Package Builder automatically starts monitoring system changes and runs the specified installation. You follow the installation deployment steps, and after the deployment is finished, EMCO MSI Package Builder automatically stops monitoring and generates an MSI package that includes the captured changes performed by the monitored installation. | As the last step of the wizard, EMCO MSI Package Builder automatically starts monitoring system changes. You manually run one or several installations, apply the required software customization or perform any changes that should be captured and included into the MSI to be generated. Once you are done, you need to switch back to EMCO MSI Package Builder and stop monitoring manually. EMCO MSI Package Builder automatically generates an MSI package that includes the captured changes reproducing the activity you performed during monitoring. |

Copyright © 2001-2023 EMCO. All rights reserved.

| | Capture Installations | Capture System Changes |
|---|---|---|
| Requirements | You provide an installation to be repackaged.<br>You follow the deployment steps of the repackaged installation by selecting the install options manually. | You run installations and/or perform any changes you need to capture manually.<br>You manually stop monitoring in the program. |
| What it can be used for | To repackage one or multiple non-silent installations into a silent MSI package. | To repackage installations that require pre/post customizations, such as software registration or tuning of the settings.<br>To capture any custom changes performed manually or by any executable or script to create an MSI reproducing those changes. |
| Examples | Repackage a non-silent installation of *Open Office* into a silent MSI. | Repackage a non-silent installation and its registration after deployment.<br>Create a silent MSI to install fonts. |

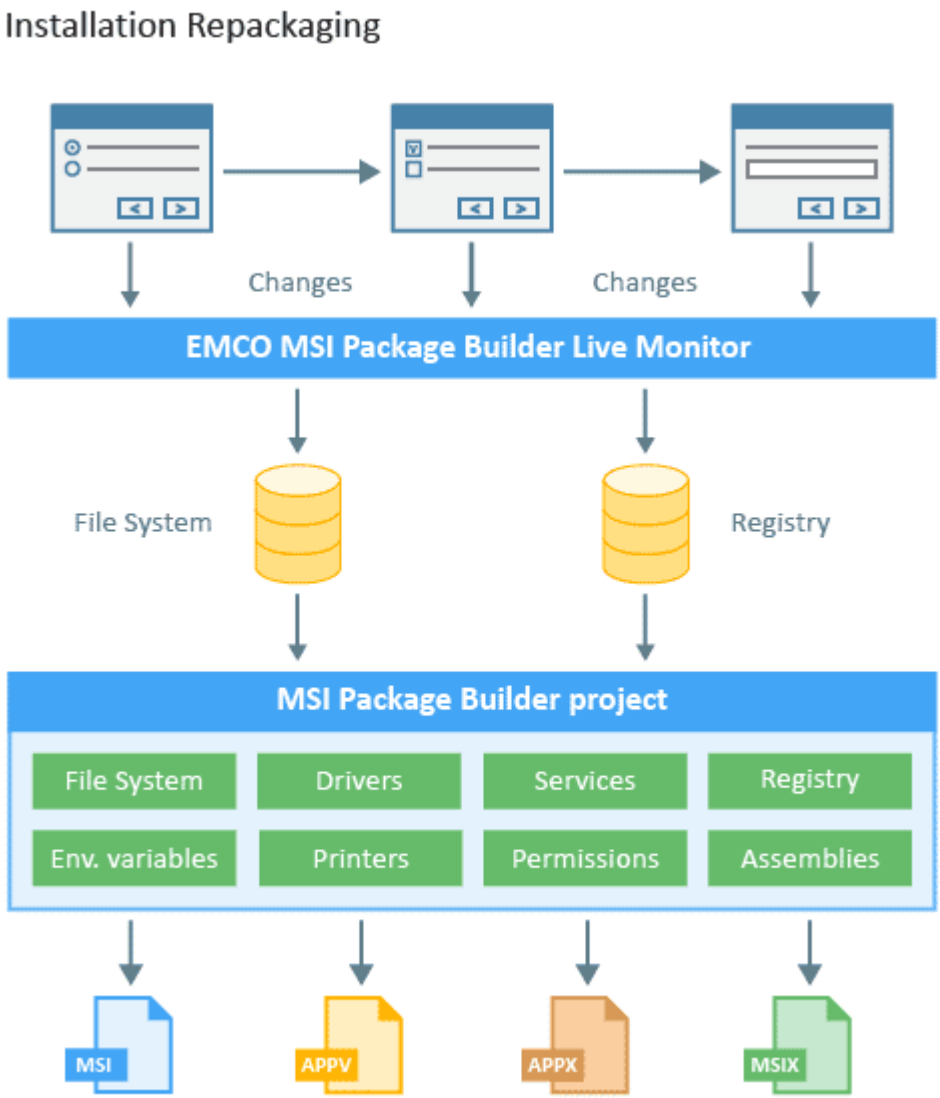
In the next chapter, you can learn how monitoring works and what advantages and limitations it has. In addition, you can find recommendations that will help you to use monitoring successfully in the Overview of the Repackaging Best Practices chapter.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## How Installation Repackaging Works

Installations monitoring is the primary repackaging method available in EMCO MSI Package Builder that allows you to convert almost any non-silent EXE installation into a silent MSI package. Usually, this kind of repackaging is required to deploy software on remote PCs, because only silent installations can be deployed remotely.

EMCO MSI Package Builder incorporates the best-in-class Live Monitoring technology used for repackaging. It integrates into Windows to capture file system, registry and permission changes performed by the repackaged installation. During repackaging, you just need to follow the installation steps of the repackaged installation manually. After you select the installation options, the monitored installer applies the corresponding file system, registry and permission changes that are captured by EMCO MSI Package Builder.




Copyright © 2001-2023 EMCO. All rights reserved.

Once the installation is completed, EMCO MSI Package Builder automatically generates an MSI package that includes the captured changes, so the MSI installation is identical to the original installation and includes all the customizations you applied during the monitoring process. When you install the MSI, it applies the same changes that were applied when you deployed the original installation manually.

After the installation is repackaged, you can review the changes captured by EMCO MSI Package Builder. The program intellectually interprets raw-data file system and registry changes into Windows objects, so you can see, for example, Windows services and drivers that were deployed by the monitored installation and edit them through visual editors, if required, to generate a modified MSI.

## Advantages of the Live Monitoring Technology

The Live Monitoring technology used in EMCO MSI Package Builder is the most capable installation repackaging technology available on the market today. It has the following features that allow you to perform repackaging quickly and easily.

- **Successful repackaging of simple and complex installations**. It does not matter what kind of changes are performed by the repackaged installation - EMCO MSI Package Builder can repackage anything, including complex Windows drivers, services, permissions, etc.
- **Changes are captured on the fly**. An MSI package is generated immediately after the monitored installation is finished. Using other repackaging solutions, you need to make Windows snapshots before and after the installation to compare them, so an MSI generation takes too long.
- **Monitoring is started and stopped automatically**. The program tracks the system information and knows when monitoring should be started and stopped without the risk of losing important data.
- **Repackaging is automated from start to finish**. You just follow a simple wizard and get a generated MSI at the end.
- **Unwanted changes are filtered out**. The program has a set of filters to avoid capturing changes performed by unrelated processes. Any captured change is associated with the process that generated it, so unwanted changes are removed automatically.

It is very easy to repackage an installation using the Live Monitoring technology. You can see how to use it in practice in the Repackage EXE to MSI Using Monitoring and Create Customized Installations Using Monitoring chapters.

## Repackaging Limitations

EMCO MSI Package Builder allows you to repackage almost any installation, but still here are a few cases when repackaging using the Live Monitoring technology is not possible.

- **Windows cloning**. It is impossible to repackage Windows itself, because the repackaging technology is designed to be used for applications only. There are third-party tools that can clone Windows images, so you can use them if needed.
- **Installation of Windows components**. During repackaging, EMCO MSI Package Builder captures file system and registry changes, but when you install a system component such as .NET Framework, for example, it installs resources into protected areas of the file system and registry, so it is not possible to reproduce those changes. In this case, if the repackaged installation supports silent deployment, you can use the wrapping approach to repackage the installation.

*Copyright © 2001-2023 EMCO. All rights reserved.*

- **Installation of system software that modifies Windows on reboot**. When Windows is modified on reboot, it is not possible to capture and reproduce those changes, so use wrapping instead or repackaging for installations that apply such changes.

The limitations listed above are common for all repackaging tools, and there is no way to bypass them using the Professional edition of the program. The Enterprise and Architect editions allow you to use the wrapping method that should be used instead of monitoring to avoid the above limitations. The Architect edition of the program can also generate App-V packages if you need to repackage applications into App-V. The recently introduced MSIX/AppX package formats are also supported by the Architect edition of the program.

## Overview of Repackaging Best Practices

The installation repackaging process is based on monitoring file system and registry changes performed by the monitored installation as explained in the How Installation Repackaging Works chapter. Depending on the computer state before and during the repackaging process, you may obtain either correct or incorrect repackaging results. It is recommended that a special environment be used and the repackaging best practices explained below be followed to achieve correct repackaging results.

## How to Create a Clean Environment for Monitoring

It's important to use a clean environment for monitoring to prevent the appearance of any unwanted changes in the monitoring results. A clean environment doesn't have any third-party software installed that can generate file system, registry and other OS changes that may appear in the monitoring results.

Follow the recommendations below to create a clean environment.

- **Use a fresh Windows installation**. It's recommended to use a brand-new Windows installation where no additional software is installed.
- **Disable an anti-virus, firewall and Windows updates**. The system software running in the background may generate changes that will appear in the created package. To avoid that, it is recommended that Windows updates be disabled. An anti-virus, a firewall and other security software may also produce unwanted changes and might cause an unstable or incorrect monitoring operation, so they should be disabled as well.
- **Close all running programs**. You need to close all running programs to avoid monitoring of the changes they produce.

Copyright © 2001-2023 EMCO. All rights reserved.

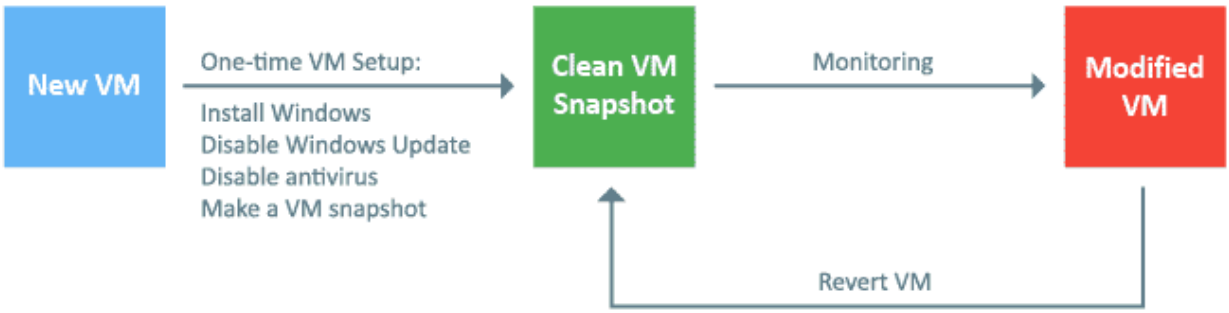
## Use Windows Sandbox for Monitoring

The simplest option to use a clean environment is monitoring in Windows Sandbox. Windows Sandbox is a part of Windows and it is available on Windows 10 or later OS. To use it you need to turn on virtualization capabilities in BIOS on your machine and enable Windows Sandbox in the list of Windows Features. Every time you run Sandbox, it runs a brand-new clean Windows that works in an isolated environment. This clean environment doesn't require any setup, it is available immediately. A sandbox is temporary. When it's closed, all the software and files and the state are deleted.

Note that some software installations cannot be installed in Windows Sandbox, because Sandbox environment has some limitations comparing with the regular environment. If repackaged software fails to deploy in Sandbox, you can monitor it on a virtual machine.

## Use a Virtual Machine for Monitoring

Using a virtual machine (VM) allows you to create a reusable clean environment for monitoring. A virtual machine can run a regular Windows environment that doesn't have any limitations of Windows Sandbox, so it's recommended to use a virtual machine (VM) for monitoring. When you use a VM, it is possible to create a snapshot of the machine and return to the same state repeatedly. To follow the repackaging best practices, you should create a VM, set up a clean environment there as explained above and make a snapshot of the clean VM state.




Every time you need to perform monitoring, you should revert the VM to the snapshot with the clean state. The VM revert works quickly and allows you to use a clean environment for every repackaging operation.

To optimize the monitoring process, it's recommended to use "online" snapshots. These snapshots are with running Windows and logged-on user, so you don't have to start Windows on the VM and to log in manually every time you need to repackage an installation.

Depending on the used edition of EMCO MSI Package Builder and the monitoring environment selected in the program, there are different approaches to configuring and using a VM.

- If you use the Professional edition of the program, which allows monitoring on the local computer only, you can use a VM for monitoring. Configure the VM as explained above and install EMCO MSI Package Builder on the same VM to perform monitoring locally. In this case, you should run EMCO MSI Package Builder on the VM, perform repackaging and revert the VM to the clean state after repackaging.

Copyright © 2001-2023 EMCO. All rights reserved.

- When you select the **Monitor on This Computer** option (available in the Enterprise and Architect editions of the program), you can use a VM for monitoring. Configure the VM as explained above and install EMCO MSI Package Builder on the same VM to perform monitoring locally. In this case, you should run EMCO MSI Package Builder on the VM, perform repackaging and revert the VM to the clean state after repackaging.
- When you select the **Monitor on an Existing Virtual Machine** option (available in the Enterprise and Architect editions of the program), a VM is connected remotely by the program to perform monitoring. You can configure the VM as explained above and run EMCO MSI Package Builder on your machine and use the VM for monitoring.



You can benefit from the repackaging automation using the **Monitor on an Existing Virtual Machine** option available in the Enterprise and Architect editions of the program. In this case, the program automatically maintains the clean state of the VM. You can learn more about all advantages of this approach in the Selecting the Monitoring Environment chapter.

## How to Perform Repackaging If You Don't Have a Virtual Machine or Sandbox

Using a clean environment for repackaging is important to get correct repackaging results. The repackaging process can be simplified when you use a VM, because you can quickly and easily restore the clean VM state utilizing VM snapshots. Alternatively you can use Windows Sandbox that runs an isolated clean machine and doesn't require any setup. If you plan to perform repackaging regularly, it's strongly recommended to consider creating a VM.

If for some reason you cannot use a VM or Sandbox, you need to follow the repackaging best practices listed in the next section, as well as some additional steps, namely:

- **Close all running applications and processes**. You should close all running applications and processes except EMCO MSI Package Builder before you start repackaging since they are not related to the repackaged installation and may cause changes that will appear in the repackaging results.
- **Uninstall the monitored application**. If an installation you need to repackage is already installed, uninstall it with all its components and dependencies.

Note that even if these recommendations are complied with, the monitoring results may not be as precise as those produced in a clean environment.

## Repackaging Best Practices

When repackaging installations, you should always keep in mind that it is strongly advised:

- **Use a clean environment**. It's important to use a clean environment for repackaging to prevent the appearance of any unwanted changes in the repackaging results. Use a VM for repackaging and create a clean repackaging environment as explained above.
- **Monitor and deploy on the same platform**. When you deploy the same installation on different platforms (x86 and x64), it can create different resources, so you should monitor the installation on the same platform and, preferably, under the same OS that will be used for deployment of the package to be generated. If you need to deploy an installation on both the x86 and x64 platforms, you should create two separate packages for these platforms.
- **Prevent an auto-start of the repackaged application after its installation**. When you use the Capture Installations option for monitoring, EMCO MSI Package Builder automatically starts monitoring before running the repackaged installation and stops monitoring when the installation

Copyright © 2001-2023 EMCO. All rights reserved.

is finished. Some installations prompt you to run the installed application as the final step of the deployment, which prevents EMCO MSI Package Builder from detecting the end of the installation and stopping the monitoring process at the right place automatically. You have to skip running the installed application in the course of the installation process. If the application starts automatically after the installation, just close it to notify EMCO MSI Package Builder of the end of the installation process.

- **Skip reboots**. If the monitored installation requires a reboot, you should stop monitoring and generate a deployment package before rebooting.

Usually, repackaging problems are caused by non-compliance with one or more of the best practice recommendations, so if you get a deployment package that does not work properly, make sure you use a clean environment for repackaging and follow other best practices.

*Copyright © 2001-2023 EMCO. All rights reserved.*

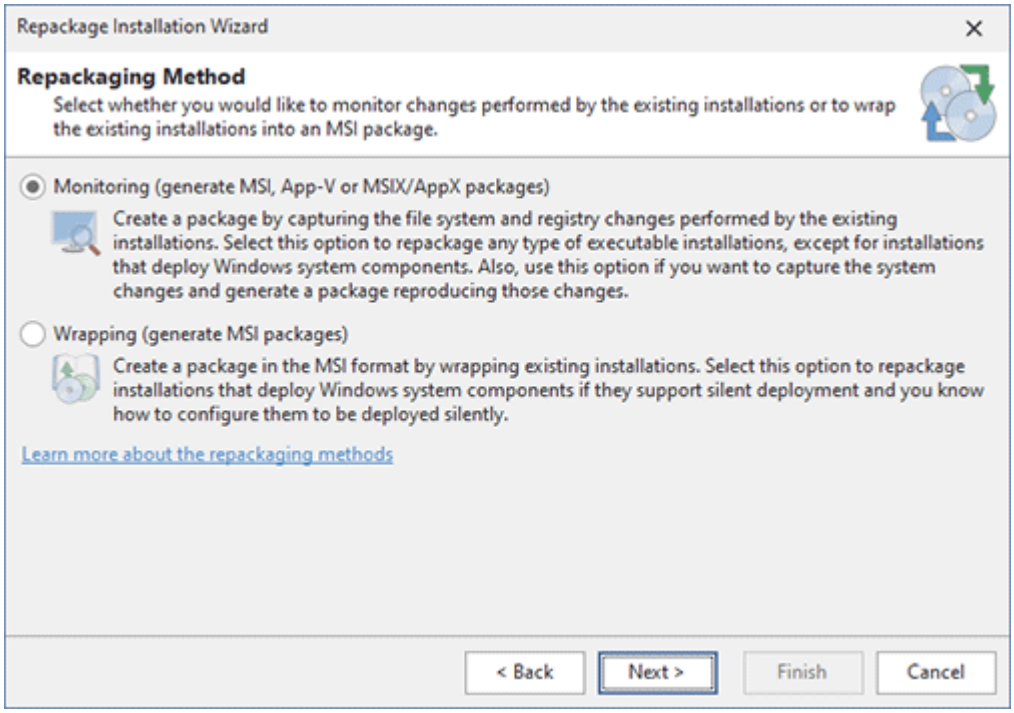
## Demo: Repackage EXE to MSI Using Monitoring

Repackaging of existing non-silent EXE installations into silent MSI packages is one of the main features of EMCO MSI Package Builder, so let's see how to use it. Let's assume we need to distribute *Open Office* across our entire network. When you run the *Open Office* installation, it requires that you select the installation options and the packages to be deployed. You cannot use the original installation for remote deployment because it is not silent. To prepare the installation for remote deployment, we can repackage it into a silent MSI package using EMCO MSI Package Builder.

In this demo, we only need to repackage an *Open Office* installation without any pre- and post-install customization, so, as you learned in the Overview of the MSI Creation Methods chapter, we can use the **Capture Installations** option for repackaging and follow the steps below.

**Step 1. Open the Repackage Installation wizard**
The **Repackage Installation** wizard is automatically opened when you start EMCO MSI Package Builder. If you do not see it on the screen, press the **Repackage Installation** button on the **Home** tab in the Ribbon. When the wizard is started, read the welcome information and press the **Next** button.

**Step 2. Select the Monitoring option**
The wizard allows you to repackage an installation using either monitoring or wrapping. To repackage a non-silent installation into an MSI package, select the **Monitoring** option and press the **Next** button Pic 1.

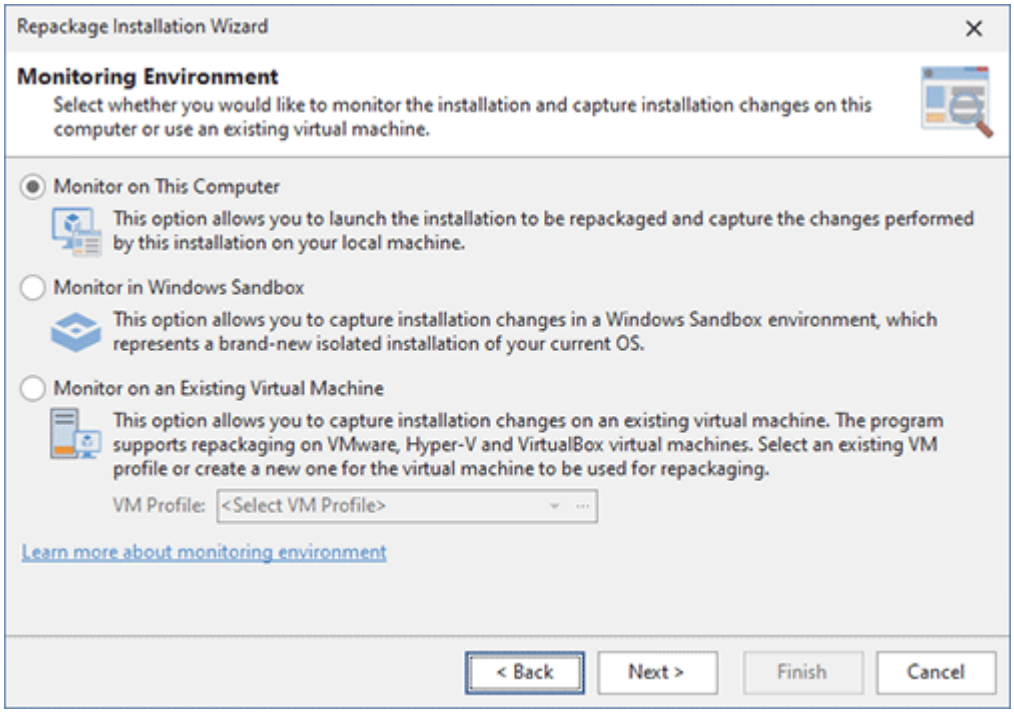



**Pic 1. Select the repackaging method**

Copyright © 2001-2023 EMCO. All rights reserved.

**Step 3. Select the Monitor on This Computer option**

The program allows monitoring an installation on the local machine, on Windows Sandbox or on a virtual machine. In this demo, we will perform repackaging on the local machine, so select the **Monitor on This Computer** option. You can learn how to monitor on Windows Sandbox or on a virtual machine in the Selecting the monitoring environment chapter.

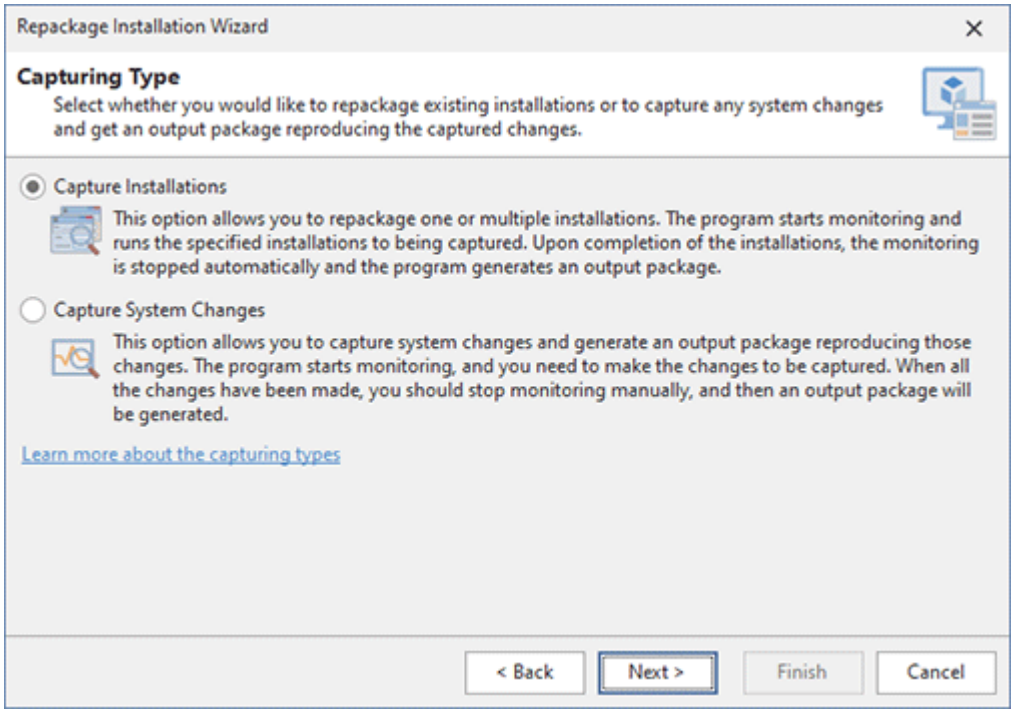



**Pic 2. Select monitoring environment**

*Copyright © 2001-2023 EMCO. All rights reserved.*

### Step 4. Select the Capture Installations option

As you learned in the Overview of the MSI Creation Methods chapter, you can convert a non-silent EXE into a silent MSI package using the **Capture Installations** option, so select this option in the displayed dialog Pic 3. In this case, the program will automatically stop the monitoring and generate an MSI once the monitored installation is completed. Press the **Next** button to proceed to the next step.




**Pic 3. Select the capturing type**

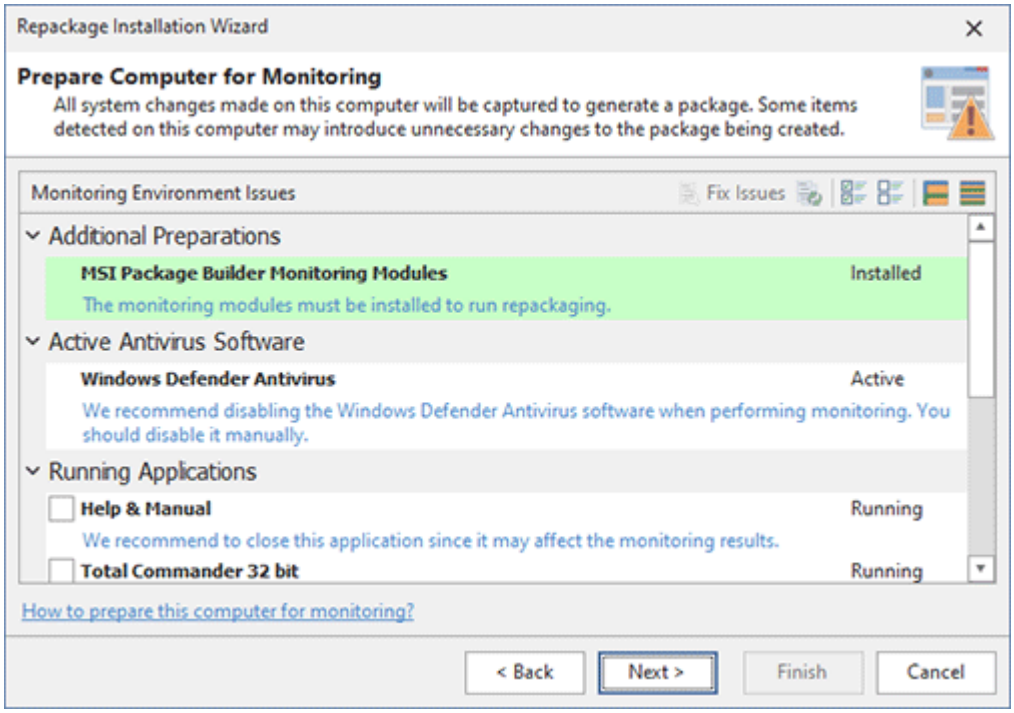
Copyright © 2001-2023 EMCO. All rights reserved.

**Step 5. Check the repackaging best practices**

The repackaging results depend on the environment where such repacking is preformed, so the wizard shows you a brief summary of the repackaging best practices. You can find more details on this topic in the Overview of the Repackaging Best Practices chapter. Make sure you perform repackaging in a clean environment and follow other recommendations, then press the **Next** button.

**Step 6. Prepare the computer for monitoring**

The program automatically checks the computer for potential monitoring problems and displays the discovered issues Pic 4. The program checks the status of the Windows services, such as Windows Update and Windows Search, detects if the antivirus is disabled and determines the list of running applications. All those services and applications may introduce unnecessary changes to the package being created. If any issues are detected, you need to follow the displayed instructions. To resolve the environment issues automatically, select the displayed items and click the **Fix Issues** button. Once all the issues have been resolved, you can proceed to the next step.

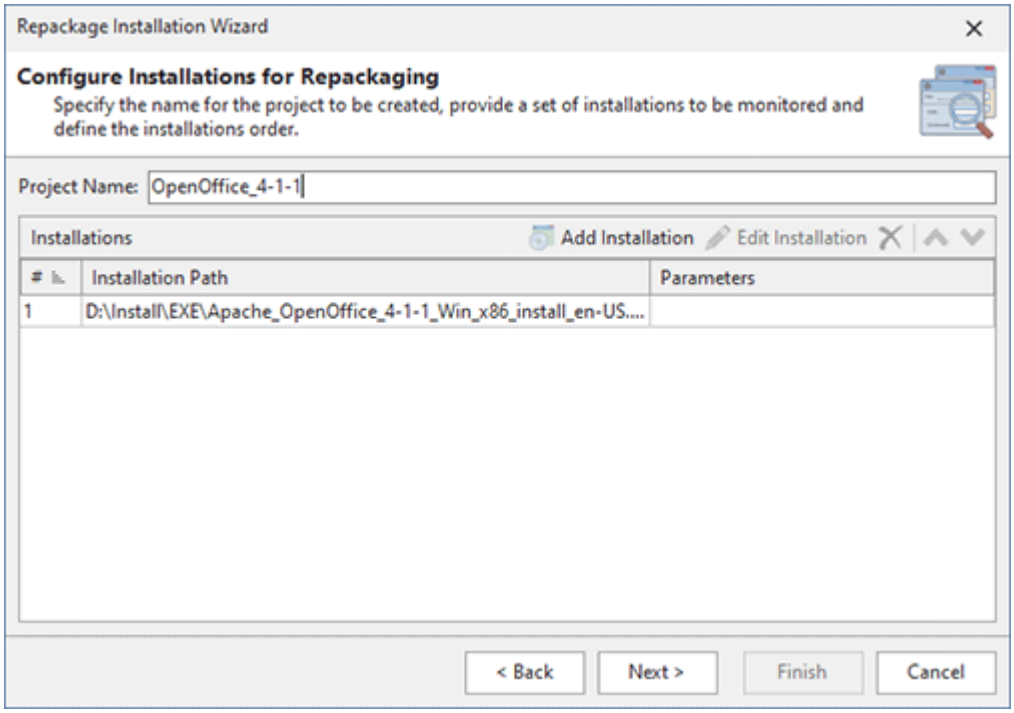



**Pic 4. Monitoring environment issues**

Copyright © 2001-2023 EMCO. All rights reserved.

### Step 7. Select the installation file

At this step, we need to specify the path to the original installation file to be monitored, so select the path to the downloaded *Open Office* installation. Once the path to the installation file is specified, you are prompted to enter the installation parameters. The parameters are optional and should be specified only if you need to run the monitored installation in a special way. In our case, we can leave the parameters empty and press the **OK** button to confirm the installation configuration. As you can see, the wizard allows you to specify multiple installations to be monitored, so the installations will be executed one-by-one and will be repackaged into a single MSI package. We need to create an MSI for the *Open Office* installation only, so we can press the **Next** button Pic 5.

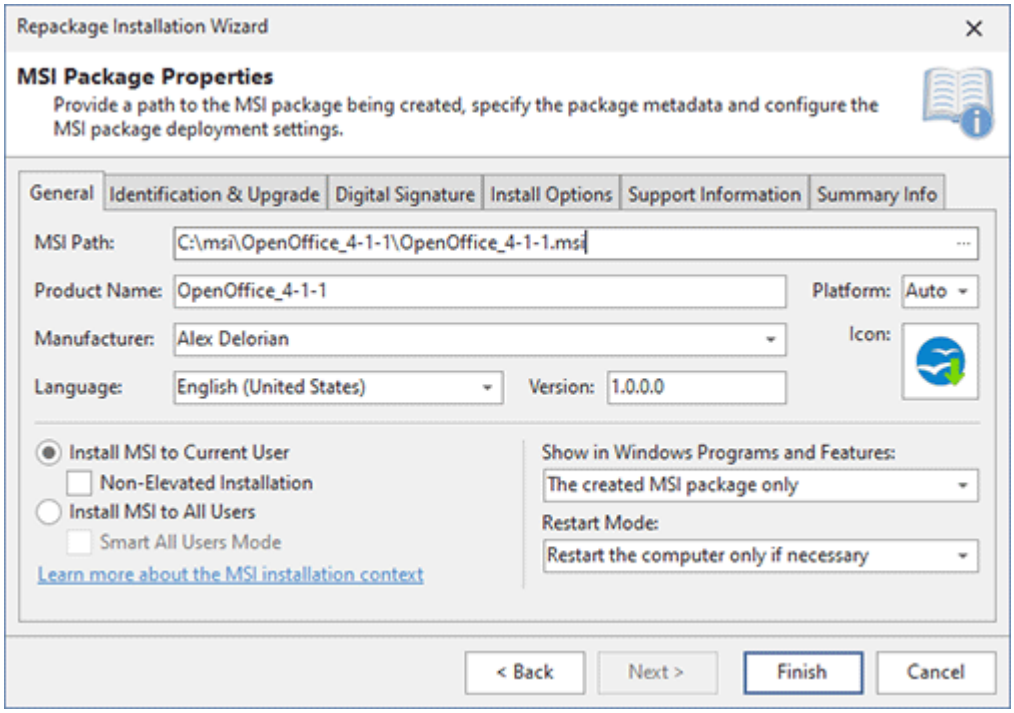



**Pic 5. Specify the installation to be repackaged**

Copyright © 2001-2023 EMCO. All rights reserved.

### Step 8. Specify the MSI properties

The wizard prompts you to select the output package type, so select MSI. Also, you need to specify the properties of the MSI package to be generated as a result of the repackaging, so you should enter the MSI path. In this dialog, you should also specify the required **Manufacturer** name, so you can enter your company name, for example. All other information, such as **Product Name**, **Version** and **Icon** is extracted automatically from the original installation, and you can edit it, if required Pic 6. You can also manage other options in this dialog, but for most of the installations, including this one, the default options are optimal, so you do not need to change them. You can learn more about the options in this dialog in the Creating an MSI Package chapter. Once you have specified the required settings in the dialog, you can press the **Next** button.




**Pic 6. Specify the MSI properties**

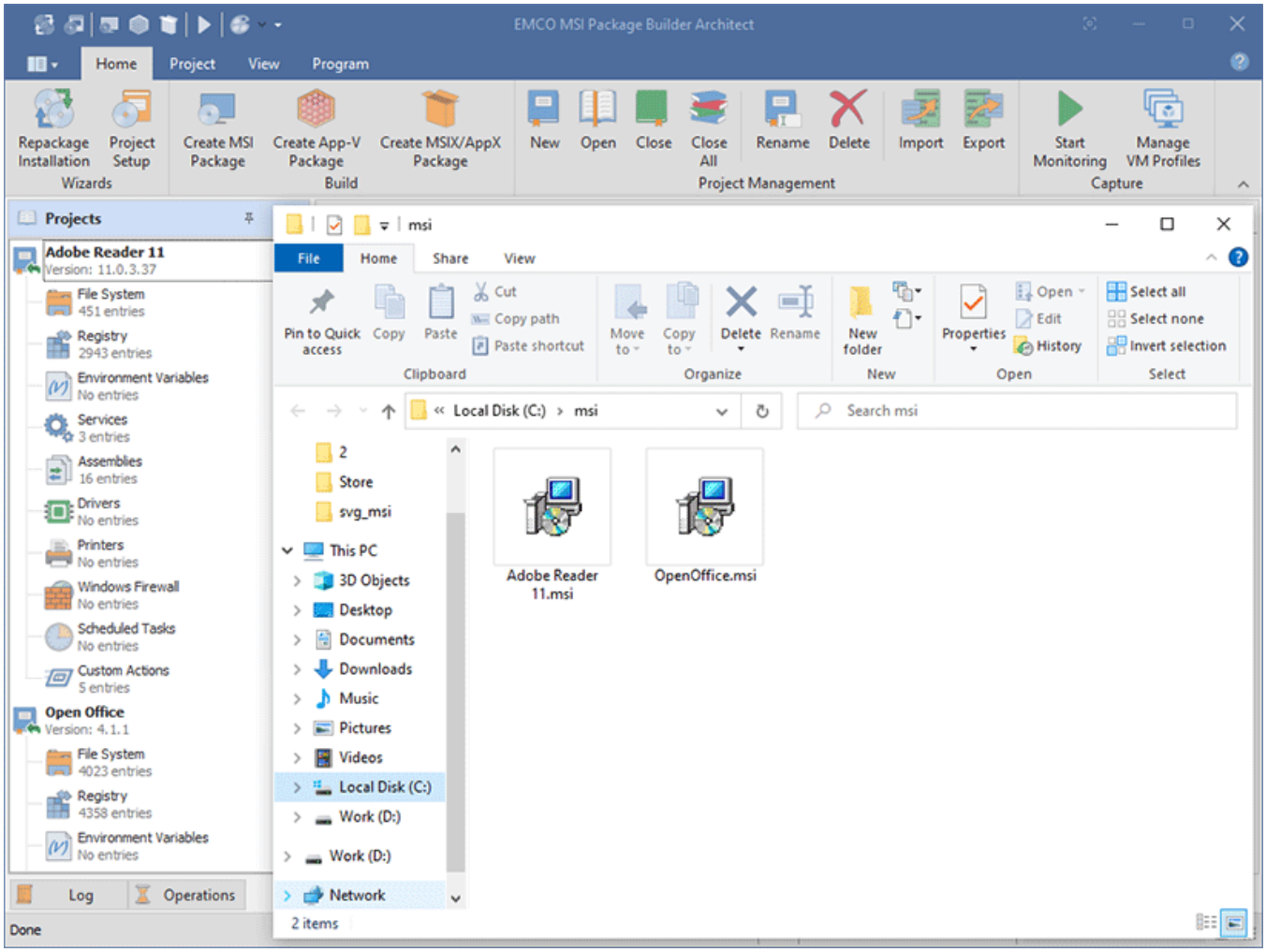
Copyright © 2001-2023 EMCO. All rights reserved.

### Step 9. Follow the installation steps

As you can see, the wizard has been closed and the specified original installation has been started. It means that monitoring is in progress now, and you need to follow the steps of the original installation. You can select the required installation options and, as a result, the original installation will create corresponding files and registry keys that are monitored by the program. Once the installation is completed, EMCO MSI Package Builder will automatically stop the monitoring and generate an MSI package. Note that if the installation prompts you to run the installed application, you should skip this step to let EMCO MSI Package Builder detect that the installation has been completed. If the application starts automatically after the installation, just close it to notify EMCO MSI Package Builder about the end of the installation process.

### Step 10. Get the generated MSI package

Once the installation is completed, you can return to EMCO MSI Package Builder that should automatically start generating an MSI package using the capturing results. Depending on the captured data size, the generation process may take some time. Once the generation is completed, you can see the generated MSI package in Windows Explorer.




**Pic 7. Getting the generated MSI**

Copyright © 2001-2023 EMCO. All rights reserved.

If you use an evaluation version of EMCO MSI Package Builder to generate an MSI, you get a trial MSI package with a few limitations. The generated package can be deployed during 30 days only and after deployment is displayed in Windows **Programs and Features** with the evaluation mark. These limitations do not influence the functionality of the repackaged application, so you can test it to make sure it works as expected. You can learn how to test MSI packages in the MSI Packages Testing and Troubleshooting chapter.

When generation of an MSI package is finished, EMCO MSI Package Builder automatically opens the MSI project, so you can review its content and customize it, if required. You can see the file system, registry and other resources captured during repackaging, modify them and add new resources to generate a new custom MSI package using the data stored in the MSI project. Besides, you can add custom pre- and post-install actions to be executed before and after the MSI deployment. You can learn how to use those features in the Installation Project Editing and Customizing chapter.

## Demo: Create Customized Installations Using Monitoring

In some cases, you may need not only to repackage a non-silent installation but also to customize it after installation to apply a license code or change the installed application settings and to get a deployment package that includes the installation and its customization.

As you may have learned in the Overview of the MSI Creation Methods chapter, you can repackage such installations using the **Capture System Changes** option of repackaging. It allows you not only to monitor an installation but also to monitor any changes performed in Windows and create a deployment package reproducing those changes. This very powerful feature of EMCO MSI Package Builder allows you to create any type of installation quickly and easily. Let's have a look at some examples of how you can use it.

## Create a Customized Firefox Installation

Let’s suppose we need to distribute *Firefox* across a local network. The standard *Firefox* installation doesn't work silently, so we need to repackage it into an MSI. An additional requirement here is that the *Firefox* installation should show our company home page at start and should have certain changed proxy settings. Those changes can be applied in *Firefox* options after installation, so we need to monitor the installation and application customization steps to create the required MSI.

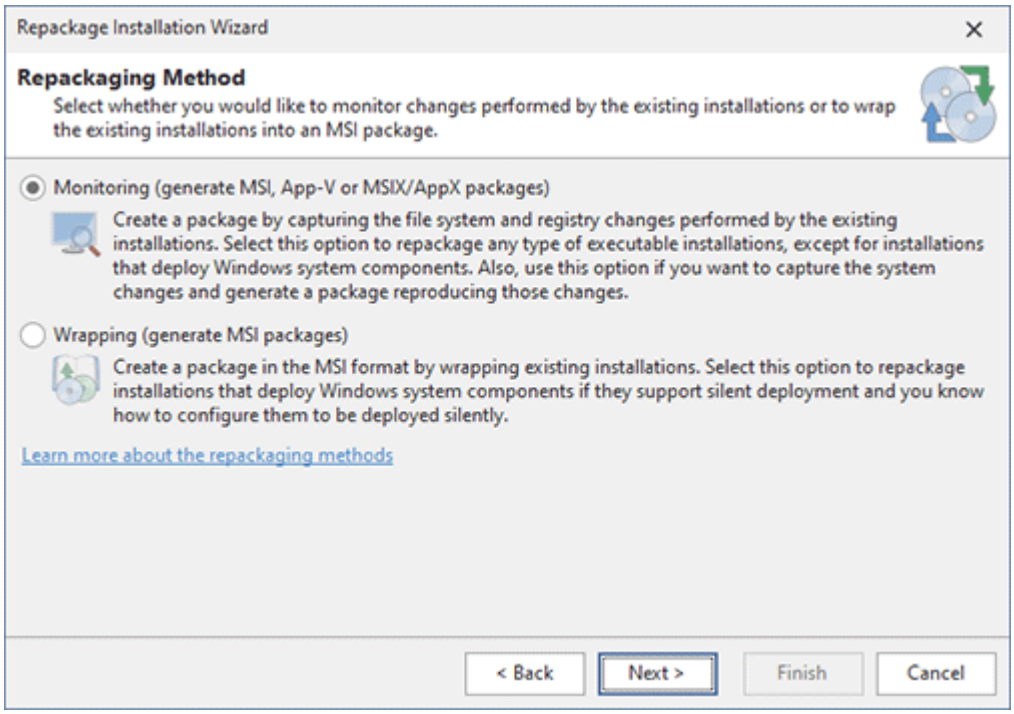
*Copyright © 2001-2023 EMCO. All rights reserved.*

**Step 1. Open the Repackage Installation wizard**

As the first step, you need to open the **Repackage Installation** wizard. If it isn't opened automatically, press the **Repackage Installation** button in the **Home** Ribbon page, read the welcome information and press the **Next** button.

**Step 2. Select the Monitoring option**

We are going to use installation monitoring, so select the corresponding option and press the **Next** button to proceed Pic 1.

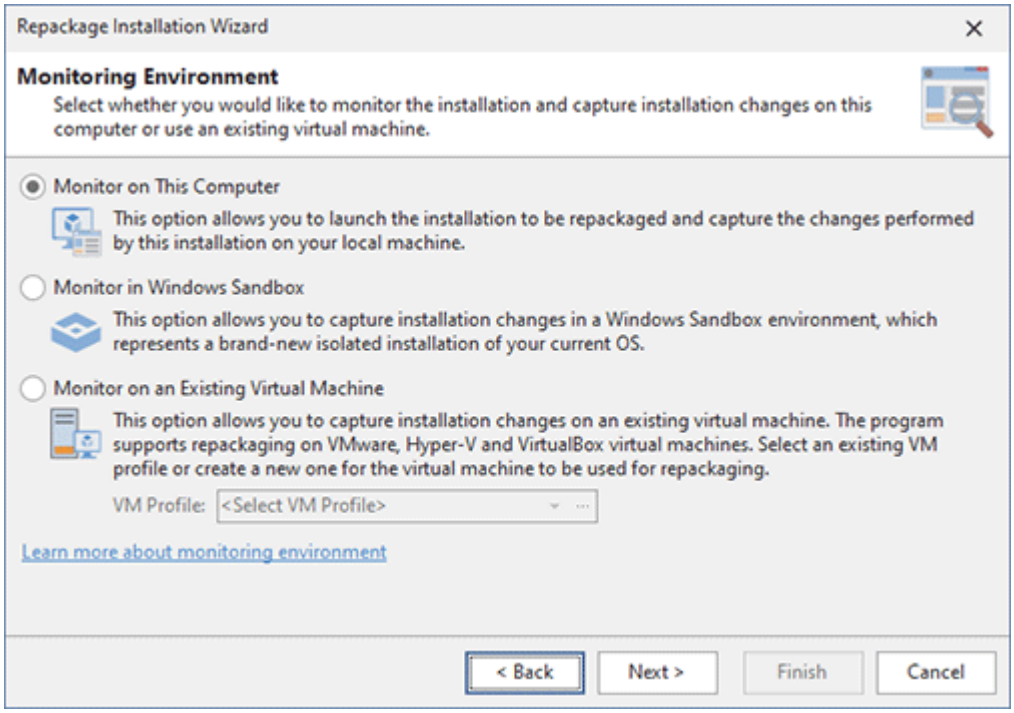



**Pic 1. Select the repackaging method**

Copyright © 2001-2023 EMCO. All rights reserved.

### Step 3. Select the Monitor on This Computer option

The program allows performing repackaging on the local computer, on Windows Sandbox or on an existing virtual machine. In this tutorial, we will repackage an installation on the local computer, so select the **Monitor on This Computer** option and proceed to the next step. To learn how to repackage an installation on Windows Sandbox or an existing virtual machine, refer to the Selecting monitoring environment chapter.

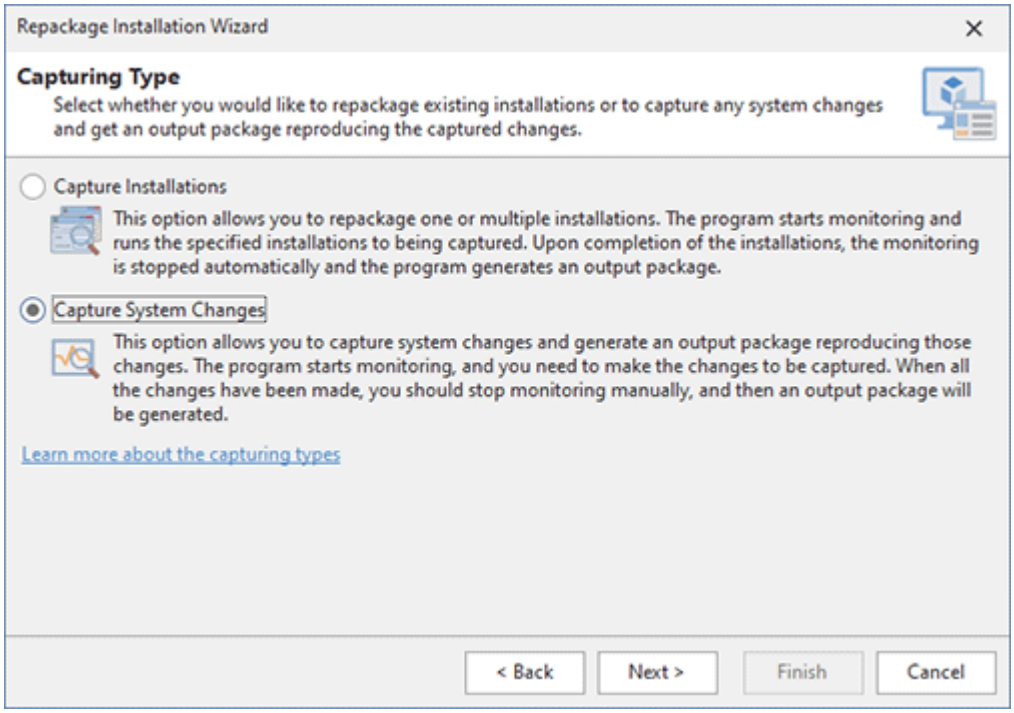



**Pic 2. Select monitoring environment**

Copyright © 2001-2023 EMCO. All rights reserved.

**Step 4. Select the Capture System Changes option**

We need to monitor not only the *Firefox* installation but also the post-install customization, so we cannot rely on automatic monitoring stopping, that is provided by EMCO MSI Package Builder for installations repackaging. As you learned in the Overview of MSI Creation Methods chapter, we need to use **Capture System Changes** to stop monitoring manually, so select the corresponding option and press the **Next** button Pic 3.




**Pic 3. Select the capturing type**

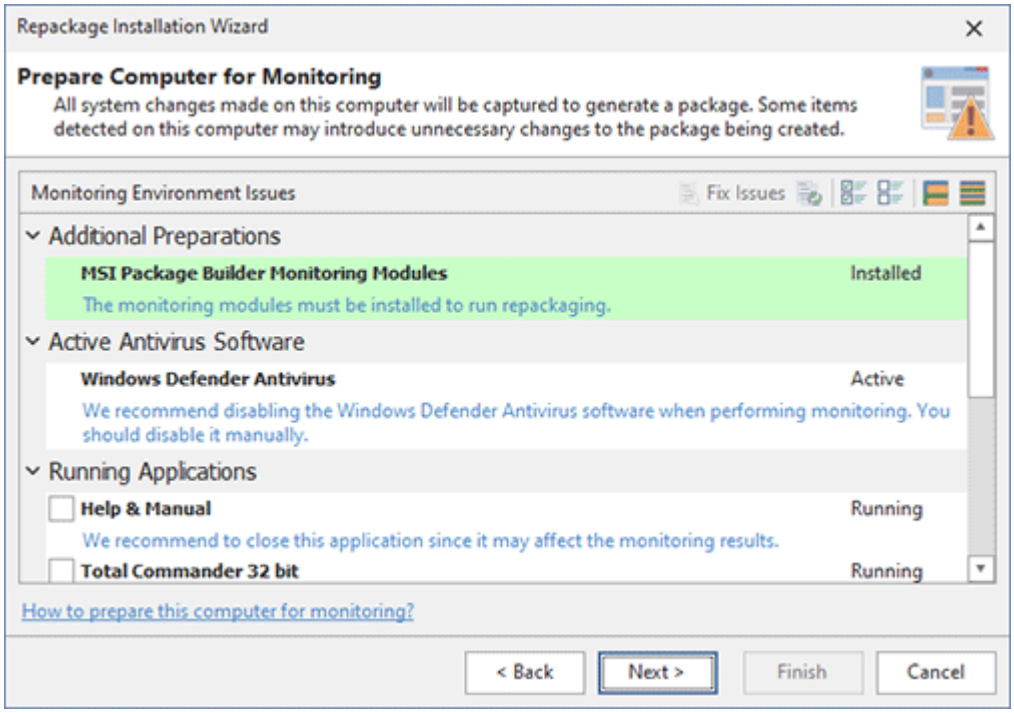
*Copyright © 2001-2023 EMCO. All rights reserved.*

**Step 5. Check the repackaging best practices**

When you use monitoring, you need to follow the repackaging best practices to get accurate repackaging results. Review the recommendations displayed by the wizard, make sure that you perform repackaging in a clean environment and follow other best practices, and then press the **Next** button. You can learn more about this topic in the Overview of Repackaging Best Practices chapter.

**Step 6. Prepare the computer for monitoring**

The program automatically checks the computer for potential monitoring problems and displays the discovered issues Pic 4. The program checks the status of the Windows services, such as Windows Update and Windows Search, detects if the antivirus is disabled and determines the list of running applications. All those services and applications may introduce unnecessary changes to the package being created. If any issues are detected, you need to follow the displayed instructions. To resolve the environment issues automatically, select the displayed items and click the **Fix Issues** button. Once all the issues have been resolved, you can proceed to the next step.




**Pic 4. Monitoring environment issues**

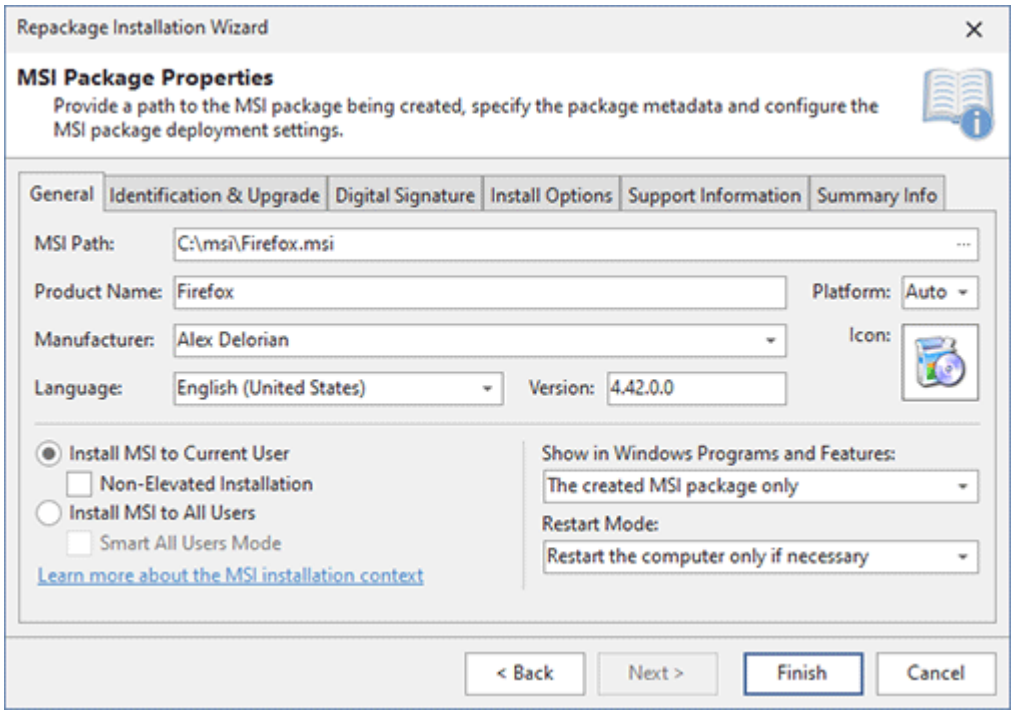
Copyright © 2001-2023 EMCO. All rights reserved.

**Step 7. Specify the project name**

EMCO MSI Package Builder will automatically create an installation project when the monitoring is completed, so you need to enter the project name and press the **Next** button.

**Step 8. Specify the MSI properties**

At this step, you need to select the output package type, so select MSI. Also, you need to specify the properties of the MSI package to be generated. You need to specify the path to the MSI file that will be generated automatically after the monitoring ends. You also need to specify **Product Name** and **Version** that should be the same as in the repackaged *Firefox* installation. You can enter your company name as **Manufacturer** and click on **Icon** to pick it from the *Firefox* installation file Pic 5. To learn more about the options in this dialog, refer to the Creating an MSI Package chapter. Press the **Finish** button when you finish editing the MSI properties.




**Pic 5. Specify the MSI properties**

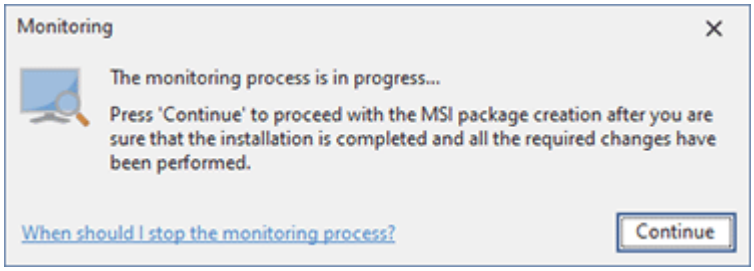
Copyright © 2001-2023 EMCO. All rights reserved.

**Step 9. Follow the installation and customization steps**

At this step, the monitoring is started, and any file system and registry changes are being captured. Note that you haven't specified the *Firefox* installation in the wizard, so it won't be started automatically, and you need to find the installation file in Windows Explorer and run it manually. You need to follow the *Firefox* installation process until *Firefox* is installed. The difference from installation repackaging lies in the fact that monitoring isn't stopped automatically when the installation is completed, so you can apply the required customization. After *Firefox* is deployed, you can run the *Firefox* application and specify your company website as the home page, then set the required proxy settings in the program options.

**Step 10. Stop the installation and get an MSI package**

Now that all the Firefox installation and customization steps are finished, you can go back to EMCO MSI Package Builder and press the **Continue** button to stop the monitoring Pic 6. As you can see, once the monitoring is stopped, EMCO MSI Package Builder generates an MSI package automatically, and you can see it in the Windows Explorer.




**Pic 6. Monitoring is in progress**

After getting an MSI package, you need to test it before you can deploy it across a network. The testing approach is explained in the Packages Testing and Troubleshooting chapter. If you generate an MSI using the trial version of EMCO MSI Package Builder, you get a trial MSI package that can be deployed during 30 days only and that displays a trial message in Windows **Programs and Features** after the deployment. These limitations do not influence the functionality of the repackaged application, so you can fully test it to make sure it works as expected.

As you can see, installation repackaging using the **Capture System Changes** option works automatically and is identical to the **Capture Installations** option except for the requirement to run the repackaged installation and stop the monitoring manually, but that allows you to repackage installations that require customization.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Create a Custom Font Installer

Using repackaging with the **Capture System Changes** option, you can easily get custom installations for any software deployment or Windows administration task. For example, let's assume that you need to distribute a font across your network, but you only have a font file. Of course, you can create an installation from scratch manually using the visual editors of EMCO MSI Package Builder, but in this case, you need to know a lot of technical details like the folder where the font should be installed, how to register the font, etc. Using **Capture System Changes**, you can create the required MSI automatically and let EMCO MSI Package Builder capture all the required changes while you install the font manually. Let's see how it works.

**Steps 1 - 8. Select the Capture System Changes option and configure the MSI properties**
Every time you use system changes capturing, the initial steps are the same: you need to open the **Repackage Installation** wizard, select the **Capture System Changes** option and make sure you follow the repackaging best practices. You also need to specify the MSI properties and to use the installation-specific properties such as the **MSI path**, **Product Name** and **Version**. You can see the details steps 1 to 8 in the example above. In the dialog where you need to provide the MSI properties, you should use the **Always restart computer after the installation** option for **Restart Mode**. It is required to make the installed font visible in the system after the deployment.

**Step 9. Install the font**
Now that EMCO MSI Package Builder monitors all the changes, you can select the font file to be installed in Windows Explorer and choose the option to install it from the context menu. Check the Windows **Control Panel > Fonts** to make sure the font has been installed correctly.

**Step 10. Stop the installation and get an MSI package**
Once the font installation is finished, you can go back to EMCO MSI Package Builder and press the **Continue** button to stop the installation and generate an MSI. As a result, the MSI generation is started automatically, and you get an MSI package that installs the font.

Using this approach, you can create an MSI automatically for any system administration task. For example, if you need to create an installation that deploys a set of files and registry keys. At Step 9, you can copy these files to the required target folders in Windows Explorer and create the required registry keys in Registry Editor manually, and EMCO MSI Package Builder will create an MSI that reproduces the captured changes.

## Demo: Repackage Silent Installations Using Wrapping

As you learned in the Overview of the MSI Creation Methods chapter, you can use the wrapping method to repackage specific installations that install Windows components and cannot be repackaged using monitoring because it isn't possible to reproduce these installations by capturing their changes.

When you wrap an installation, the original installation file is included into the wrapped MSI package and is executed during the MSI deployment. Since the generated MSI package should be deployed silently, the included installation should be configured to run silently as well, so you can use this method if you know how to deploy an installation silently.

Let's see how to use wrapping. For example, we need to repackage a *.NET Framework* installation that is a Windows component, so we cannot repackage it using monitoring.

Copyright © 2001-2023 EMCO. All rights reserved.

**Step 1. Determine and test the silent installation parameters**

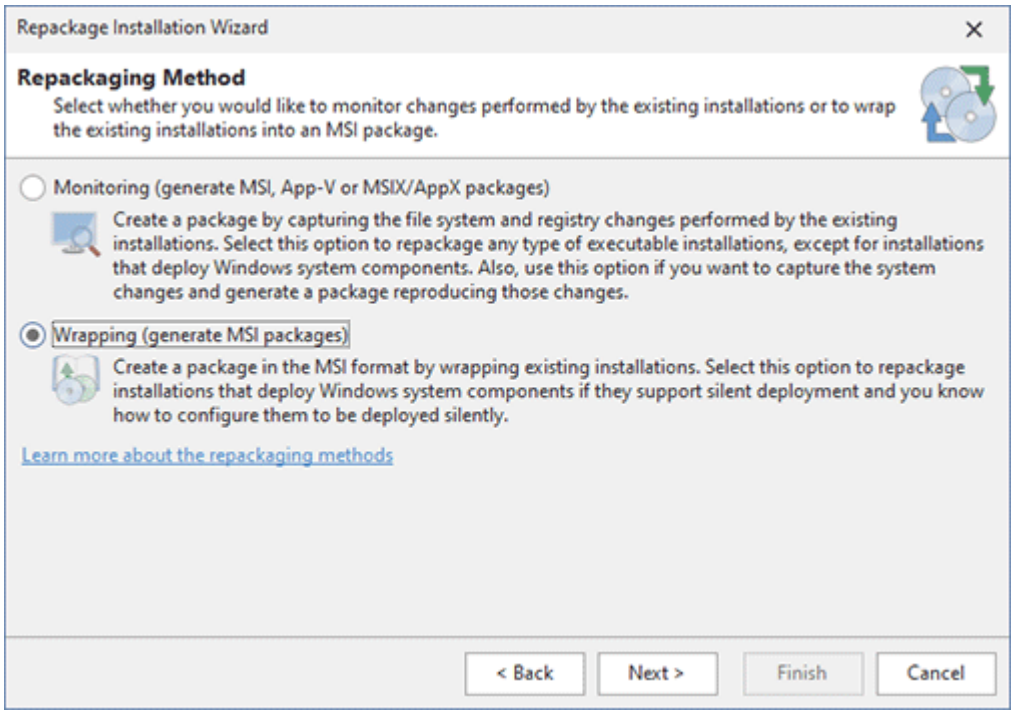
Since we need to create a wrapping MSI package ready for silent deployment, we should ensure that the included installation could be deployed silently. Using the Internet search, we can find out that *.NET Framework* installation can be deployed silently if the installation file is executed with specific parameters. Before generating an MSI package, we need to ensure that the silent deployment of the original installation works as expected. Use a virtual machine with a clean environment to test the silent installation of *.NET Framework*, which should be completed automatically while you should not be prompted to select any installation settings. If the tested installation can be deployed silently, you can proceed to the next step.

**Step 2. Open the Repackage Installation wizard**

Run EMCO MSI Package Builder. By default, you should see the opened **Repackage Installation** wizard at the start of the program. If it doesn't open automatically, you may click the **Repackage Installation** button on the **Home** Ribbon page. Once the wizard is launched, read the welcome information and press the **Next** button.

**Step 3. Select the Wrapping option**

We are going to use installations wrapping, so select the **Wrapping** option in the wizard and press the **Next** button to proceed to the next step Pic 1.

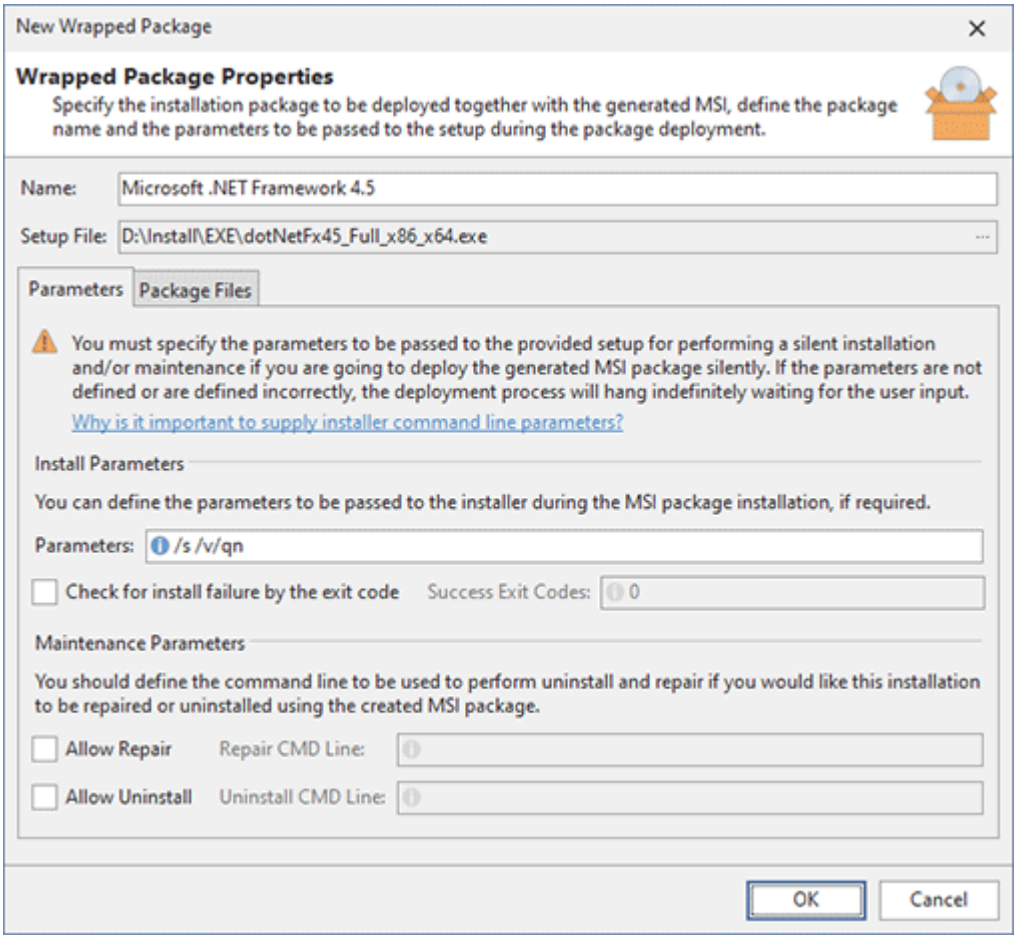



**Pic 1. Select the repackaging method**

Copyright © 2001-2023 EMCO. All rights reserved.

**Step 4. Specify the installation file and the installation options**

You are prompted to enter the path to the installation. In the file picker dialog, specify the path to the *.NET Framework* executable installation file. You should specify the installation options in the **New Wrapped Package** dialog. You can change the package name to *.NET Framework*, if you wish. In this dialog, you also need to specify the silent command-line parameters to be passed to the installation Pic 2. As we know, a *.NET Framework* installation can be deployed silently when executed with the */q /norestart* parameters, so enter them into the **Parameters** field and press **OK** to close the dialog. Finally, press the **Next** button in the wizard to proceed to the next step.

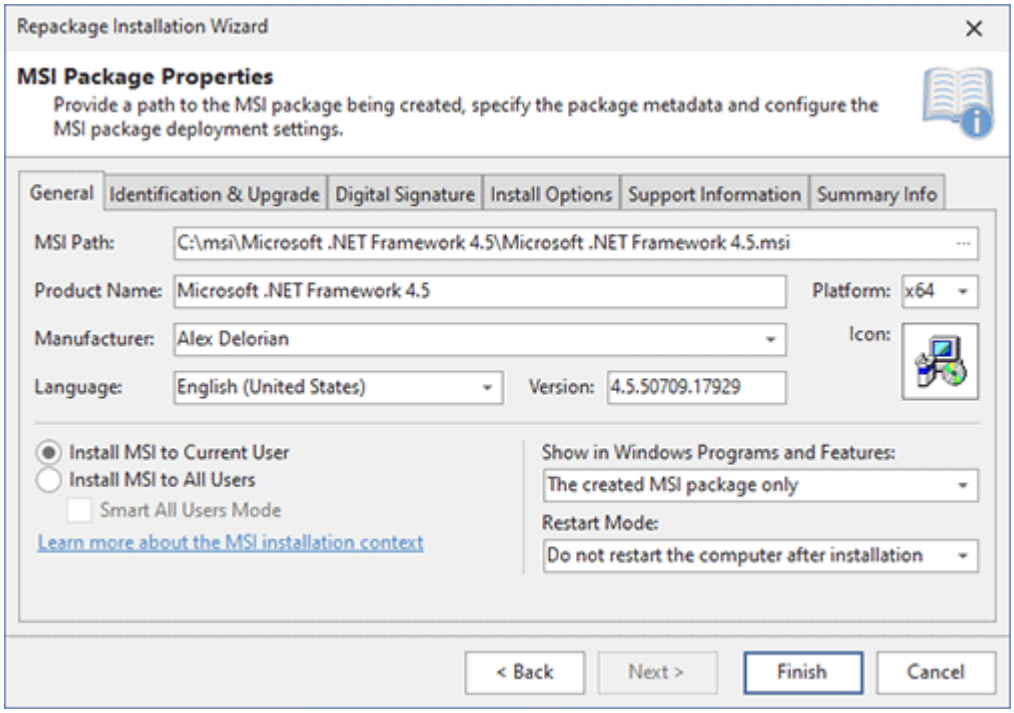



**Pic 2. Configure the .NET Framework installation options**

*Copyright © 2001-2023 EMCO. All rights reserved.*

### Step 5. Specify the MSI properties

You should enter the path to the MSI package to be generated. You also need to enter the **Manufacturer** name and change other options, if required Pic 3. Press the **Finish** button to start the MSI generation.




**Pic 3. Specify the MSI generation options**

### Step 6. Get the MSI package

EMCO MSI Package Builder generates an MSI package that wraps the specified silent installations and deploys them when you deploy the MSI. You can see the generated package in Windows Explorer that opens after an MSI is generated.

You need to test the generated MSI package before you can deploy it across a network to make sure it works as expected. Follow the testing recommendations provided in the MSI Packages Testing and Troubleshooting chapter.

If you use the trial version of EMCO MSI Package Builder, the generated MSI has trial limitations, i.e. it can be deployed during 30 days only, and its entry displayed in Windows **Programs and Features** includes the trial mark after the installation.

Using wrapping, you can also create an MSI package that deploys multiple installations. The wrapped package may include both EXE and MSI installations, the only requirement for wrapping being the ability to support silent deployment.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Installation Project Editing and Customizing

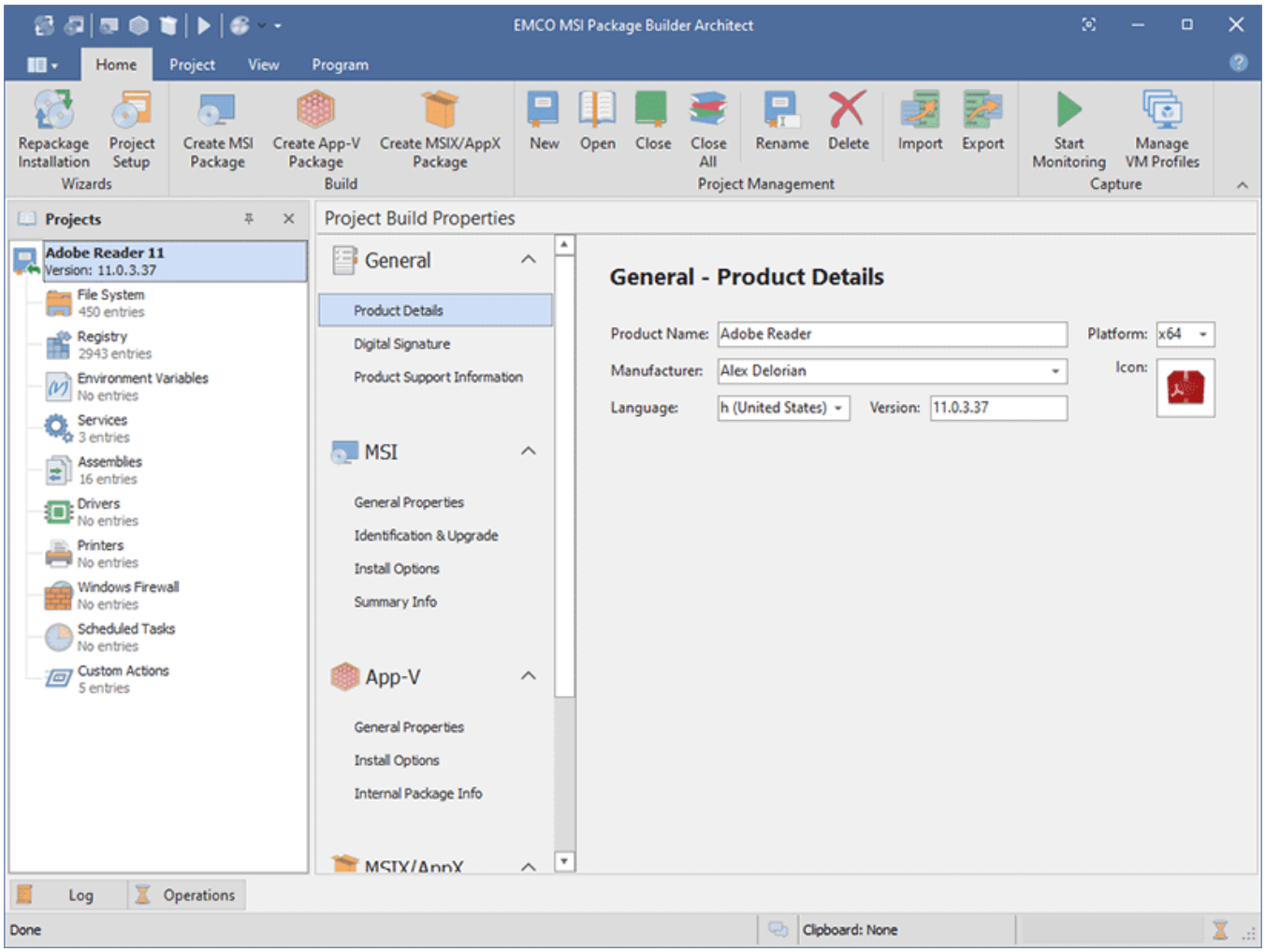
Regardless of the repackaging method you use to create a deployment package, you can see that the main screen of EMCO MSI Package Builder changes when the package generation is finished. EMCO MSI Package Builder automatically creates an installation project when you use installation monitoring and wrapping and opens such a project after the deployment package is generated, so you can review and edit it. The projects are created in the program's storage, and you can open and edit them anytime. You can apply custom changes through the visual editors and generate a modified deployment package.

In most cases, you don't need to review and edit installation projects because you get deployment packages at the end of the **Repackage Installation** wizard, but you can get access to the advanced features if you know how to edit and customize installation projects. Let's see how it works in practice.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Review and Modify Project Properties

When a deployment package generation is finished, you can see that the **Projects** view on the left of the main screen contains a number of new items, which are the recently created installation project and a set of the installation changes located under it. If you have any previously created deployment packages, you may open them to add to the **Projects** view. If you select the root node of a project in the **Projects** view, you can see the project properties on the right of the main screen. Those are the properties you specified in the **Repackage Installation** wizard Pic 1.




**Pic 1. Editing MSI project properties**

If you made a mistake when creating a deployment package and specified, for example, a wrong **Product Name** or **Version**, you can edit them in the project settings and press the **Create MSI Package**, **Create App-V Package** or **Create MSIX/AppX Package** button on the Ribbon to generate a new package with the modified properties.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Configure an MSI to Update Another MSI

You can configure an MSI package to be installed as an update for an older version of the same product installed as another MSI package. Windows Installer supports updates automatically; so if it detects that you install an update, it uninstalls the old MSI package and then installs the new one.

To configure an MSI to be deployed as an update, you only need to modify the MSI package properties. First, you need to modify the properties displayed in the **Identification and Upgrade** section. Enable the **Allow an installation upgrade** option and specify the same **Upgrade GUID** as the one used in the MSI you wish to update. You also need to update the **Version** number to ensure that the new MSI has a higher version number than the old one. Learn more about the technical requirements for an upgrade in the How should I configure the repackaged installations to support an upgrade? chapter.

## Digitally Sign MSI Packages

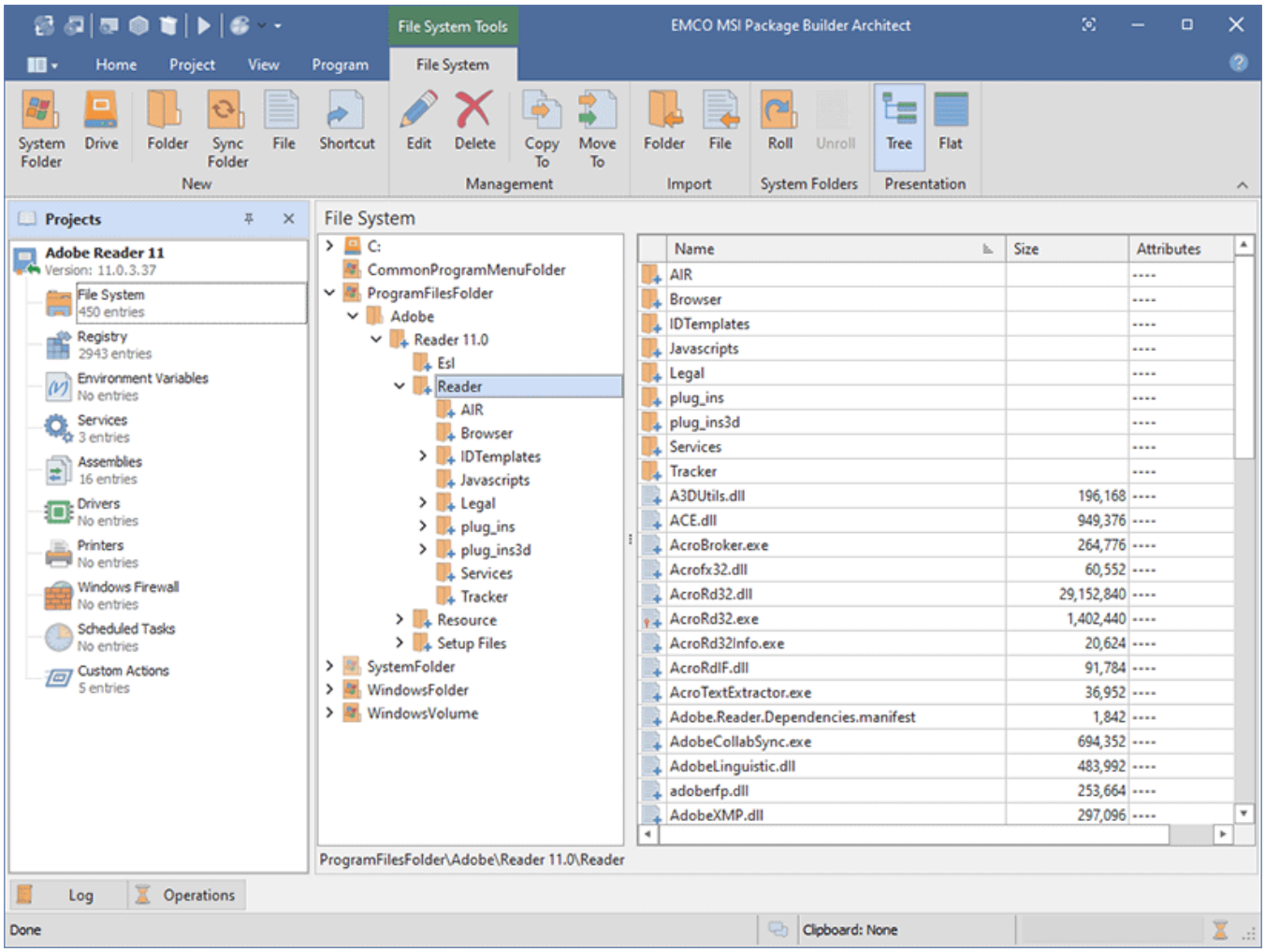
MSI packages apply Windows changes on deployment, so Windows Installer requires that they should be run under an account with administrative permissions The Windows UAC automatically checks the validity of digital signatures when software is executed with administrative permissions. By default, MSI packages generated by EMCO MSI Package Builder aren't digitally signed, so you can see the signature warning in the Windows UAC when you deploy such MSI packages manually. If you need to deploy the generated MSI packages remotely, note that such deployment tools as Microsoft SCCM don't deploy unsigned MSI packages.

EMCO MSI Package Builder allows signing MSI packages if you have a digital code-signing certificate and specify it in the program **Preferences** dialog. You can enable and disable digital signing of the generated MSI packages by configuring the corresponding options in the **Digital Signature** section of the MSI package properties. You can learn more about digital MSI singing in the Signing Packages chapter.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Review and Edit Changes

Under the project node in the **Projects** view, you can find the list of changes applied by the installation. These changes are grouped by types and are displayed as separate nodes in a tree, so you can see **File System**, **Registry**, **Assembles**, etc. If you select a node, on the right you can see a specific editor for the corresponding type of resources and can use it to review and modify the changes Pic 2. You can learn how to use every available editor in the Installation Projects chapter, but here you can get a quick overview of the available features.

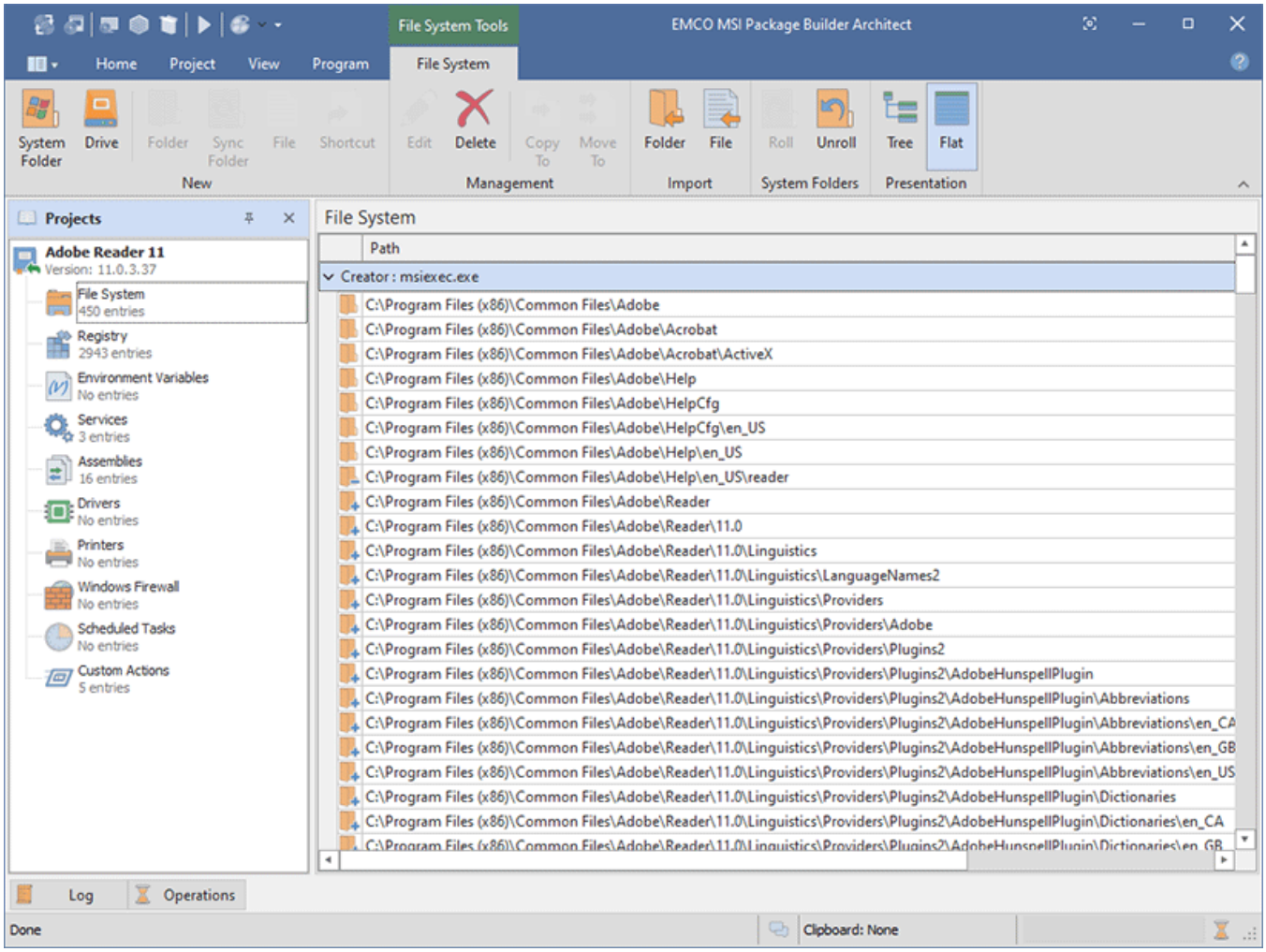



**Pic 2. Editing File System changes**

If you select the **File System** node, for example, you can see the structure of folders on the left and the modified files on the right. The resource icons shows the applied operation type, so you can see the files that were created, modified or deleted. If you created an installation project automatically using repackaging, you can notice that the file system paths are not absolute and include so-called System Folders. This is required to make installations portable from one PC to another, because different PCs may have different paths to standard Windows folders that store applications, their data and so on. EMCO MSI Package Builder can convert an absolute path to system folders automatically, so it's recommended that this conversion be performed if you edit a project manually. You can learn more about System Folders in the System Folder Definition Placeholders chapter.

Copyright © 2001-2023 EMCO. All rights reserved.

If you wish, you can switch the editor to the flat representation mode by clicking the **Flat** button on the contextual Ribbon page. In this case, the changes are displayed as a grid and this grid is grouped by default by the **Creator** values. This mode is helpful if the project was created using monitoring, because it allows seeing the captured processes and the files they changed Pic 3. If you need to troubleshoot a problematic package, you can use this mode to find and delete the changes performed by unrelated process monitored during the repackaging. The advanced package editing mode can only be used in special cases.




**Pic 3. File System changes grouped by processes that generated the changes**

You can modify the set of file system and other resources using the actions available in the editor's context menu and in the contextual Ribbon pages. If you need to review and modify other types of changes, you can select the required node in the **Projects** view and use the corresponding editors.

Copyright © 2001-2023 EMCO. All rights reserved.

## Define Custom Actions

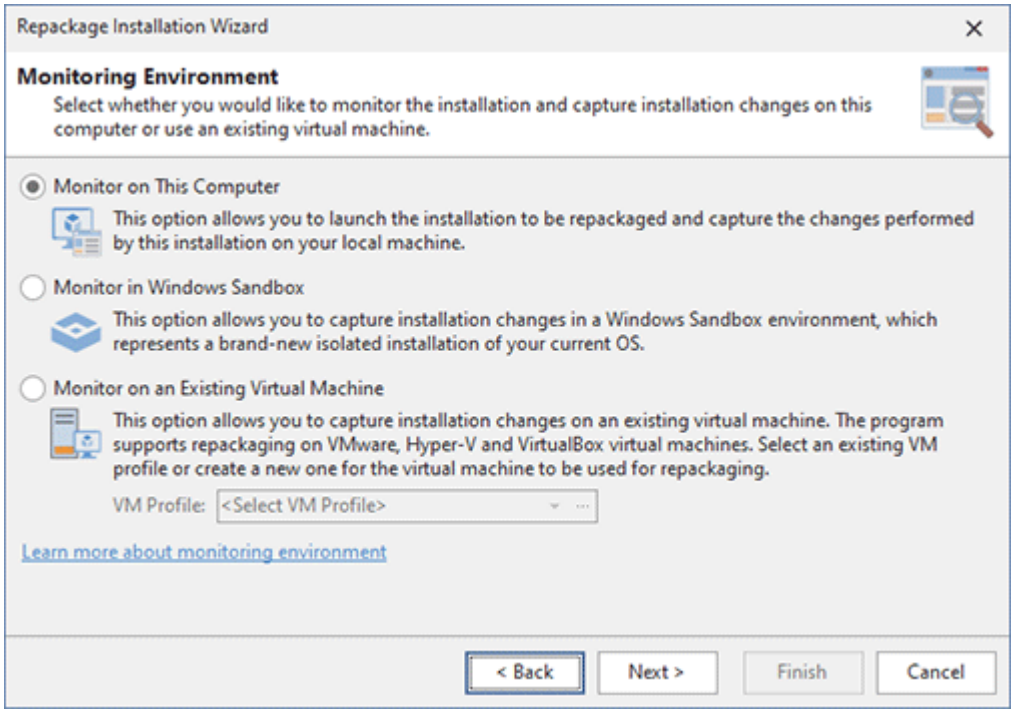
EMCO MSI Package Builder allows you to configure a deployment package to execute custom actions before and/or after the installation or uninstallation and to perform system configuration actions. Such actions can be managed through a special editor displayed after selecting **Custom Actions** in an installation project. You can configure a package to run any executable commands, scripts or applications as pre and post actions. The pre-install and post-uninstall executables should be available on the deployment PC so they cannot use the project files; but for post-install and pre-uninstall, you can run the executables deployed by the generated package. You can learn more about configuring custom actions in the Using Custom Actions chapter.

Using EMCO MSI Package Builder, you can customize installations according to your needs. For example, you can automatically create an installation project by monitoring the required actions and then review and modify the project contents by adding additional resources or a configuration of custom actions.

Copyright © 2001-2023 EMCO. All rights reserved.

## Selecting the Monitoring Environment

When you follow the **Repackage Installation** wizard, you need to select where you would like to monitor an installation and capture the installation changes: on the local computer, on Windows Sandbox, or on an existing virtual machine Pic 1.




**Pic 1. Selecting the monitoring environment**

Monitoring on a virtual machine is a preferred option, so if you have virtual machines hosted on a Hyper-V, VMware or VirtualBox server, you can select the option of monitoring an installation on an existing virtual machine, which should be specified in the wizard. If a VM isn't available, you can monitor on Windows Sandbox or on the local computer.

## Monitoring on the Local Computer

This is the default option that allows monitoring an installation on the same computer where EMCO MSI Package Builder is running. To use it, ensure that the local computer is selected as your monitoring environment and proceed to monitoring. Before starting monitoring, make sure you follow the repackaging best practices and use a clean environment as explained in the Overview of Repackaging Best Practices chapter.

For successful repackaging, you need to use a clean environment and restore the clean state every time you perform monitoring. When you monitor on the local computer, it is required that you perform manual actions before or after repackaging to restore the clean state of the monitoring environment. You can avoid this manual extra work if you use the **Monitor on Existing Virtual Machine** or **Monitor on Windows Sandbox** options.

Copyright © 2001-2023 EMCO. All rights reserved.

It's recommended to use existing virtual machines for repackaging, if possible. The advantages of this approach are explained in the corresponding section below.

## Monitoring on Windows Sandbox

Windows Sandbox is a feature available on the latest Windows versions starting from Windows 10 and above. Windows Sandbox allows you to run applications safely in isolated environment. Sandbox runs a clean, brand-new Windows installation that is sandboxed and runs separately from the host machine. All changes in sandbox are temporarily and are lost when sandbox is closed. These features of Sandbox meet requirements of a clean repackaging environment explained in the Overview of Repackaging Best Practices chapter, so you can use Windows Sandbox for monitoring.

Windows Sandbox doesn't require any setup or configuration. It is available out of the box on modern Windows versions. To use it you just need to enable virtualization features on your hardware and enable Sandbox it in Windows Features list. You can learn more about enabling Sandbox in the Repackaging on Windows Sandbox chapter.

Using Sandbox is easier than using a virtual machine for repackaging because you don't need to configure environment. Sandbox provides you with a clean environment each time when you run Sandbox. Note that Sandbox environment isn't equal to the regular OS environment, so some installations cannot be installed in Sandbox. If you need to repackage an installation that fails to deploy in sandbox, it's recommended to use a virtual machine to repackage this installation.

## Monitoring on an Existing Virtual Machine

This option allows you to connect to a Hyper-V, VMware or VirtualBox virtual machine remotely and use it for monitoring. In this case, virtual machines are used for monitoring only, but all other operations, such as creation of installation projects, projects management and packages generation, are performed on the local computer.

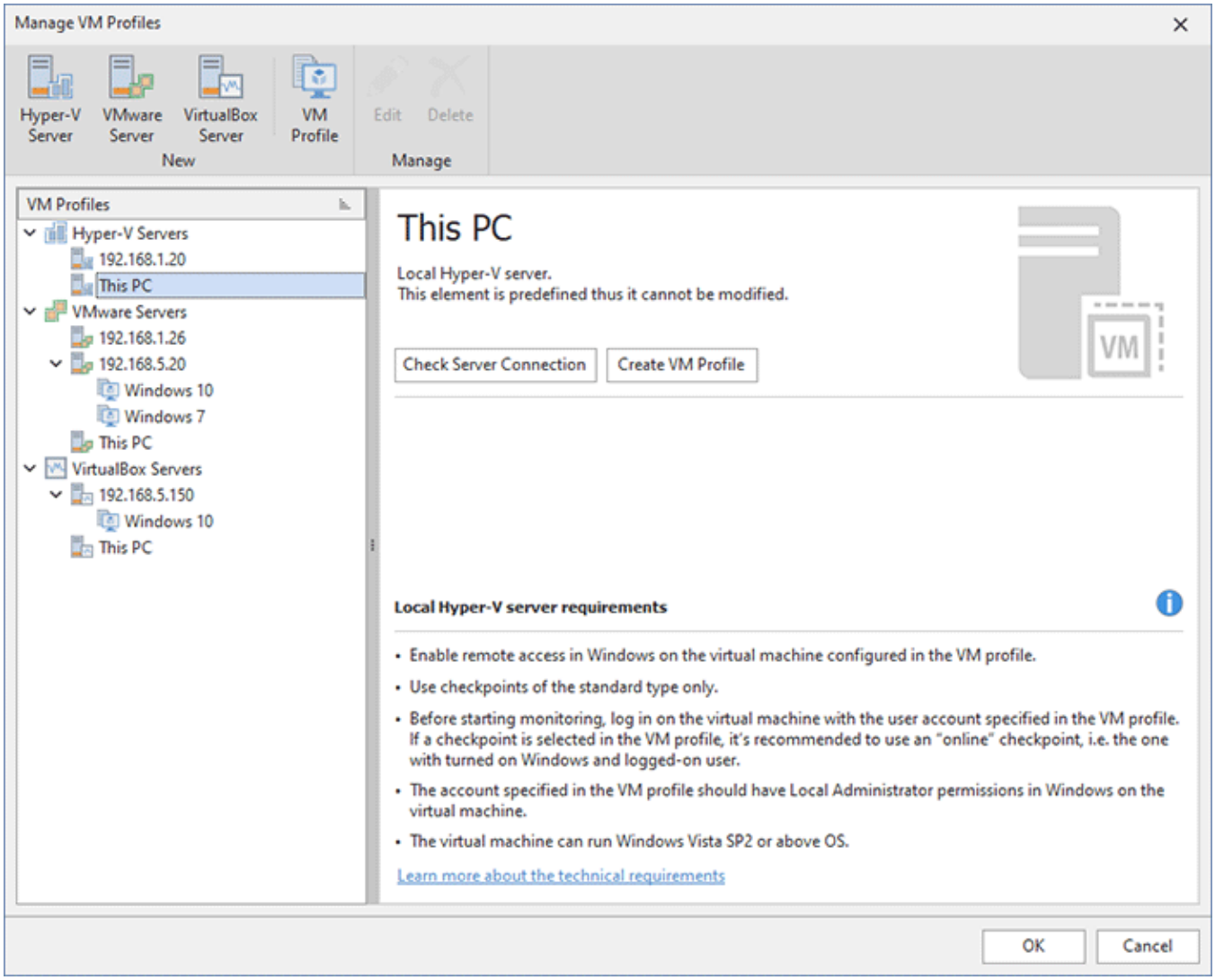
This approach is preferable to the traditional repackaging on the local computer for the following reasons.

- You can switch among repackaging environments quickly and easily. You can install a single copy of the program that can connect to different VMs hosted on Hyper-V, VMware and VirtualBox servers to use them for monitoring. It allows you to have one local copy of EMCO MSI Package Builder, and you don't need to install the program on every VM used for monitoring.
- Installation projects are stored on the computer where EMCO MSI Package Builder is installed. If you revert a VM, your projects stay intact since they are stored on another machine. You don't need to care about manual copying of projects from the VMs because the program automatically copies projects into a central projects storage at the end of every repackaging.
- You can configure the program to restore a VM automatically to a snapshot with a clean state at the end of repackaging. You don't need to restore VMs manually. If a VM has multiple snapshots, you can configure the program to use a specific snapshot for repackaging.

Copyright © 2001-2023 EMCO. All rights reserved.

These features help to automate the repackaging process, so it is recommended to use monitoring on an existing virtual machine if possible.

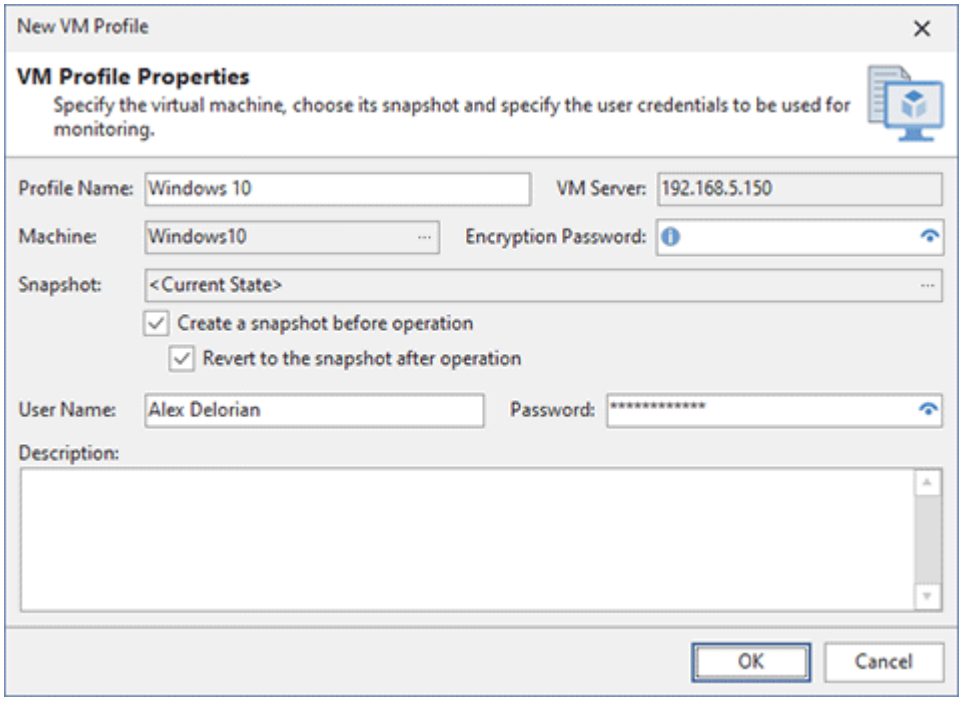
When the **Monitor on an Existing Virtual Machine** option is selected, you need to specify a VM to be used for monitoring. All VM configurations are stored as VM profiles, so you can select the required profile. If you don't have profiles, you can click the ... and open the **VM Profiles Manager** to configure a new VM Pic 2.




**Pic 2. VM profiles configuration**

*Copyright © 2001-2023 EMCO. All rights reserved.*

The program is able to work with Hyper-V, VMware and VirtualBox virtual machines hosted on the local PC or on remote servers. To create a VM profile, you need to configure where the VM is hosted. There are available default server configurations working on the local PC. If you need to work with a remote server, you can configure it by specifying its host name, username and password. Then you can create a VM profile by specifying the machine name, username, password and the optional snapshot of the virtual machine Pic 3.




**Pic 3. VM profile configuration**

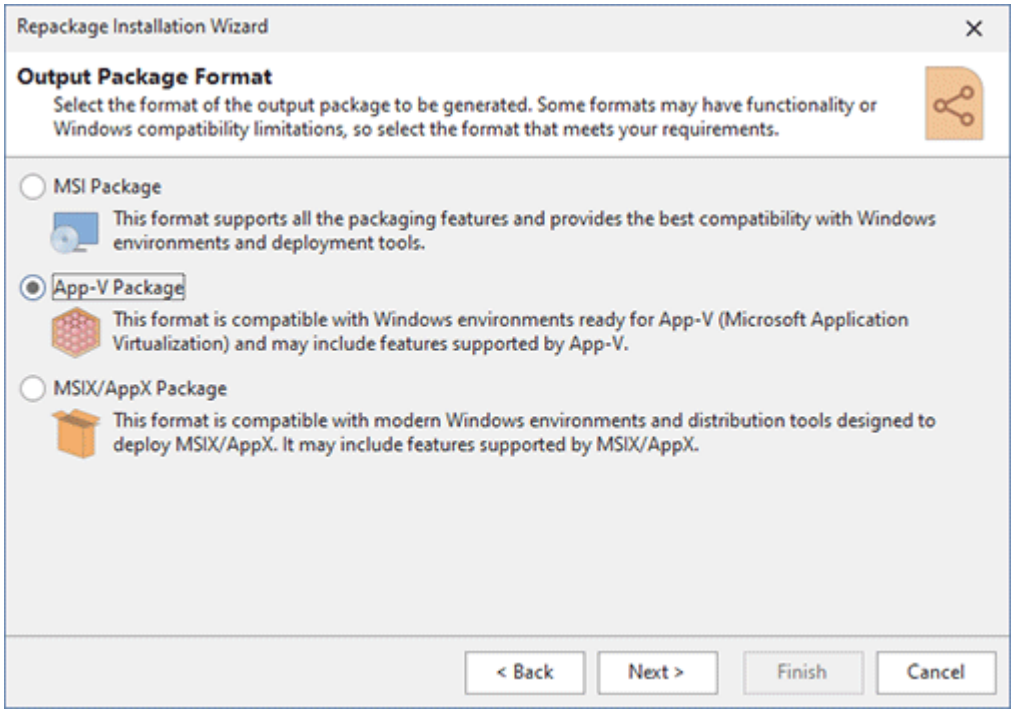
After creating a VM profile, you can select it in the **Repackage Installation** wizard, so the program will use the specified VM for monitoring. You can learn more about VM configuring and other topics in the Repackaging on a Virtual Machine chapter. There you can find important information on the technical requirements for virtual machines and servers that can be used by the program.

Copyright © 2001-2023 EMCO. All rights reserved.

## How to Create an App-V Package

The Architect edition of the program allows generating packages in multiple formats, including App-V. The App-V format is used by Microsoft Application Virtualization technology, so you can use the program to convert installations into virtual applications for Microsoft App-V.

There are multiple ways to create virtual applications in the program. If you have an installation to be converted into an App-V package, you can use installation monitoring in the same way as you use it to create an MSI package. In this case, you need to open the **Repackage Installation Wizard** and follow its steps. After selecting the **Monitoring** option in the wizard, you will be prompted to choose the output package type. To create an App-V package, you need to select the App-V option there Pic 1. The next steps are the same as those for repackaging of installations into MSI packages explained in the Demo: Repackage EXE to MSI Using Monitoring and Demo: Create Customized Installations Using Monitoring chapters. The program starts monitoring to track changes applied by the repackaged installation, and you get an App-V package instead of MSI in the end.




**Pic 1. Selecting the output format**

Copyright © 2001-2023 EMCO. All rights reserved.

If you have previously repackaged an installation into an MSI and have a saved project, you can open it in the program and click on **Create App-V Package** on the Ribbon to generate App-V from the project contents. You don't need to repeat repackaging in this case, because the project already includes the captured changes. It also works in the reverse direction: if you have created an App-V package before, you can just open the project and generate MSI or MSIX/AppX packages. An installation project stores all changes regardless of the output format, so you can generate MSI, App-V and MSIX/AppX packages using the same project.

You can also create an App-V package manually. In this case, you should create a new empty project and use **File System**, **Registry** and other editors in the program to add the required resources into the project. Having the project ready, you can press the corresponding button on the Ribbon to generate an App-V package.

When generating an App-V package, you get a set of files. Those files allow you to deliver an App-V application from App-V Server to App-V Clients and to deploy an App-V package locally in the standalone mode. The MSI file generated together with App-V allows you to install and uninstall the virtualized application locally in the standalone mode.

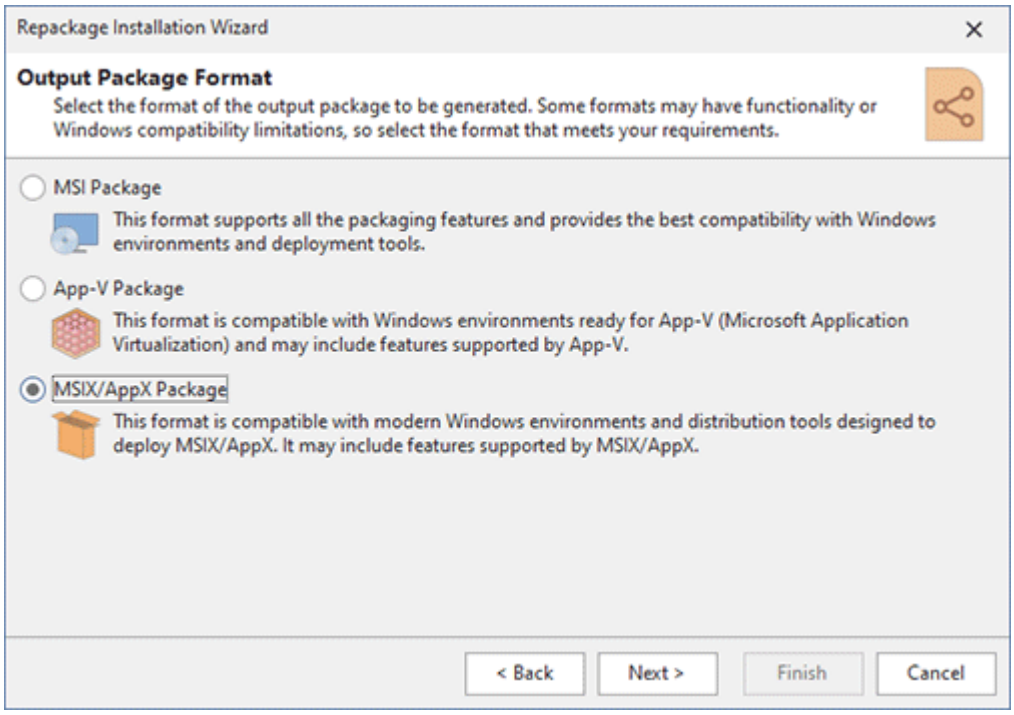
*Copyright © 2001-2023 EMCO. All rights reserved.*

## How to Create an MSIX Package

Using the Architect edition of the program, you can create MSIX/AppX packages and convert existing installations into MSIX/AppX. MSIX is a new installation package format designed for Windows 10. MSIX is a new and improved version of AppX that extends the features of AppX. AppX was originally designed for distribution of UWP apps, so MSIX extends it to provide support for Win32 applications. The generated MSIX/AppX packages can be distributed through Windows Store, installed by sideloading or distributed in local networks.

The contents and structure of an MSIX package are very similar to those of MSI, AppX and App-V packages, so technically it is possible to use the same installation project to generate packages in all these formats. It means that for creating MSIX/AppX packages you can use the same approaches you use to create regular MSI packages: you can repackage an existing EXE installation into MSIX/AppX, create installation contents manually using the editors, or, if you have an existing installation project, use it to generate MSIX/AppX.

Installation repackaging works as described in the Demo: Repackage EXE to MSI Using Monitoring and Demo: Create Customized Installations Using Monitoring chapters. You only need to select MSIX/AppX as the output format in the **Repackage Installation Wizard** Pic 1 and set MSIX/AppX options for the generated package, such as the target platform.




**Pic 1. Selecting the output format**

Copyright © 2001-2023 EMCO. All rights reserved.

It's worth noting that MSIX/AppX packages can only be generated on PC running Windows 10 and above. If you need to repackage installations into the MSIX/AppX format, you should run repackaging on Windows 10 (or above). Depending on the selected target platform, the program will automatically select the output package format. MSIX packages are generated only for target platforms supporting MSIX. For earlier versions of Windows 10 that don't support MSIX, the program generates AppX packages.

All MSIX/AppX packages must be digitally signed. So, if you have a digital certificate for code signing, you can configure it in the program's preferences. Alternatively, you can use an auto-generated self-signed certificate. Learn more about this topic in the Signing Packages chapter.

If you have previously repackaged an installation and there is an installation project available, or if you have created a project manually, you can open it in the program and generate an MSIX/AppX package by pressing the **Create MSIX/AppX Package** button on the Ribbon.

## Packages Testing and Troubleshooting

Once an MSI package is generated, it needs to be thoroughly tested before being deployed remotely across a network. To test an MSI, it's recommended that it be installed manually on a virtual machine with a fresh OS installation. Following the deployment, you need to run the installed application to make sure it works as expected. Sometimes, the changes applied by an MSI aren't visible for Windows after the MSI deployment. This may cause improper operation of the installed application. It's recommended that the PC be rebooted to let Windows apply the changes if you see that the installed application works incorrectly.

If you notice a problem with the generated MSI, you need to identify the reason of the problem, resolve it and repeat the repackaging process. Many problems can be resolved simply by using a clean environment for repackaging and following the best practices explained in the Overview of the Repackaging Best Practices chapter. Below, you can find a list of the most common MSI problems and recommendations on how to resolve them.

Copyright © 2001-2023 EMCO. All rights reserved.

**A file is locked during MSI install**

This problem usually happens when you don't use a clean environment for repackaging, and an unrelated process, such as an antivirus running in the background during the repackaging, generated a file change that is captured and included into the MSI. When you deploy the MSI, it tries to modify the file, but the file is in use by the running antivirus, therefore the MSI deployment cannot be finished. To resolve this problem, you need to repeat the repackaging using a clean environment as suggested in the repackaging best practices.

**A file is locked during MSI uninstall**

This problem is similar to the one described above. An MSI includes a change initiated by an unrelated application, which may be an antivirus, for example. When you deployed the MSI, the file change is accepted, but when you try to uninstall the MSI, the file is locked by the antivirus. To resolve this problem and uninstall the MSI, you need to close the program that locks the file and then uninstall the MSI. In any case, the generated MSI is incorrect and you need to create a new MSI using a clean environment for repackaging.

**An MSI was installed successfully, but the installation does not appear in Windows Programs and Features**

By default, EMCO MSI Package Builder creates MSI packages that are deployed per user, so if you didn't use any special options during the MSI deployment, the installation is visible for the current user only, and if you switch to another user, you won't be able find it in the Windows **Start menu** or in **Programs and Features**. To resolve this problem, you may provide special deployment options for the *msiexec* utility to be deployed to all user accounts during the MSI deployment. Alternatively, you may change the MSI project settings to use the **All Users** option instead of the **Current User** option and generate a new MSI package.

**An MSI upgrade doesn't remove the old version of the same application**

Windows Installer uninstalls the previous version of MSI automatically if a new version is being installed. Note that this only works for MSI packages, so you cannot rely on this feature if the previous version of the application was installed by an EXE installer. Besides, to use this feature, you need to properly configure the MSI version, Upgrade GUID and the installation scope as explained in the How should I configure the repackaged installations to support an upgrade? chapter.

**An MSI doesn't replace some files**

When you deploy an MSI package, Windows Installer checks the file properties for every file to be replaced. For files that contain version information in their properties, like DLL or EXE files, Windows Installer compares the versions of the new file and the existing one and replaces the existing file only if the new file version is greater than that of the existing one. This behavior is configured in the **Install Options** section of the MSI properties, so you can change it to replace all files unconditionally.

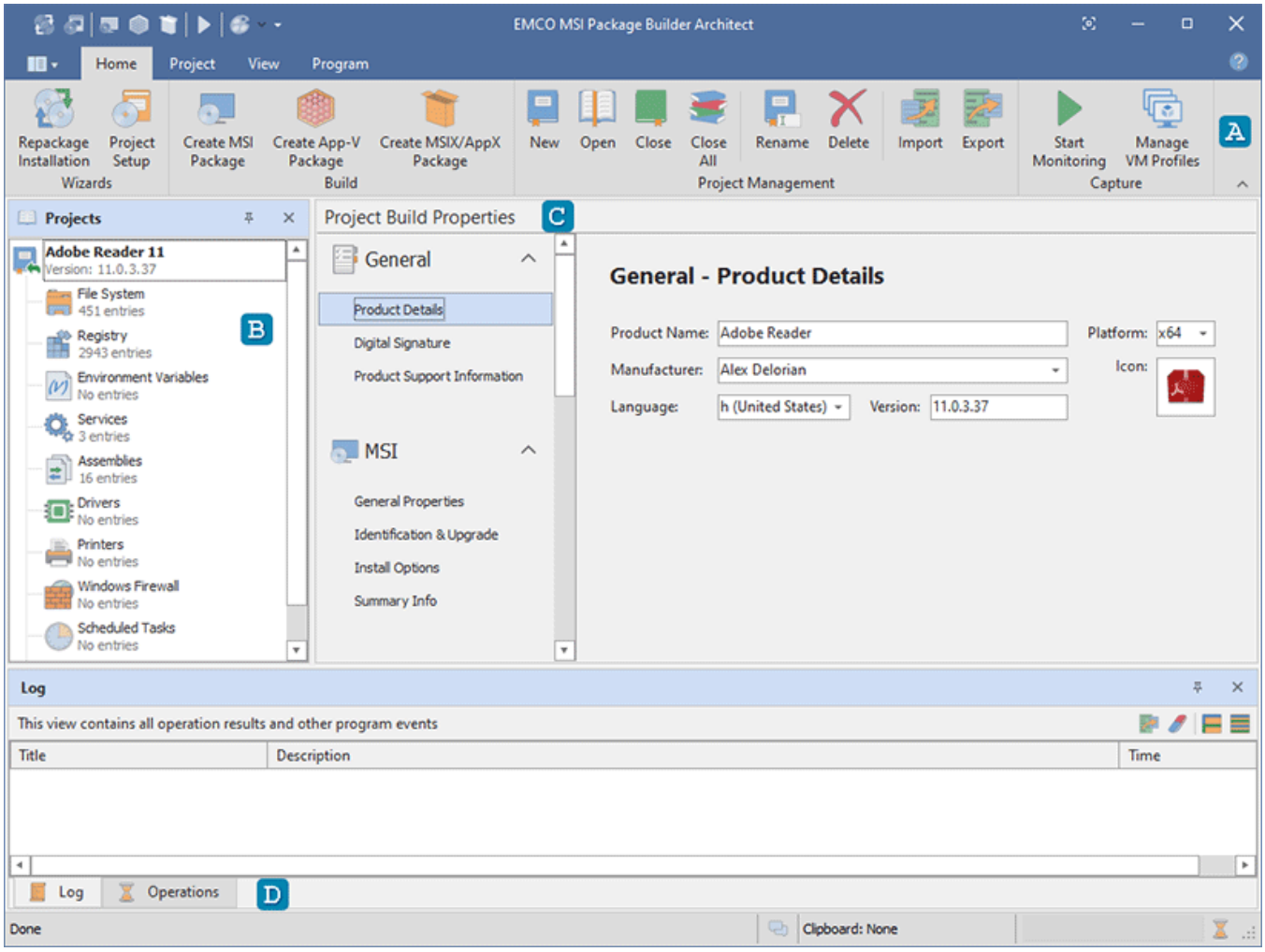
If you face a problem, first make sure that you follow all the repackaging best practices and repeat the repackaging in a clean environment. If you still have questions after that, you can *contact support* and provide information on the problem and the installation you try to repackage.

*Copyright © 2001-2023 EMCO. All rights reserved.*

# Chapter 3: Program Interface Overview

EMCO MSI Package Builder is a system program that needs to track file system and registry changes in the monitoring mode, so you need to have administrative permissions in Windows to run it. When you run the program on Windows Vista or a later OS, you need to allow the program to apply system changes in the User Account Control (UAC) prompt. If you use the evaluation version of the program, you need to press the **Evaluate** link to close the **Evaluation Wizard**, which appears at startup.

The program has a modern Ribbon-based interface Pic 1: the **Ribbon** is available at the top of the main screen A. It consists of four main tabs with items and can also display additional contextual tabs highlighted on the Ribbon when you use some of the available views and editors.




**Pic 1. The main program window**

*Copyright © 2001-2023 EMCO. All rights reserved.*

Under the Ribbon, on the left side of the screen, you can find the **Projects** view B. This view displays one or multiple installation projects that are currently opened in the program. For a project you can define the project settings, a set of file system and registry changes, changes to the environment variables configuration, the windows services deployment actions, the side-by-side assemblies configuration, drivers deployment actions, printers, Windows firewall configuration entries, custom actions and wrapping settings.

On the right side of the **Projects** view, you can see the main area, which shows different views depending on your current selection C. If there are no projects opened, you can see the **Welcome Screen** there with a set of available actions. If a project is selected in the **Projects** view, you can define the project build properties in the main area. To define a set of file system, registry or other changes to be performed by a deployment package, you should select the corresponding node under the project node in the **Projects** view. The content of the main area will be switched to the appropriate view. The available views for a regular installation project are the following: **File System**, **Registry**, **Environment Variables**, **Services**, **Assemblies**, **Drivers**, **Printers Windows Firewall**, **Scheduled Tasks** and **Custom Actions**. As for a wrapped packages installation project, there is only the **Wrapped Packages** view.

At the bottom of the main screen, you can find the **Log** and **Operations** views D. The **Log** view can be used to see all the events and error messages reported by the program. The **Operations** view can be used to manage the running operations.

To configure the program settings, you can open the **Preferences** dialog by selecting the corresponding option under the **Application** menu. The **Preferences** dialog allows you to configure the monitoring filters and other settings in the program.

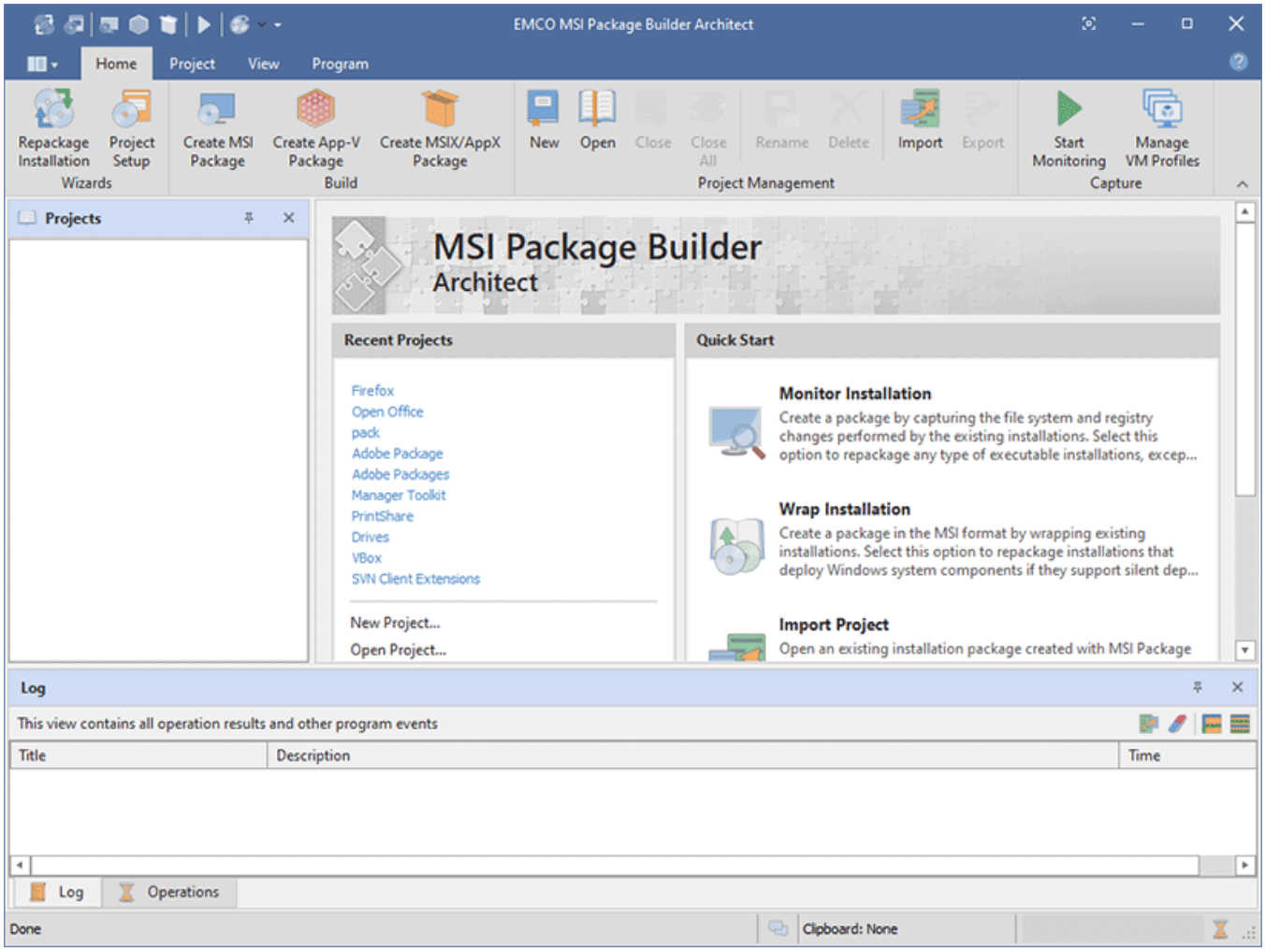
## What's Inside

Welcome Screen
Projects View
Project Details View
File System View
Registry View
Environment Variables View
Services View
Assemblies View
Drivers View
Printers View
Windows Firewall View
Scheduled Tasks View
Custom Actions View
Wrapped Packages View
Log View
Operations View
Graphical User Interface features

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Welcome Screen

The **Welcome Screen** is displayed just when you start the program. It is located right in the middle of the program main window and is intended to help you start working with MSI Package Builder Pic 1.



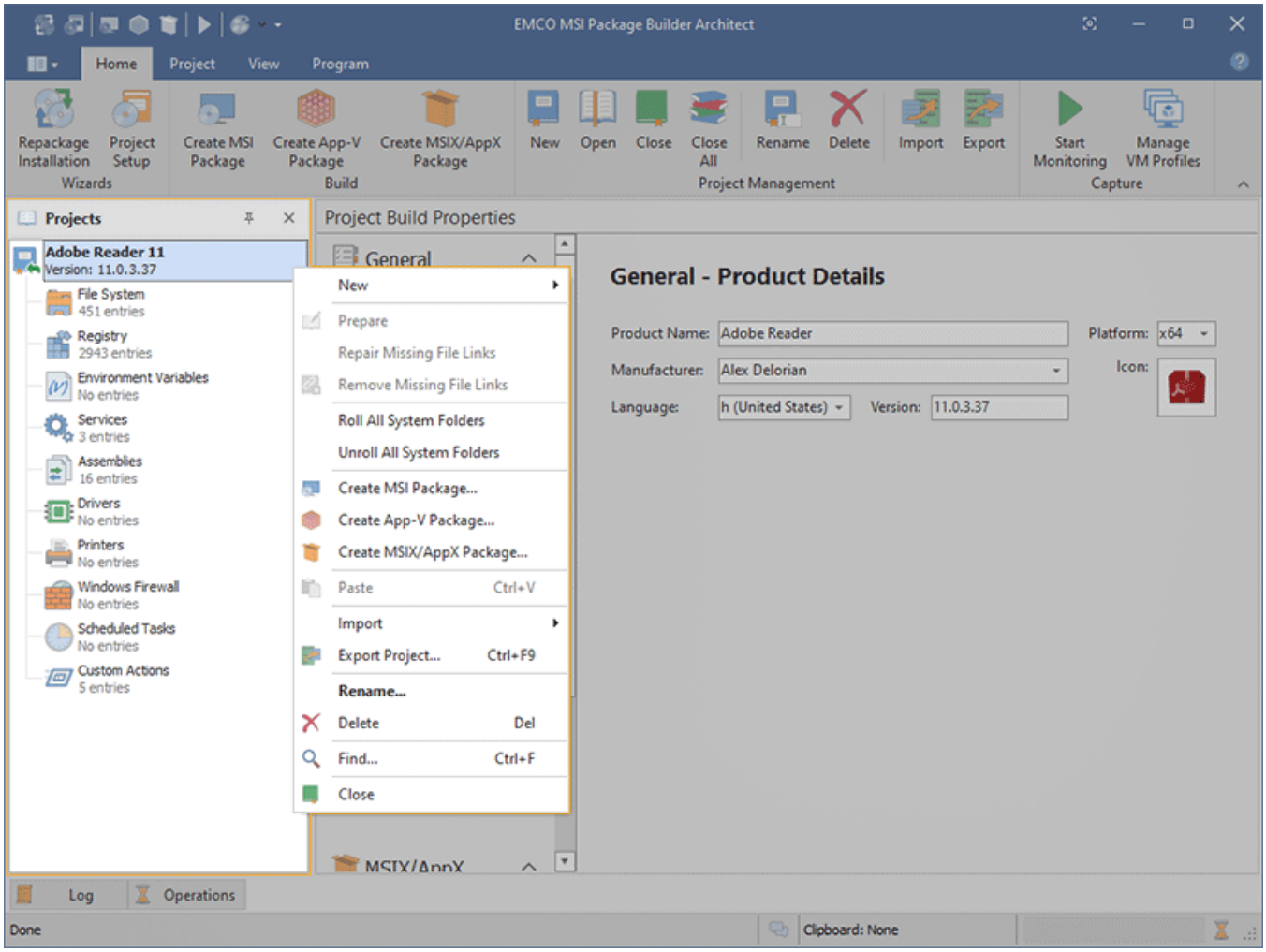

**Pic 1. The Welcome Screen**

The **Recent Projects** part of the **Welcome Screen** contains links to open recently used projects, open another project or create a new project. The **Get Help** part contains links to the information that is useful to anyone who is new to MSI Package Builder and would like to know more about the program. The main part of the **Welcome Screen** is the **Product Actions** one. It contains shortcuts to the actions that are used to perform installations repackaging.

Copyright © 2001-2023 EMCO. All rights reserved.

## Projects View

The **Projects** view is located by default on the left of the main program window. It is designed to display a set of currently opened MSI Package Builder projects and the nodes representing the changes to be performed by the deployment package Pic 1. On the project node, you can also see the current version of the deployment package to generate, and on each changes node, you can see how many entries are available in the corresponding set of changes.




**Pic 1. The Projects view**

The icon next to every item represented in the **Projects** view is used to describe the item type and state. The state icons are provided to help you understand what is currently happening in the program. You can always see if the item is being processed now by some operation or not, and if there are any processing problems. Below is the list of icons used:

- an MSI Package Builder project;
- an MSI Package Builder project is being operated;
- an MSI Package Builder project containing errors that should be resolved before creating a deployment package;
- an MSI Package Builder project based on monitoring results that is not prepared;
- an MSI Package Builder project based on monitoring results that is prepared;

Copyright © 2001-2023 EMCO. All rights reserved.

 - an MSI Package Builder project based on monitoring results that could not be fully prepared (not all of required files were available during the prepare operation).

The **Projects** view is the starting point for managing changes and creating a deployment package based on existing projects. It is also a master view for the main program area: the views displayed there depend on the selection in the **Projects** view.

## Functions Overview

| | |
|---|---|
| **Projects Management** | From the **Projects** view, you can create, rename, delete, open and close MSI Package Builder projects. To perform these actions, you can either use the pop-up menu of the **Projects** view or the buttons from the **Project Management** group on the **Home** Ribbon page. The actions for creating a new project, opening existing projects and closing all projects are available in the pop-up menu displayed on an empty selection, and to edit, rename, delete and close specific projects, you should first select those projects. Detailed information about the projects management abilities is available in the Projects Management section of this document. |
| **Projects Preparation** | Within the **Projects** view, it is possible to prepare the projects created on a basis of the monitoring results. The preparation steps consist of copying the file system resources to the projects storage, repairing missing links and rolling system folders. These actions are available in the pop-up menu and on the **Project** Ribbon page. To learn more about preparing the projects, refer to the Projects Preparation section of this document. |
| **Capturing Changes** | From the **Projects** view, it is possible to capture changes to the underlying operating system and create a project, containing the performed changes. To start a new session of changes capturing use either the **Start Monitoring** item from the pop-up menu displayed on an empty selection, or the **Start Monitoring** button from the **Capture** group on the **Home** Ribbon page. To stop the changes capturing session, use the **Stop Monitoring** items. The capturing abilities can be used for low-level repackaging and for simple monitoring of the application activity. |
| **Deployment Package Generation** | You can easily create a deployment package containing the changes represented by any project displayed in the **Projects** view. Just select that project and choose the **Create MSI Package**, the **Create App-V Package** or the **Create MSIX/AppX Package** item in the pop-up menu – it is equivalent to using the corresponding buttons from the **Builder** group on the **Home** Ribbon page. Detailed information about the deployment package generation abilities is available in the Creating an MSI Package, Creating an App-V Package and Creating an MSIX/AppX Package sections of this document. |
| **Deployment Package Import** | It is possible to create a project by extracting any deployment package generated with MSI Package Builder in the **Projects** view. This feature is called a deployment package import. To perform import, use either the **Import Package** item from the pop-up menu displayed on an empty selection, or the **Import Package** button from the **Project Management** group on the **Home** Ribbon page. |
| **Search** | Within the Projects view, you can execute a search for specific project using the Find item from the pop-up menu. |

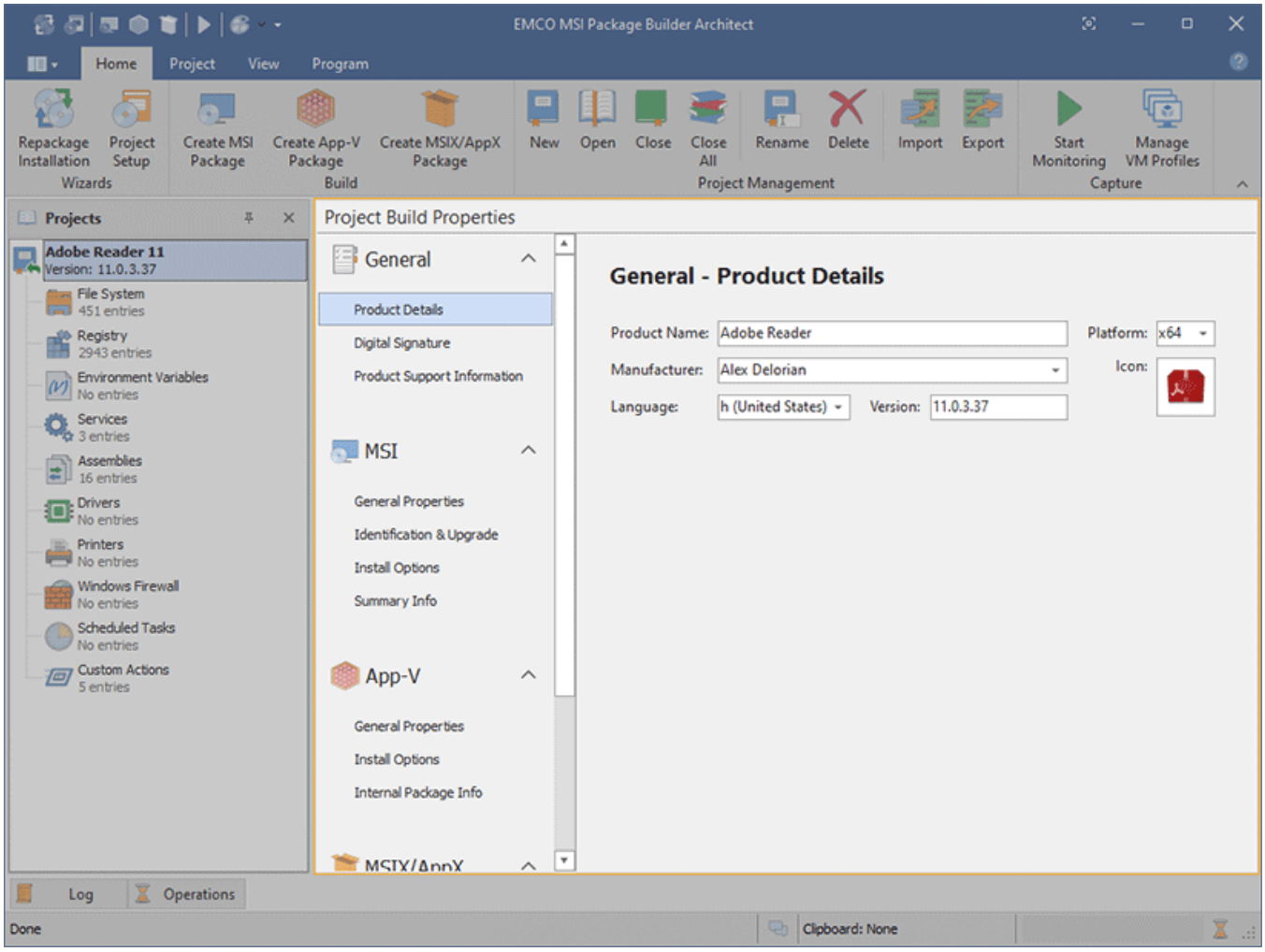
Copyright © 2001-2023 EMCO. All rights reserved.

From the **Projects** view pop-up menu, depending on the selection, it is possible to perform projects management, capture changes, import existing deployment packages, etc.

Copyright © 2001-2023 EMCO. All rights reserved.

## Project Details View

The **Project Details** view is displayed within the main program area when an MSI Package Builder project is selected in the **Projects** view. This view is used to define the data that is used to configure a deployment package being created on a basis of the project Pic 1. On the left side of the view you can find a navigator that allows you to switch between project configuration sections. These sections are grouped by functionality, so you can navigate through general settings and settings for MSI, MSIX and App-V packages.




**Pic 1. The Project Details view**

*Copyright © 2001-2023 EMCO. All rights reserved.*

The **Product Details** section of the **General** group allows you to define the generic package properties such as the product name, manufacturer, language, etc. Within the **Digital Signature** section, it is possible to override the common package signing configuration for the selected project, if it is required. You can also select there a set of project files that should be digitally signed during the package generation. The **Product Support Information** section can be used to override the common support configuration defined within the program preferences.

The **MSI** group is used to configure the Windows Installer package settings of the project. The **General Properties** section allows you to define the MSI installation context, the way the generated package behaves regarding the Programs and Features section of the Control Panel and if the PC should be restarted for the installation to complete. The **Identification & Upgrade** section is used to define the product GUID and upgrade GUID, if it is required. The **Install Options** section allows you to define the supported operating systems, the required Microsoft .NET Framework version and install mode. And the **Summary Info** section is responsible for defining summary information for the package.

This configuration is stored between sessions and is used to complete the corresponding properties during an MSI package creation. For a detailed description of each option, refer to the Creating an MSI Package section of this document.

The **App-V** and **MSIX/AppX** groups are available only for regular installation projects and are is used to configure the corresponding deployment packages output based on the project.

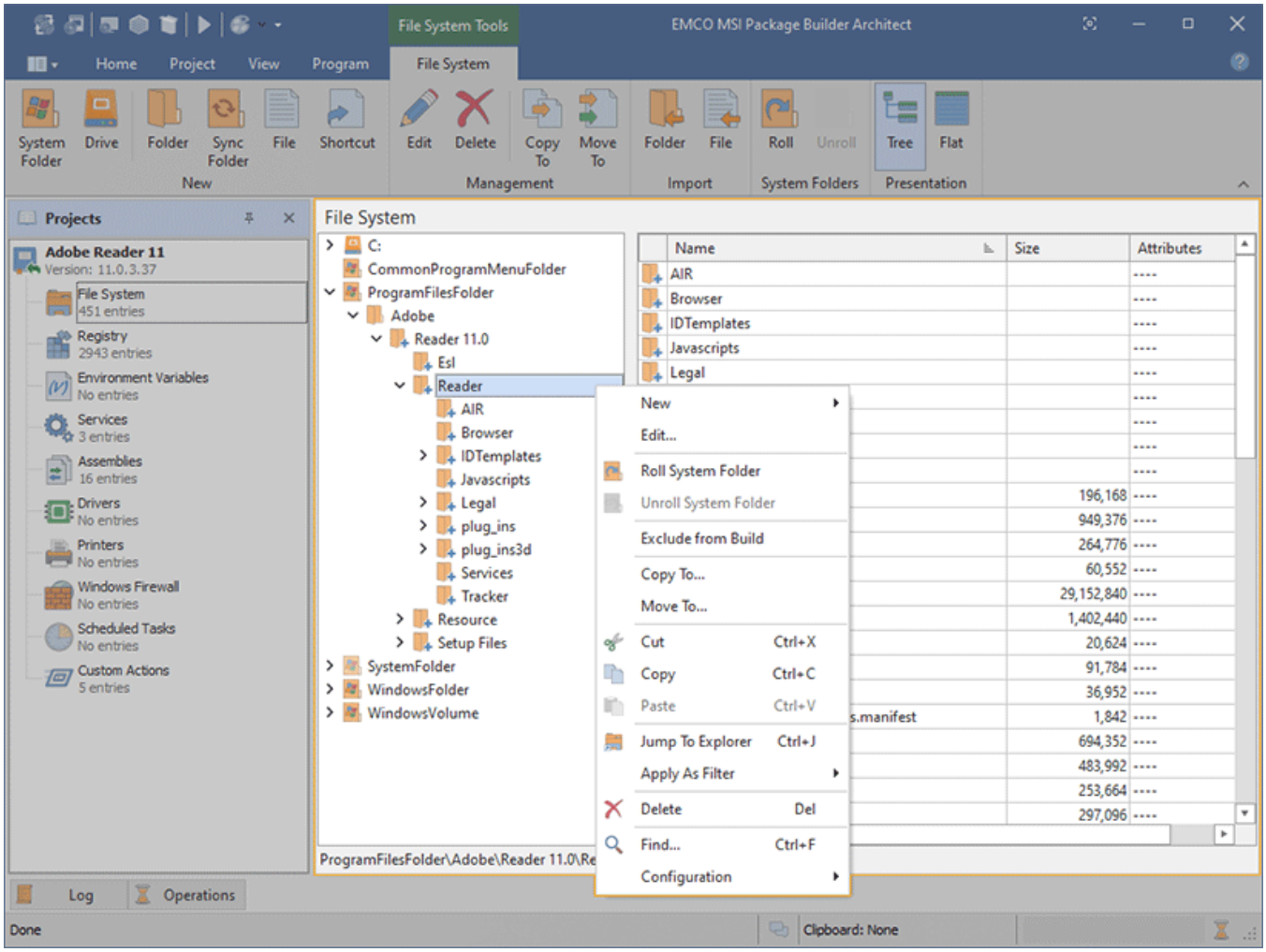
As for the **App-V** one, within the **General Properties** section, you may define the package GUID (witch is generated automatically by default) and the package description. The **Install Options** section allows you to define the supported operating systems and the model of interaction between the virtual application and the underlying operating system. The **Internal Package Info** section is used to maintain the App-V package history.

In the **MSIX/AppX** group, the **General Properties** section contains the fields to define a package name, a publisher for the package and a description. The **Applications** section is used to configure MSIX entry points and fixups. The **Target Platforms** section allows you to configure the minimum and maximum desktop and server operating system versions, that are supported by the package. The **Capabilities** section allows you to configure MSIX package capability options.

*Copyright © 2001-2023 EMCO. All rights reserved.*

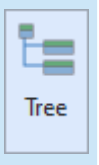
## File System View

The **File System** view is displayed within the main program area when the **File System** node of any project is selected in the **Projects** view. This view is used to define and review the changes to be performed to the file system when installing a deployment package created on a basis of this project Pic 1.




**Pic 1. The File System view**

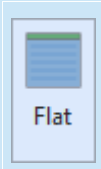
The file system modifications can include modifications of folders, files and shortcuts. The location of each element is represented with a file system path, that consists of drive name or system folder and several folders included into each other. When monitoring existing installations, the required changes are created automatically and added to the corresponding project representing the installation process. It is also possible to create the changes manually or import files and folder from local file system.




**Tree**

The **Tree** button from the **Presentation** group of the contextual **File System** tab should be used to display the file system modifications in form of a tree, similar to Windows Explorer.

Copyright © 2001-2023 EMCO. All rights reserved.




**Flat**

The **Flat** button from the **Presentation** group of the contextual **File System** tab allows you to display the file system modifications as a raw list, where each element is represented with its full path.

Information on the file system modifications can be represented both in form of a folder tree with data on each folder and in form of a resource list. When displayed as a resource list, it is by default grouped by processes responsible for the creation of each entry. For entries created manually, such a procedure is not applicable. You can clear the grouping using the table features, and it is possible to revert to the default layout at any time using the **Reset Layout** item from the configuration menu.

The icon next to every item represented in the **File System** view is used to describe of the item's type and state. The state icons are provided to help you understand what is currently happening in the program. You can always see if the item is being processed now by some operation or not, and if there are any processing problems.

Below is the list of icons used to represent the item type and state:

- a logical drive;
- a logical drive is being processed;
- a logical drive that contains a permanent path;
- a system folder;
- a system folder is being processed;
- a system folder that contains a permanent path;
- a folder;
- a folder is being processed;
- a folder is a part of a permanent path;
- a sync folder;
- a folder in a sync folder;
- a file;
- a file is permanent;
- a file in a sync folder;
- a shortcut.

The following overlays are used to represent the operation to be performed with an item:

- an item should be created;
- an item should be modified;
- an item should be deleted;
- permissions should be defined for an item;
- an item is excluded from the build.

As for the problematic situations, the following overlays are used:

- an item containing errors that should be resolved before creating a deployment package;

Copyright © 2001-2023 EMCO. All rights reserved.

 - an item containing changes that may lead to problems (e. g. removing of system-critical resources);

 - an item is either a missing link or contains missing links.

The actions for adding, editing and deleting the file system changes as well as copying and moving those changes between projects are available in the **File System** view pop-up menu and on the contextual **File System** Ribbon page.

## Functions Overview

| | |
|---|---|
| **Changes Management** | From the **File System** view, you can create, edit and delete the modifications to be performed by a deployment package to the file system. The actions for creating a new drive, system folder, folder, file and shortcut can be found within the **New** group on the contextual File System Ribbon page. The **System Folder** and **Drive** buttons from the **New** group on the **Project** page can also be used to create a new system folder and drive, respectively, together with the **New System Folder** and **New Drive** items in the pop-up menu displayed on an empty selection in the **File System** view. When a drive or folder is selected in the **File System** view, you can use the **New** sub-menu of the pop-up menu to create a new folder, sync folder, file or shortcut in the selected location. To change any file system modification, you can use the **Edit** item from the pop-up menu or the **Edit** button from the **Management** group on the contextual **File System** Ribbon page, and to delete the modification, use the **Delete** items. |
| **System Folders** | Within the **File System** view, it is possible to replace the common system folders with their definitions and vice-versa. The system folder definition is expanded to the actual path when a deployment package is being installed on a target PC. To replace a path, representing a system folder, with its definition placeholder, you can use the **Roll System Folder** menu item from the pop-up menu or the **Roll** button from the **System Folders** group on the contextual **File System** Ribbon page. To expand a system folder definition to the path that is defined on a current PC, use the **Unroll System Folder** menu item from the pop-up menu or the **Unroll** button from the **System Folders** group on the contextual **File System** Ribbon page. It is also possible to roll and unroll all system folders within a project using the **Roll All** and **Unroll All** buttons from the **System Folders** group on the **Project** Ribbon page. |
| **Copy/Move** | You can easily copy and/or move the modifications to be performed by a deployment package to the file system from the **File System** view. You can use the drag/drop and copy/paste techniques as well as the **Cut**, **Copy** and **Paste** menu items to reach the goal. It is also possible to use the **Copy To** and **Move To** items available both in the pop-up menu and on the contextual **File System** Ribbon page to perform copy/move immediately choosing a target project in a dialog. |

Copyright © 2001-2023 EMCO. All rights reserved.

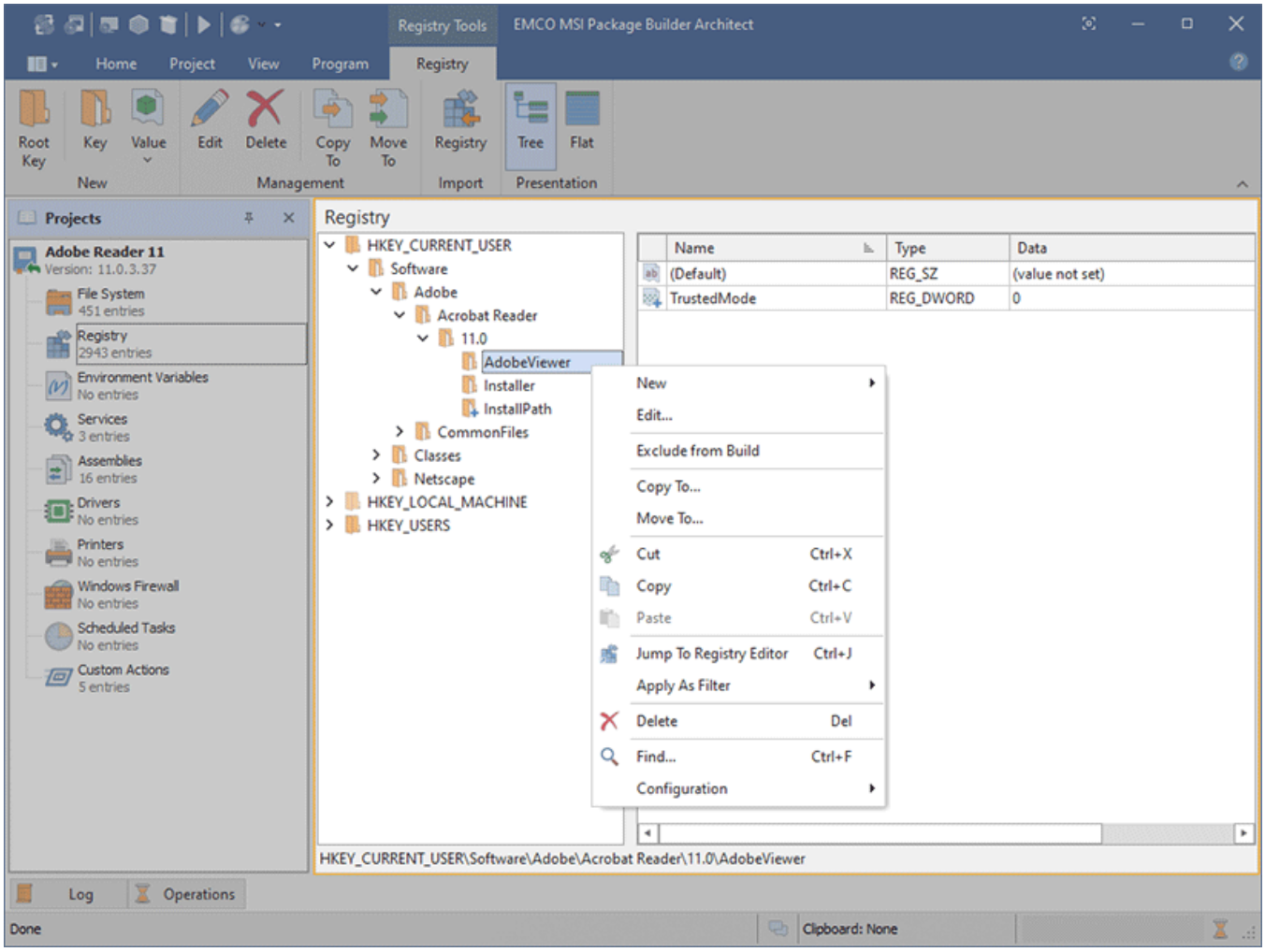
| | |
|---|---|
| **External Tools** | From the **File System** view, it is possible to open a location of any file system item in the Windows Explorer using the **Jump to Explorer** item from the pop-up menu, when a file system item is selected. When a file is selected, you can open this file using the **Open File** item from the pop-up menu. You can also import changes from a file system of the PC MSI Package Builder is running on. To perform import, you should either use the items available in the **Import** sub-menu of the pop-up menu displayed on an empty selection and from the **Import** group on the contextual **File System** Ribbon page, or the **File System** button from the Import group on the **Project** Ribbon page. |
| **Filters' Settings** | Any file system item from the **File System** view can be easily added either to monitoring or uninstall file filters using the **Apply As Filter** sub-menu from the **File System** view pop-up menu when a file or folder is selected. |
| **Presentation** | The changes to a file system performed by a deployment package can be displayed in the **File System** view in form of a tree or in form of a table. To switch between different data presentation models, you can use either the **Tree** and **Flat** buttons from the **Presentation** group on the contextual **File System** Ribbon page, or the **Presentation** sub-menu of the view pop-up menu displayed on an empty selection. |
| **Search** | Within the **File System** view, you can execute a search for specific changes to the file system performed by a deployment package using the **Find** item from the pop-up menu. |

For detailed information on the changes to a file system that can be defined in a project, refer to the File System Modifications section of this document.

Copyright © 2001-2023 EMCO. All rights reserved.

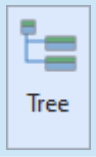
## Registry View

The **Registry** view is displayed within the main program area when the **Registry** node of any project is selected in the **Projects** view. This view is used to define and review the changes to be performed to the Windows registry when installing a deployment package created on a basis of this project Pic 1.



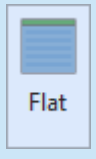

**Pic 1. The Registry view**

The registry modifications can include modifications of keys and values. The location of each element is represented with a registry path, that consists of root key and several keys included into each other. When monitoring existing installations, the required changes are created automatically and added to the corresponding project representing the installation process. It is also possible to create the changes manually or import changes from a registration entries (.reg) file.




**Tree**

The **Tree** button from the **Presentation** group of the contextual **Registry** tab should be used to display the registry modifications in form of a tree, similar to Registry Editor.




**Flat**

The **Flat** button from the **Presentation** group of the contextual **Registry** tab allows you to display the registry modifications as a raw list, where each element is represented with its full path.

Copyright © 2001-2023 EMCO. All rights reserved.

Information on the registry modifications can be represented both in form of a keys tree with data on each key and in form of a resource list. When displayed as a resource list, it is by default grouped by processes responsible for the creation of each entry. For the entries created manually, such a procedure is not applicable. You can clear the grouping using the table features, and it is possible to revert to the default layout at any time using the **Reset Layout** item from the configuration menu.

The icon next to every item represented in the **Registry** view is used to describe of the item's type and state. The state icons are provided to help you understand what is currently happening in the program. You can always see if the item is being processed now by some operation or not, and if there are any processing problems.

Below is the list of icons used to represent the item type and state:

 - a root key;

 - a root key containing permanent registry keys or values;

 - a registry key;

 - a registry key is a part of a permanent path;

 - a string registry value;

 - a string registry value is permanent;

 - a binary registry value;

 - a binary registry value is permanent;

 - a DWORD registry value;

 - a DWORD registry value is permanent;

 - a QWORD registry value;

 - a QWORD registry value is permanent;

 - a multi-string registry value;

 - a multi-string registry value is permanent;

 - an expandable string registry value;

 - an expandable string registry value is permanent;

The following overlays are used to represent the operation to be performed with a registry key or value:

 - an item should be created;

 - an item should be modified;

 - an item should be deleted;

 - permissions should be defined for a key;

 - an item is excluded from the build.

As for the problematic situations, only the following overlay is used:

 - an item containing changes that may lead to problems (e. g. removing of system-critical resources).

Copyright © 2001-2023 EMCO. All rights reserved.

The actions for adding, editing and deleting the registry changes as well as copying and moving those changes between projects are available in the **Registry** view pop-up menu and on the contextual **Registry** Ribbon page.

## Functions Overview

| | |
|---|---|
| **Changes Management** | From the **Registry** view, you can create, edit and delete the modifications to be performed by a deployment package to the Windows registry. The actions for creating a new root key, registry key and registry values of different types can be found within the **New** group on the contextual **Registry** Ribbon page. The **Root Key** button from the **New** group on the **Project** page can also be used to create a new key together with the **New Root Key** item in the pop-up menu displayed on an empty selection in the **Registry** view. When a registry key is selected in the **Registry** view, you can use the **New** sub-menu of the pop-up menu to create a new key or a new value of a specific type within the selected key. To change any registry modification, you can use the **Edit** item from the pop-up menu or the **Edit** button from the **Management** group on the contextual **Registry** Ribbon page, and to delete the modification, use the **Delete** items. As for the default values for each key, you can also change the value type using the **Change Type** sub-menu from the Registry view when a default value is selected. |
| **Copy/Move** | You can easily copy and/or move the modifications to be performed by a deployment package to the Windows registry from the **Registry** view. You can use the drag/drop and copy/paste techniques as well as the **Cut**, **Copy** and **Paste** menu items to reach the goal. It is also possible to use the **Copy To** and **Move To** items available both in the pop-up menu and on the contextual **Registry** Ribbon page to perform copy/move immediately choosing a target project in a dialog. |
| **External Tools** | From the **Registry** view, it is possible to open a location of any key or value in the **Registry Editor** using the **Jump to Registry Editor** item from the pop-up menu, when a key or value is selected. You can also import the changes to a Windows registry from a registration entries (.reg) file. To perform import, you should either choose the **Registry** button available in the **Import** group on both the regular **Project** and contextual **Registry** Ribbon pages, or choose the **Import** item from the pop-up menu displayed on an empty selection. |
| **Filters' Settings** | Any key or value from the **Registry** view can be easily added either to monitoring or uninstall registry filters using the **Apply As Filter** sub-menu from the **Registry** view pop-up menu when a key or value is selected. |
| **Presentation** | The changes to a Windows registry performed by a deployment package can be displayed in the **Registry** view in form of a tree or in form of a table. To switch between different data presentation models, you can use either the **Tree** and **Flat** buttons from the **Presentation** group on the contextual **Registry** Ribbon page, or the **Presentation** sub-menu of the view pop-up menu displayed on an empty selection. |
| **Search** | Within the **Registry** view, you can execute a search for specific changes to the Windows registry performed by a deployment package using the **Find** item from the pop-up menu. |

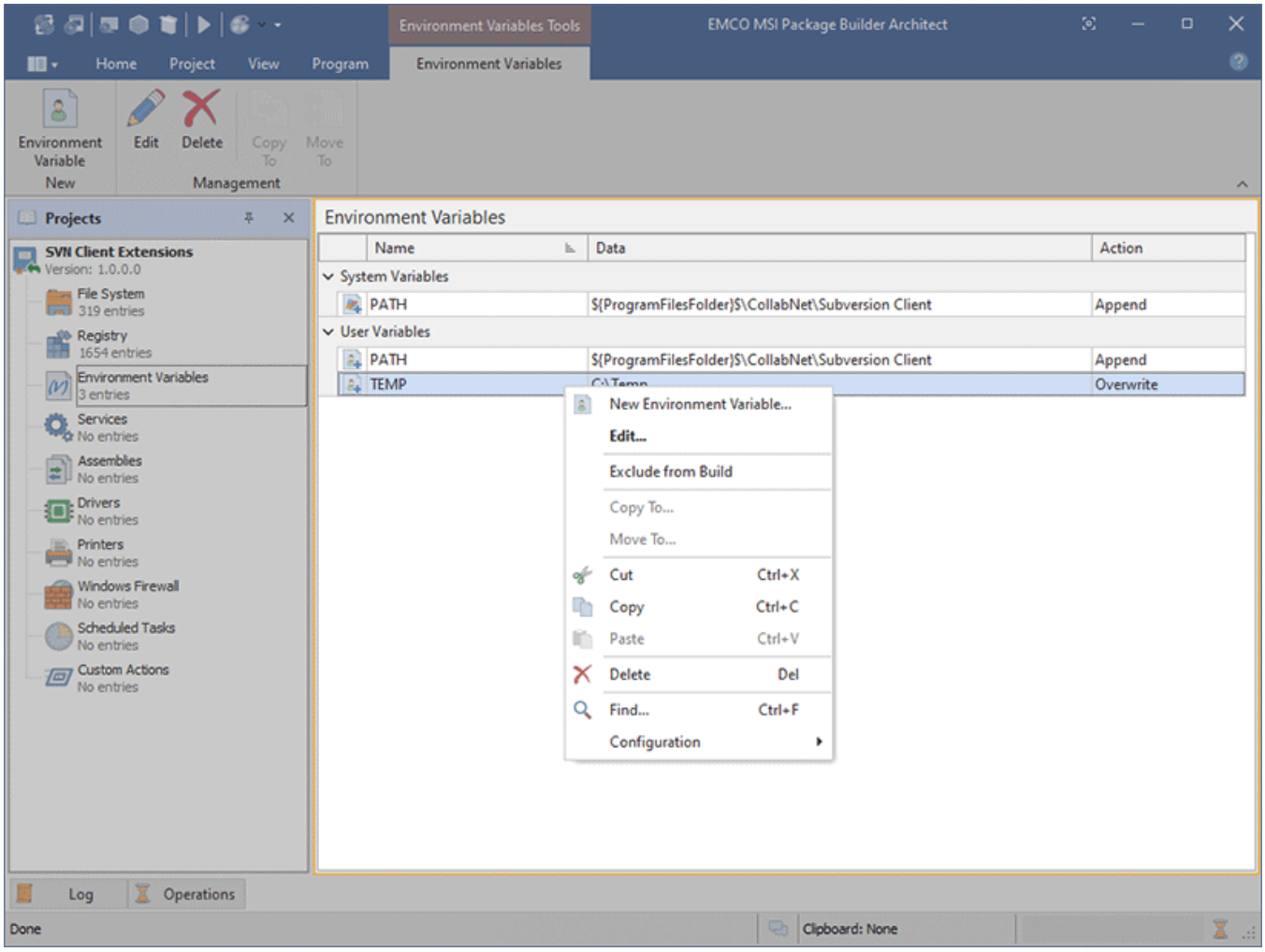
*Copyright © 2001-2023 EMCO. All rights reserved.*

For detailed information on the changes to a registry that can be defined in a project, refer to the Registry Modifications section of this document.

Copyright © 2001-2023 EMCO. All rights reserved.

## Environment Variables View

The **Environment Variables** view is displayed within the main program area when the **Environment Variables** node of any project is selected in the **Projects** view. This view is used to define and review the changes to be performed to the environment variables when installing a deployment package created on a basis of this project Pic 1.




**Pic 1. The Environment Variables view**

*Copyright © 2001-2023 EMCO. All rights reserved.*

The environment variables modifications can include modifications to user and system variables. When monitoring existing installations, the required changes are created automatically and added to the corresponding project representing the installation process. It is also possible to create the changes manually.

MSIX/AppX packages do not support configuring the environment variables during deployment, so, when generated, those changes are included into MSI and App-V deployment packages only.

The environment variables modifications are by default grouped by the variable type. You can clear the grouping using the table features, and it is possible to revert to the default layout at any time using the **Reset Layout** item from the configuration menu.

The icon next to every item represented in the **Environment Variables** view is used to describe of the item's type and state. The state icons are provided to help you understand what is currently happening in the program. You can always see if the item is being processed now by some operation or not.

Below is the list of type icons used:

 - a user environment variable;

 - a system environment variable;

The following overlays are used to represent the operation to be performed with each variable when installing a generated deployment package:

 - a variable should be created;

 - a variable should be created, if it does not exist;

 - a variable should be removed;

 - a variable should be removed, if its value matches the defined value;

 - a variable is excluded from the build.

The actions for adding, editing and deleting the changes to environment variables as well as copying and moving those changes between projects are available in the **Environment Variables** view pop-up menu and on the contextual **Environment Variables** Ribbon page.

## Functions Overview

| | |
|---|---|
| **Changes Management** | From the **Environment Variables** view, you can create, edit and delete the modifications to be performed by a deployment package to the environment variables. The **New Environment Variable** item from the **Environment Variables** view pop-up menu as well as the **Environment Variable** button from the **New** group on the contextual **Environment Variables** Ribbon page can be used to create a new modification to environment variables. It is possible to create a new variable, append/prepend data to existing variables and remove variables together with a deployment package deployment process. To change any environment variables modification, you can use the **Edit** item from the pop-up menu or the **Edit** button from the **Management** group on the contextual **Environment Variables** Ribbon page, and to delete the modification, use the **Delete** items. |

Copyright © 2001-2023 EMCO. All rights reserved.

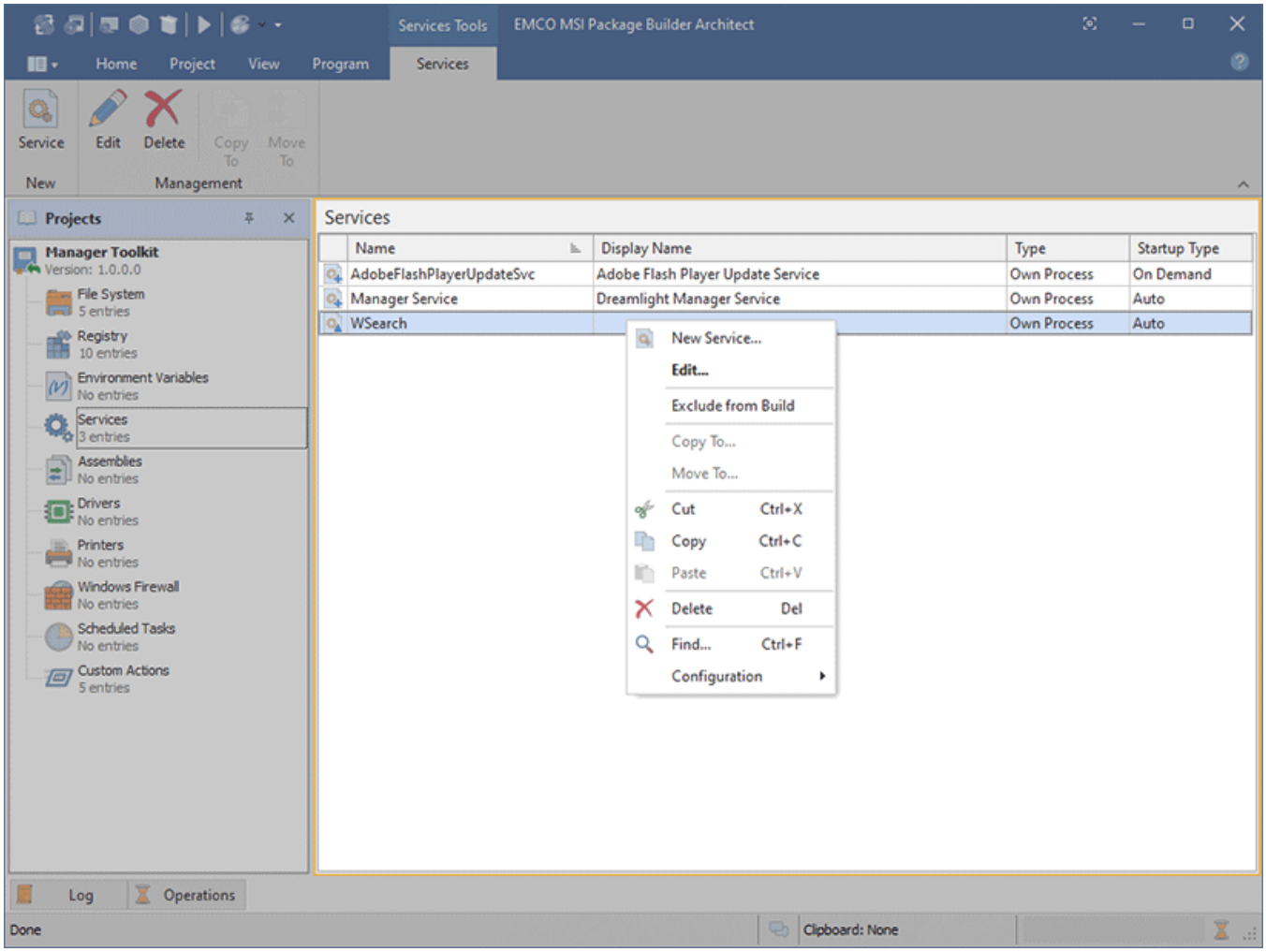
| | |
|---|---|
| **Copy/Move** | You can easily copy and/or move the modifications to be performed by a deployment package to the environment variables from the Environment **Variables** view. You can use the drag/drop and copy/paste techniques as well as the **Cut**, **Copy** and **Paste** menu items to reach the goal. It is also possible to use the **Copy To** and **Move To** items available both in the pop-up menu and on the contextual **Environment Variables** Ribbon page to perform copy/move immediately choosing a target project in a dialog. |
| **Search** | Within the **Environment Variables** view, you can execute a search for specific changes to the environment variables performed by a deployment package using the Find item from the pop-up menu. |

For detailed information about the changes to environment variables that can be defined in a project refer to the Environment Variables Modifications section of this document.

Copyright © 2001-2023 EMCO. All rights reserved.

## Services View

The **Services** view is displayed within the main program area when the **Services** node of any project is selected in the **Projects** view. This view is used to define and review the changes to be performed to the installed services when installing and/or uninstalling a deployment package created on a basis of this project Pic 1.




**Pic 1. The Services view**

Copyright © 2001-2023 EMCO. All rights reserved.

MSI Package Builder allows you to create, delete and control Windows services. When monitoring existing installations, the required changes are created automatically and added to the corresponding project representing the installation process. It is also possible to create the changes manually.

Configuring services during deployment is not supported by App-V and MSIX/AppX packages, so, when generated, those changes are included into the MSI deployment packages only.

The icon next to every item represented in the **Services** view is used to describe of the item's type and state. The state icons are provided to help you understand what is currently happening in the program. You can always see if the item is being processed now by some operation or not, and if there are any processing problems.

Below is the list of type icons used:

 - a service;

 - a service containing errors that should be resolved before creating a deployment package.

The following overlays are used to represent the operation to be performed with each service:

 - a service should be created;

 - a service should be deleted;

 - a service should be controlled (started and/or stopped);

 - a service should be restarted;

 - permissions should be defined for a service;

 - a service is excluded from the build.

The actions for adding, editing and deleting the changes to Windows services as well as copying and moving those changes between projects are available in the **Services** view pop-up menu and on the contextual **Services** Ribbon page.

## Functions Overview

| | |
|---|---|
| **Changes Management** | From the **Services** view, you can create, edit and delete the modifications to be performed by a deployment package to the Windows services configuration. The **New Service** item from the **Services** view pop-up menu and the **Service** button from the **New** group on the contextual **Service** Ribbon page can be used to create a new modification to Windows services. It is possible to create the service together with the package deployment process or control any service that is already installed when installing and/or uninstalling the deployment package. To change any service modification, you can use the **Edit** item from the pop-up menu or the **Edit** button from the **Management** group on the contextual **Services** Ribbon page, and to delete the modification, use the Delete items. |

Copyright © 2001-2023 EMCO. All rights reserved.

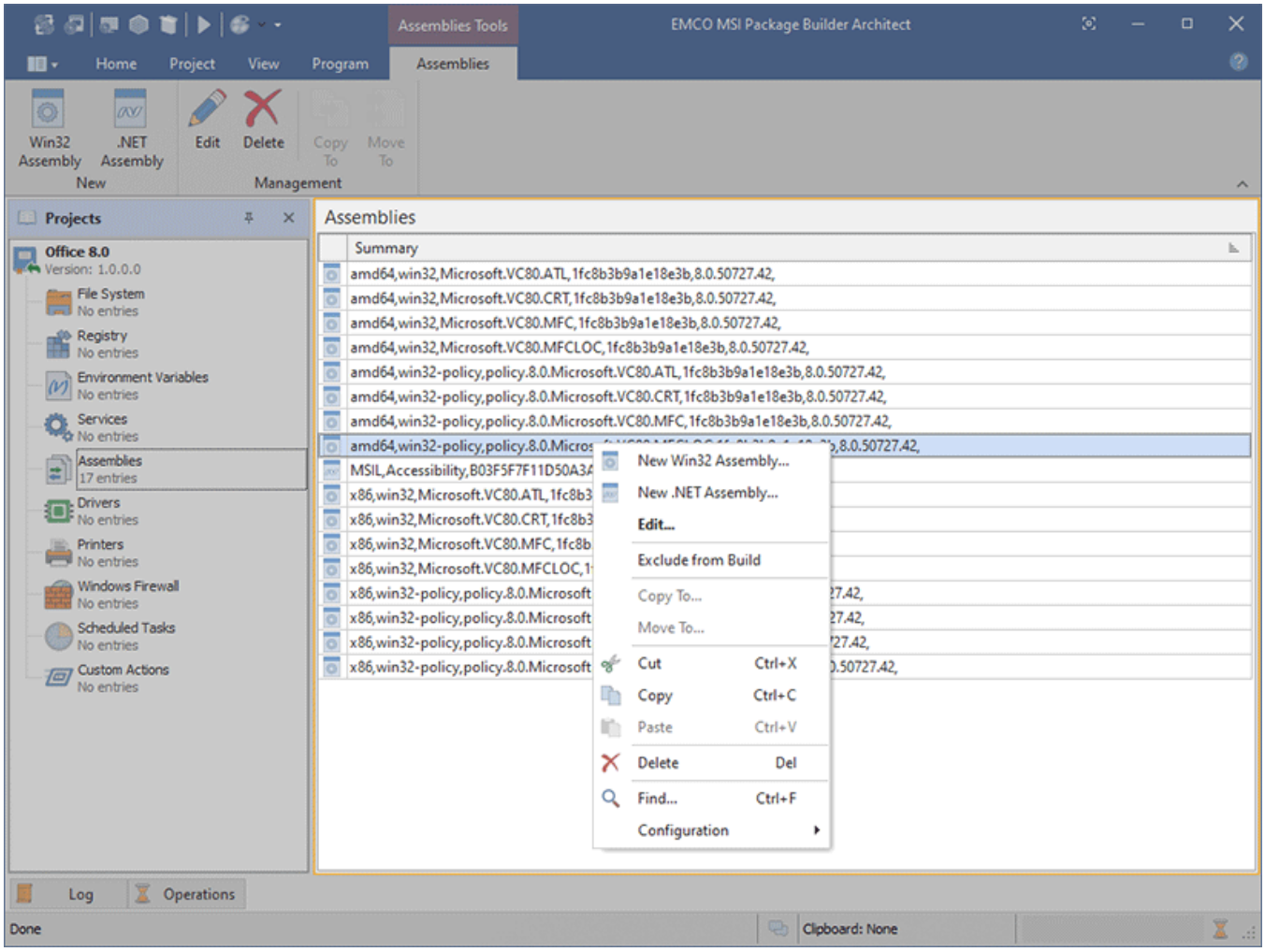
| | |
|---|---|
| **Copy/Move** | You can easily copy and/or move the modifications to be performed by a deployment package to the Windows services configuration from the **Services** view. You can use the drag/drop and copy/paste techniques as well as the **Cut**, **Copy** and **Paste** menu items to reach the goal. It is also possible to use the **Copy To** and **Move To** items available both in the pop-up menu and on the contextual **Services** Ribbon page to perform copy/move immediately choosing a target project in a dialog. |
| **Search** | Within the **Services** view, you can execute a search for specific changes to the Windows services configuration performed by a deployment package using the Find item from the pop-up menu. |

For detailed information on the changes to Windows services that can be defined in a project, refer to the Services Modifications section of this document.

Copyright © 2001-2023 EMCO. All rights reserved.

## Assemblies View

The **Assemblies** view is displayed within the main program area when the **Assemblies** node of any project is selected in the **Projects** view. This view is used to configure a set of side-by-side .NET and/or Win32 assemblies that are installed by the generated deployment package Pic 1.




**Pic 1. The Assemblies view**

This view is used to configure a set of side-by-side .NET and/or Win32 assemblies that are installed by the generated deployment package.

MSIX/AppX packages do not support side-by-side assemblies deployment, so, when generated, those changes are included into MSI and App-V deployment packages only.

The assemblies are displayed in form of a table, where each row represents a single assembly. The icon next to every assembly is used to describe of the assembly's type and state. The state icons are provided to help you understand what is currently happening in the program. You can always see if the item is being processed now by some operation or not and if there are any problems.

Below is the list of icons used to represent the item type and state:

- a .NET assembly;
- a Win32 assembly;
- an assembly is excluded from the build.

Copyright © 2001-2023 EMCO. All rights reserved.

As for the problematic situations, the following overlays are used:

 - an assembly item containing errors that should be resolved before creating a deployment package;

 - an assembly item contains missing links.

The actions for adding new assemblies, editing and deleting existing ones and others are available in the **Assemblies** view pop-up menu and on the contextual **Assemblies** Ribbon page.

## Functions Overview

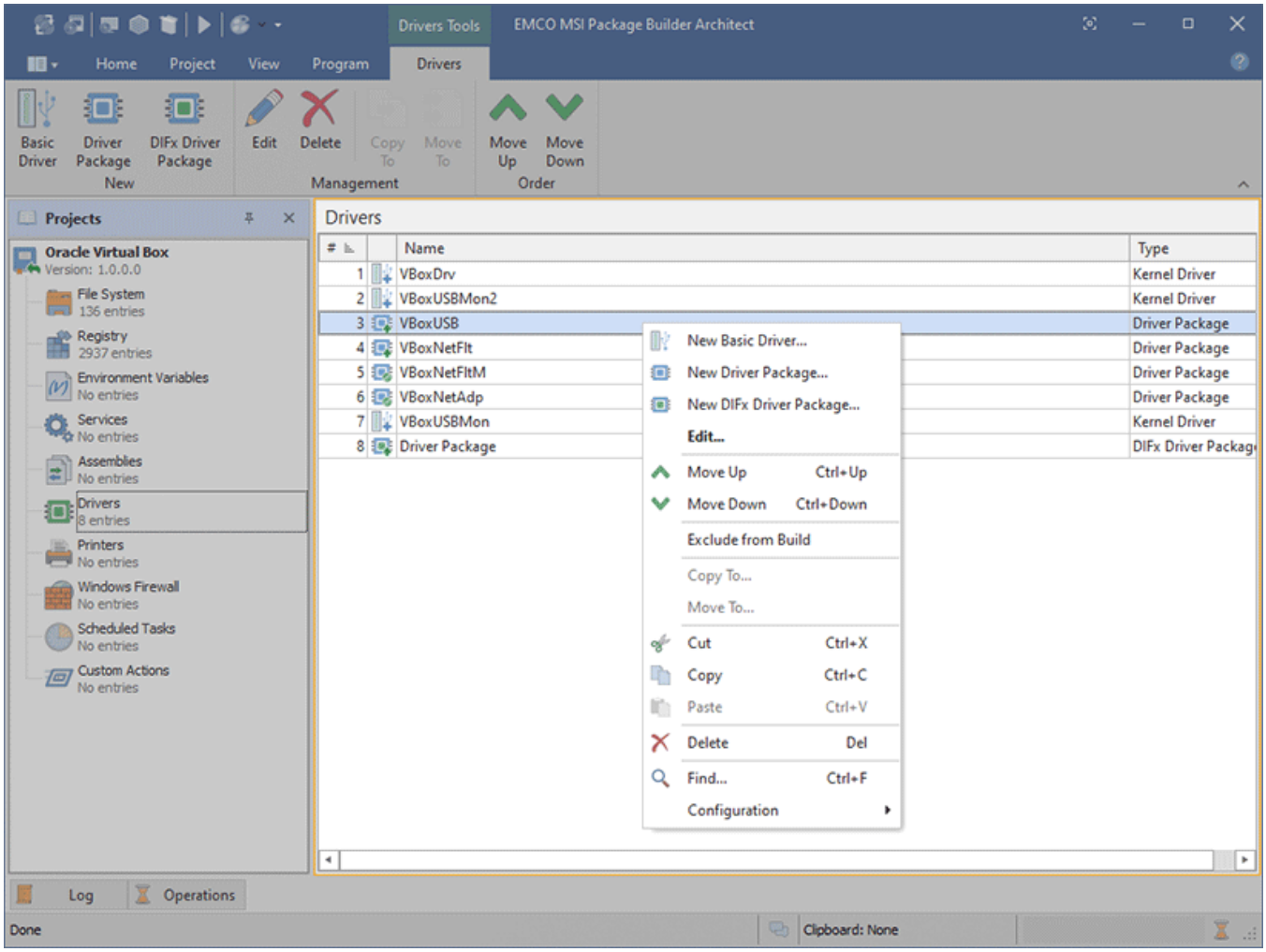
| | |
|---|---|
| **Assemblies Management** | From the **Assemblies** view, you can add, edit and delete the .NET and/or Win32 side-by-side assemblies to be installed by a generated deployment package. To add a new assembly, you can either choose the **New Win32 Assembly/New .NET Assembly** item from the pop-up menu, or press the **Win32 Assembly/.NET Assembly** button from the **New** group on the contextual **Assemblies** Ribbon page and on the **Project** Ribbon page. The **Edit** item from the pop-up menu, as well as the **Edit** button from the **Management** group on the contextual **Assemblies** Ribbon page can be used to change the selected assembly, and to delete any assembly, you can use the **Delete** items. |
| **Copy/Move** | You can easily copy and/or move the .NET and/or Win32 side-by-side assemblies to be installed by a generated deployment package from the **Assemblies** view to another project. You can use the drag/drop and copy/paste techniques as well as the **Cut**, **Copy** and **Paste** menu items to reach the goal. It is also possible to use the **Copy To** and **Move To** items available both in the pop-up menu and on the contextual **Assemblies** Ribbon page to perform copy/move immediately choosing a target project in a dialog. |
| **Search** | Within the **Assemblies** view, you can execute a search for specific assembly using the **Find** item from the pop-up menu. |

For detailed information on the side-by-side assemblies and their management process, refer to the Side-by-side Assemblies Deployment section of this document.

Copyright © 2001-2023 EMCO. All rights reserved.

## Drivers View

The **Drivers** view is displayed within the main program area when the **Drivers** node of any project is selected in the **Projects** view. This view is used to define and review the changes to be performed to the device drivers when installing and/or uninstalling a deployment package created on a basis of this project Pic 1.




**Pic 1. The Drivers view**

MSI Package Builder allows you to create and delete basic drivers as well as install and pre-install driver packages. Both basic drivers and drivers from packages can be registered as device class filters by MSI Package Builder if it is required. When monitoring existing installations, the required changes are created automatically and added to the corresponding project representing the installation process. It is also possible to create the changes manually.

Configuring device drivers during deployment is not supported by App-V and MSIX/AppX packages, so, when generated, those changes are included into the MSI deployment packages only.

Copyright © 2001-2023 EMCO. All rights reserved.

The icon next to every item represented in the **Drivers** view is used to describe of the item's type and state. The state icons are provided to help you understand what is currently happening in the program. You can always see if the item is being processed now by some operation or not, and if there are any processing problems.

Below is the list of type icons used:

 - a basic driver;

 - a driver package;

 - a DIFx driver package.

As for the problematic situations, the following overlays are used:

 - a basic driver containing errors that should be resolved before creating a deployment package;

 - a driver package containing errors that should be resolved before creating a deployment package;

 - a driver package contains missing links;

 - a DIFx driver package contains missing links.

The following overlays are used to represent the operation to be performed with each driver:

 - a basic driver should be created;

 - a basic driver should be deleted;

 - permissions should be defined for a basic driver;

 - a driver package should be installed;

 - a driver package should be pre-installed;

 - a DIFx driver package should be installed;

 - a DIFx driver package should be pre-installed;

 - a driver is excluded from the build.

The actions for adding, editing and deleting the changes to device drivers as well as copying and moving those changes between projects are available in the **Drivers** view pop-up menu and on the contextual **Drivers** Ribbon page.

## Functions Overview

| | |
|---|---|
| **Changes Management** | From the **Drivers** view, you can create, edit and delete the modifications to be performed by a deployment package to the device drivers configuration. The **New Basic Driver**, the **New Driver Package** and the **New DIFx Driver Package** items from the **Drivers** view pop-up menu, as well as the **Basic Driver**, the **Driver Package** and the **DIFx Driver Package** buttons from the **New** group on the contextual **Drivers** Ribbon page can be used to create a new modification to device drivers. To change any device drivers modification, you can use the **Edit** item from the pop-up menu or the **Edit** button from the **Management** group on the **Drivers** contextual Ribbon page, and to delete the modification, use the **Delete** items. |

Copyright © 2001-2023 EMCO. All rights reserved.

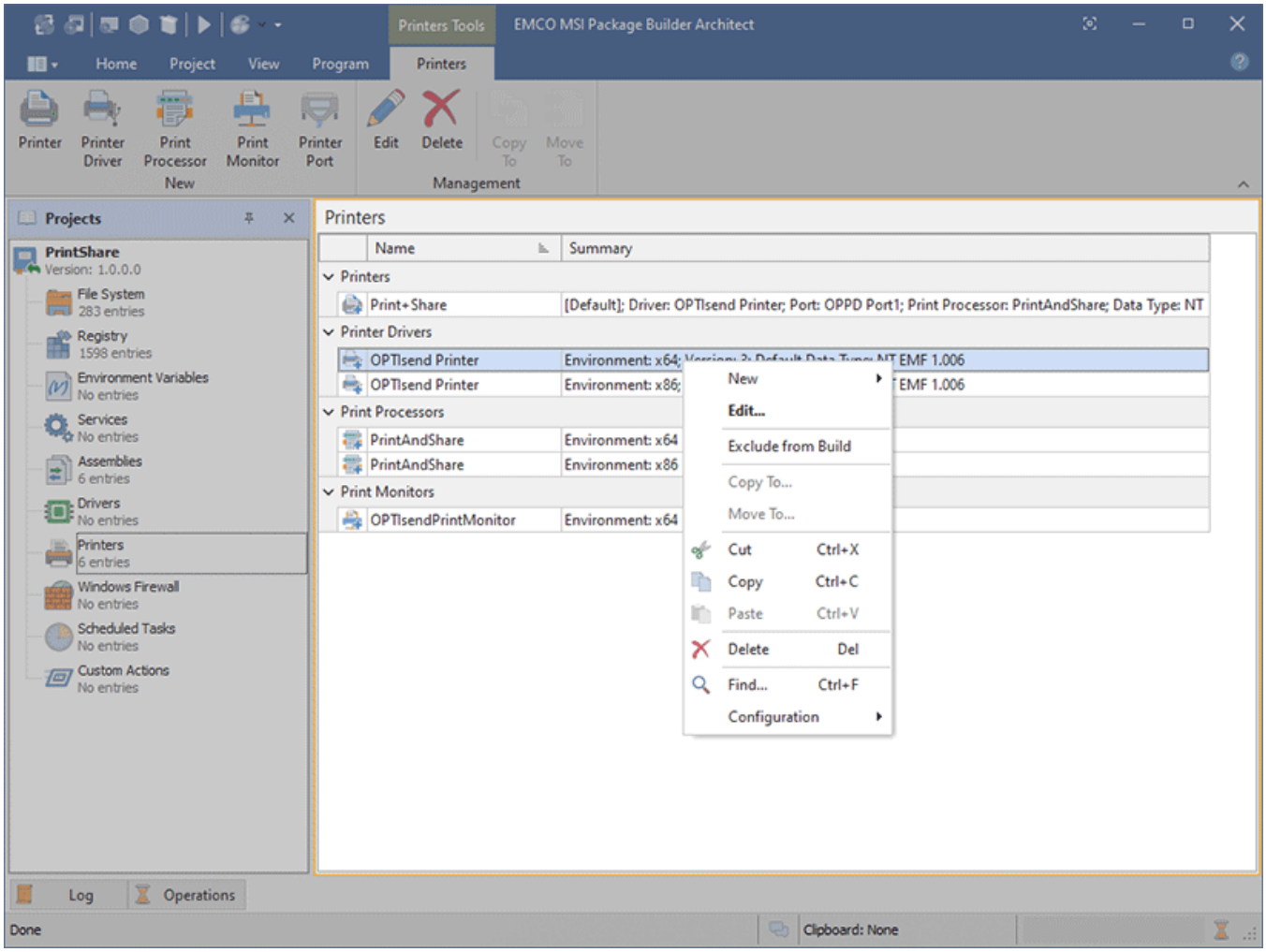
| | |
|---|---|
| **Copy/Move** | You can easily copy and/or move the modifications to be performed by a deployment package to the device drivers configuration from the **Drivers** view. You can use the drag/drop and copy/paste techniques as well as the **Cut**, **Copy** and **Paste** menu items to reach the goal. It is also possible to use the **Copy To** and **Move To** items available both in the pop-up menu and on the contextual **Drivers** Ribbon page to perform copy/move immediately choosing a target project in a dialog. |
| **Search** | Within the **Drivers** view, you can execute a search for specific changes to the device drivers configuration performed by a deployment package using the **Find** item from the pop-up menu. |

For detailed information on the changes to device drivers that can be defined in a project, refer to the Drivers Deployment section of this document.

Copyright © 2001-2023 EMCO. All rights reserved.

## Printers View

The **Printers** view is displayed within the main program area when the **Printers** node of any project is selected in the **Projects** view. This view is used to define and review the changes to be performed to the printing system when installing and/or uninstalling a deployment package created based on this project Pic 1.




**Pic 1. The Printers view**

MSI Package Builder allows you to perform the following changes to the printing system: create, modify and delete printers; install and uninstall printer drivers; add and remove print processors, print monitors and printer ports. When monitoring existing installations, required changes are created automatically and added to the corresponding project representing the installation process. It is also possible to create changes manually.

App-V and MSIX/AppX packages do not support configuring the printing system during deployment, so, when generated, those changes are included into MSI deployment packages only.

Copyright © 2001-2023 EMCO. All rights reserved.

By default, items in the view are grouped by the entry type. You can change the grouping using the table features, and it is possible to revert to the default layout at any time using the **Reset Layout** item from the configuration menu.

The icon next to every item represented in the **Printers** view is used to describe the item's type and state. The state icons are provided to help you understand what is happening in the program at the moment. You can always see if the item is being processed now by some operation or not, and if there are any processing problems.

Below is the list of the type icons used:

 - a printer;

 - a printer driver;

 - a print processor;

 - a print monitor;

 - a local printer port;

 - a TCP/IP printer port;

 - a printer is excluded from the build.

In case of problems, the following overlays are used:

 - a printer containing errors that should be resolved before creating a deployment package;

 - a printer driver containing errors that should be resolved before creating a deployment package;

 - a print processor containing errors that should be resolved before creating a deployment package;

 - a print monitor containing errors that should be resolved before creating a deployment package;

 - a printer driver contains missing links;

 - a print processor contains missing links.

The following overlays are used to represent the operation to be performed with each item:

 - a printer should be created;

 - a printer should be modified;

 - a printer should be deleted;

 - a printer driver should be installed;

 - a printer driver should be uninstalled;

 - a print processor should be added;

 - a print processor should be removed;

 - a print monitor should be added;

 - a print monitor should be removed;

 - a local printer port should be created;

 - a local printer port should be deleted;

 - a TCP/IP printer port should be created;

Copyright © 2001-2023 EMCO. All rights reserved.

 - a TCP/IP printer port should be modified;

 - a TCP/IP printer port should be deleted.

The actions for adding, editing and deleting the changes to the printing system as well as copying and moving those changes between projects are available in the **Printers** view pop-up menu and on the **Printers** contextual Ribbon page.

## Functions Overview

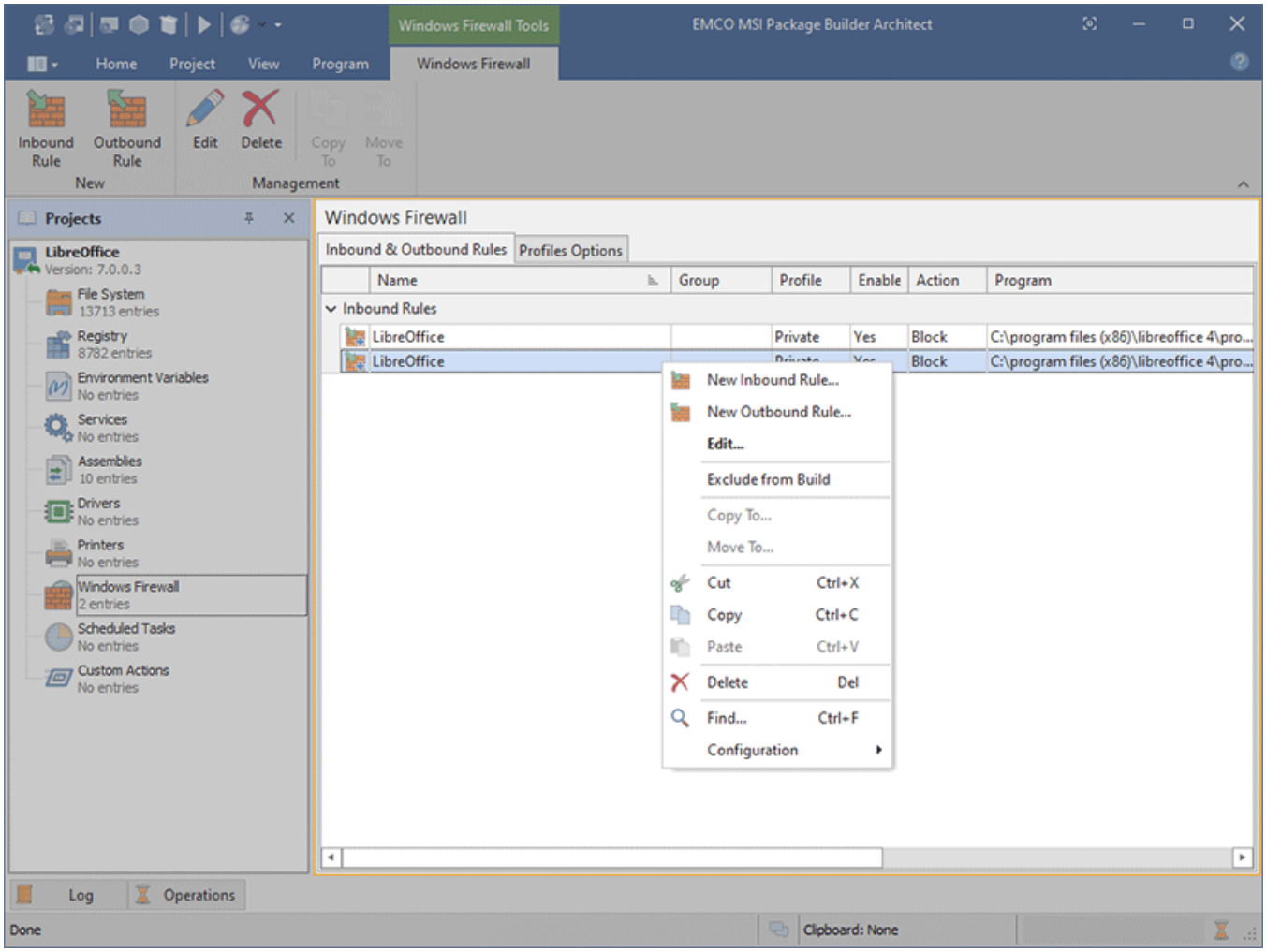
| | |
|---|---|
| **Changes Management** | From the **Printers** view you can create, edit and delete the modifications to be performed by a deployment package to the printing system configuration. The actions for creating changes in printers, printer drivers, print processors, print monitors and printer ports can be found within the **New** group on the **Printers** contextual Ribbon page and in the **New** group of the **Printers** view pop-up menu. Alternatively you can use the **Printer Entries** drop-down button from the **New** group on the **Project** Ribbon page. To change any modification to printing system, you can use the **Edit** item from the pop-up menu or the **Edit** button from the **Management** group on the **Printers** contextual Ribbon page, and to delete the modification, use the **Delete** items. |
| **Copy/Move** | You can easily copy and/or move the modifications to be performed by a deployment package to the printing system configuration from the Printers view. You can use the drag/drop and copy/paste techniques as well as the **Cut**, **Copy** and **Paste** menu items to reach the goal. It is also possible to use the **Copy To** and **Move To** items available both in the pop-up menu and on the **Printers** contextual Ribbon page to perform copy/move immediately choosing a target project in a dialog. |
| **Search** | Within the **Printers** view you can execute a search for specific changes to the printing system configuration performed by a deployment package using the **Find** item from the pop-up menu. |

For detailed information on changes to the printing system that can be defined in a project, refer to the Printers Deployment section of this document.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Windows Firewall View

The **Windows Firewall** view is displayed within the main program area when the **Windows Firewall** node of any project is selected in the **Projects** view. This view is used to manage Windows firewall rules and settings when installing and/or uninstalling a deployment package created based on this project Pic 1.




**Pic 1. The Windows Firewall view**

MSI Package Builder allows you to create, modify and delete inbound and outbound rules of Windows Firewall and change settings of Windows Firewall profiles. During existing installations monitoring, the required changes are created automatically and added to the corresponding project representing the installation process. It is also possible to create the changes manually. Windows Firewall changes can be managed on the **Inbound & Outbound Rules** and **Profiles Options** tabs of the view.

The **Inbound & Outbound Rules** tab shows the list of the firewall rules configured in the project. The icon next to every item represented in the **Windows Firewall** view is used to describe the item's type and state. The state icons are provided to help you understand what is currently happening in the program. You can always see if the item is being processed by some operation now or not, and if there are any processing problems.

Below is the list of the type icons used:

- a firewall rule;

Copyright © 2001-2023 EMCO. All rights reserved.

 - a firewall rule is excluded from the build.

The following overlays are used to represent the operation to be performed with each firewall rule:

 - an inbound firewall rule should be created;
 - an inbound firewall rule should be modified;
 - an inbound firewall rule should be deleted;
 - an inbound firewall rule should be deleted by name;
 - an outbound firewall rule should be created;
 - an outbound firewall rule should be modified;
 - an outbound firewall rule should be deleted;
 - an outbound firewall rule should be deleted by name;

The actions for adding, editing and deleting changes to Windows Firewall rules as well as copying and moving such changes among projects are available in the **Windows Firewall** view pop-up menu and on the contextual **Windows Firewall** Ribbon page.

## Functions Overview

| | |
|---|---|
| **Changes Management** | From the **Windows Firewall** view, you can create, edit and delete modifications to be performed by a deployment package to the **Windows Firewall** configuration. Actions for creating changes in firewall rules can be found within the **New** group on the **Windows Firewall** contextual Ribbon page and the corresponding buttons of the **Windows Firewall** view pop-up menu. Alternatively, you can use the **Inbound Rule** and **Outbound Rule** buttons from the **New** group on the **Project** Ribbon page. To change any modification to firewall rules, you can use the **Edit** item from the pop-up menu or the **Edit** button from the **Management** group on the **Windows Firewall** contextual Ribbon page, and to delete the modification, use the **Delete** items. |
| **Copy/Move** | You can easily copy and/or move the modifications to be performed by a deployment package to the Windows Firewall configuration from the Windows Firewall view. You can use the drag/drop and copy/paste techniques as well as the **Cut**, **Copy** and **Paste** menu items for this purpose. It is also possible to use the **Copy To** and **Move To** items available both in the pop-up menu and on the **Windows Firewall** contextual Ribbon page to perform copy/move immediately choosing a target project in a dialog. |
| **Search** | Within the **Windows Firewall** view, you can search for specific changes to the firewall rules configuration performed by a deployment package using the **Find** item from the pop-up menu. |

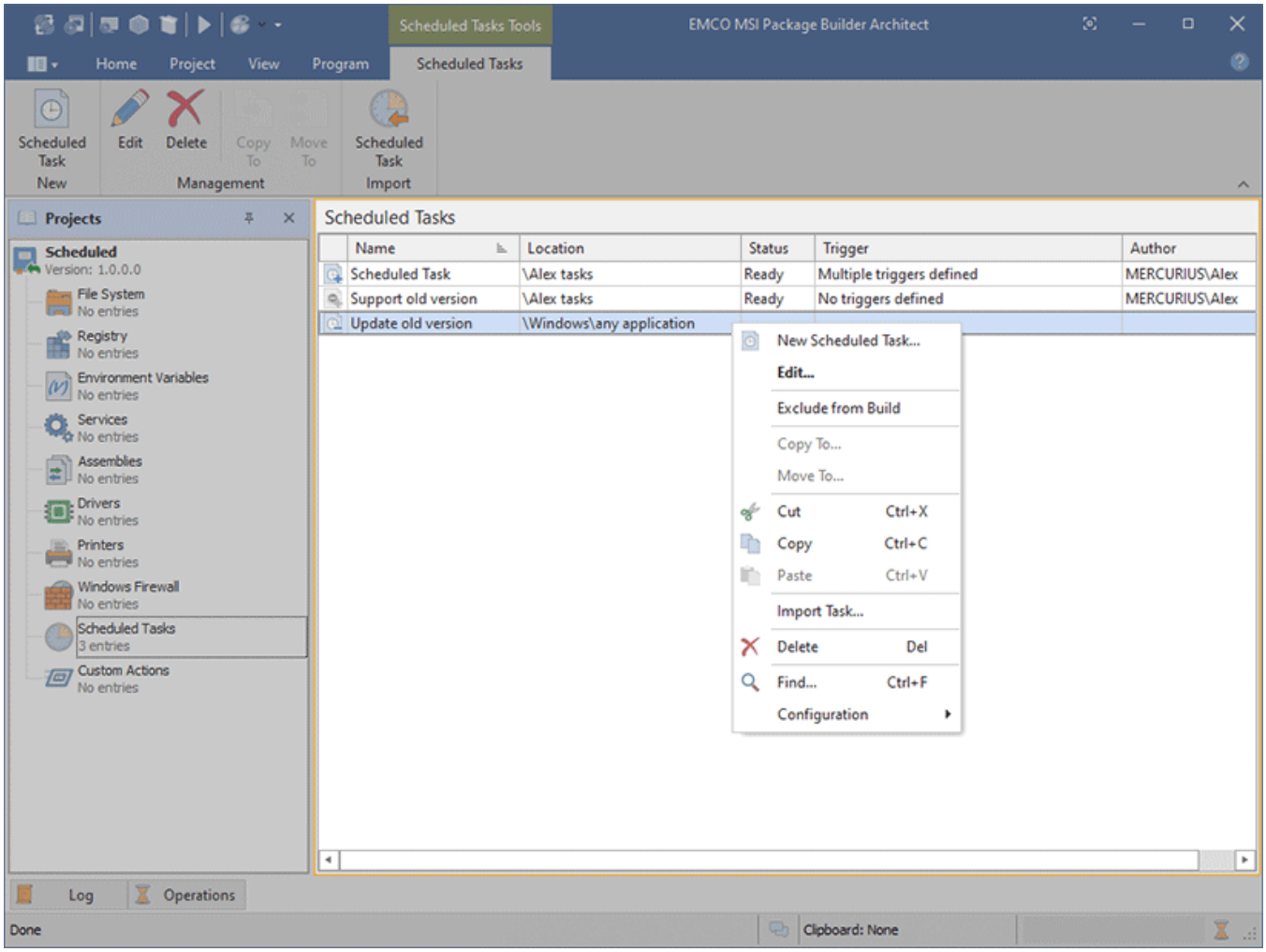
The **Profiles Options** tab of the **Windows Firewall** view allows managing firewall settings of the Domain, Private and Public profiles. In this view, you can configure the settings to be applied when installing and/or uninstalling a deployment package.

For detailed information on changes to the Windows Firewall settings that can be defined in a project, refer to the Windows Firewall Modifications section of this document.

Copyright © 2001-2023 EMCO. All rights reserved.

## Scheduled Tasks View

The **Scheduled Tasks** view is displayed within the main program area when the **Scheduled Tasks** node of any project is selected in the **Projects** view. In the **Scheduled Tasks** view you can configure Windows scheduled tasks to be created or deleted on package installation or uninstallation Pic 1.




**Pic 1. The Scheduled Tasks view**

MSI Package Builder allows you to create and delete Windows scheduled tasks. When monitoring existing installations, the required changes are created automatically and added to the corresponding project representing the installation process. It is also possible to create the changes manually.

Configuring scheduled tasks during deployment is not supported by App-V and MSIX/AppX packages, so, when generated, those changes are included into the MSI deployment packages only.

Copyright © 2001-2023 EMCO. All rights reserved.

The icon next to every item represented in the **Scheduled Tasks** view is used to describe of the item's type and state. The state icons are provided to help you understand what is currently happening in the program. You can always see if the item is being processed now by some operation or not, and if there are any processing problems.

Below is the list of type icons used:

 - a scheduled task;

 - a scheduled task containing errors that should be resolved before creating a deployment package.

The following overlays are used to represent the operation to be performed with each scheduled task:

 - a scheduled task should be created;

 - a scheduled task should be modified;

 - a scheduled task should be deleted;

 - a scheduled task is excluded from the build.

The actions for adding, editing and deleting the changes to Windows scheduled tasks as well as copying and moving those changes between projects are available in the **Scheduled Tasks** view pop-up menu and on the contextual **Scheduled Tasks** Ribbon page.

## Functions Overview

| | |
|---|---|
| **Changes Management** | From the **Scheduled Tasks** view, you can create, edit and delete the modifications to be performed by a deployment package to the Windows scheduled tasks configuration. The **New Scheduled Task** item from the **Scheduled Tasks** view pop-up menu and the **Scheduled Task** button from the **New** group on the contextual **Scheduled Tasks** Ribbon page can be used to create a new modification to Windows scheduled tasks. It is possible to create a scheduled task together with the package deployment process or control any scheduled task when installing and/or uninstalling the deployment package. To change any scheduled task modification, you can use the **Edit** item from the pop-up menu or the **Edit** button from the **Management** group on the contextual **Scheduled Tasks** Ribbon page, and to delete the modification, use the **Delete** items. |
| **Copy/Move** | You can easily copy and/or move the modifications to be performed by a deployment package to the Windows Firewall configuration from the Windows Firewall view. You can use the drag/drop and copy/paste techniques as well as the **Cut**, **Copy** and **Paste** menu items for this purpose. It is also possible to use the **Copy To** and **Move To** items available both in the pop-up menu and on the **Windows Firewall** contextual Ribbon page to perform copy/move immediately choosing a target project in a dialog. |
| **Import** | If you have a Windows schedule task configuration as an XML file, you can import this configuration into the project. The import action is available in the **Scheduled Tasks** view. To import a task select the **Import Task** option in the context menu or use the **Scheduled Task** button from the **Import** group on the contextual **Scheduled Tasks** Ribbon page. |

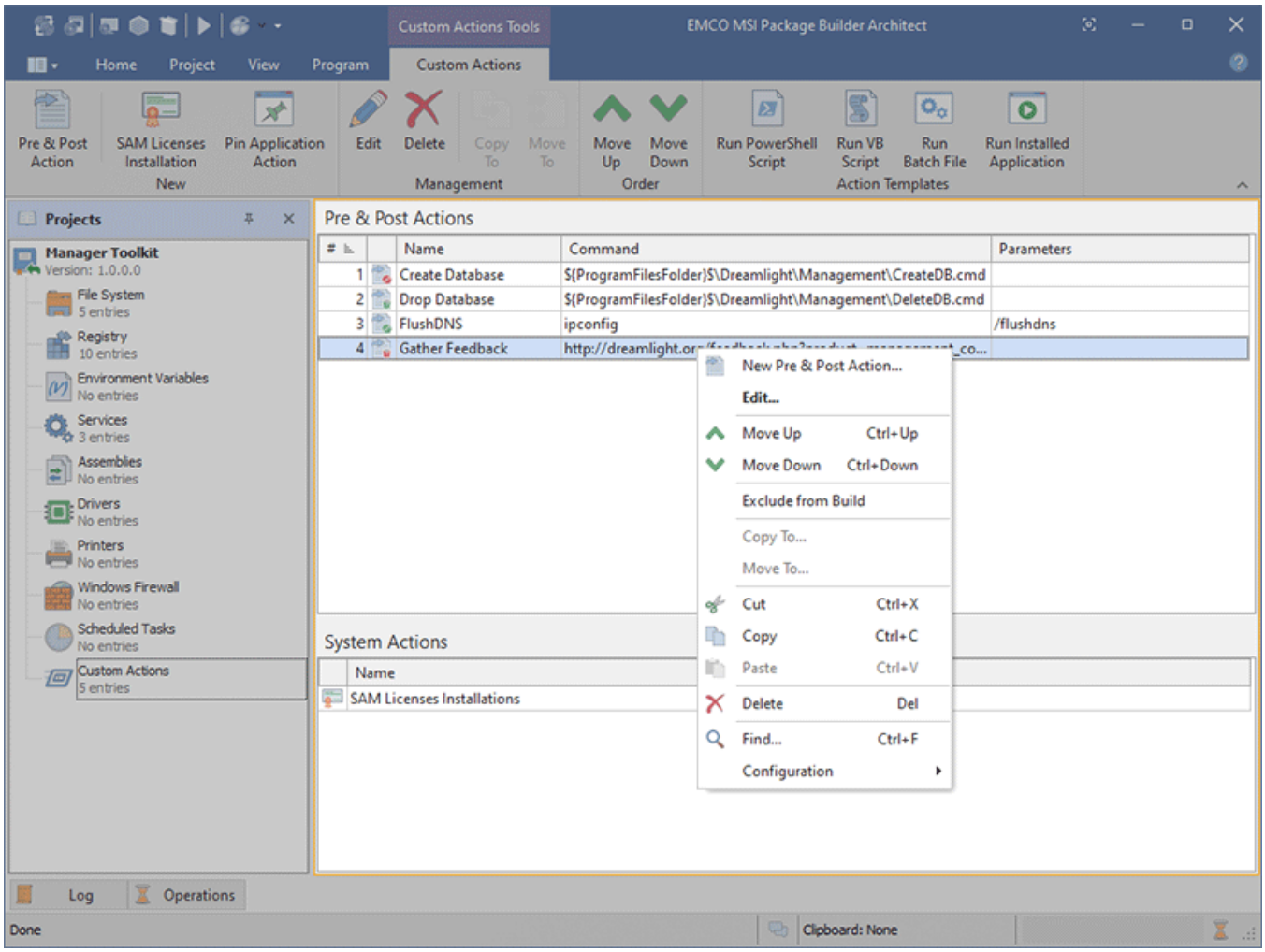
*Copyright © 2001-2023 EMCO. All rights reserved.*

| | |
|---|---|
| **Search** | Within the **Scheduled Tasks** view, you can execute a search for specific changes to the Windows scheduled tasks configuration performed by a deployment package using the Find item from the pop-up menu. |

For detailed information on the changes to Windows scheduled tasks that can be defined in a project, refer to the **Scheduled Tasks Deployment** section of this document.

## Custom Actions View

The **Custom Actions** view is displayed within the main program area when the **Custom Actions** node of any project is selected in the **Projects** view. In the Custom Actions view you can configure the actions to be executed before/after package installation or uninstallation. The view displays pre & post install/uninstall actions and system actions Pic 1.




**Pic 1. The Custom Actions view**

The pre & post install/uninstall actions are displayed in form of a table, where each row represents a single action. The icon next to every action is used to describe of the action's type and state. The state icons are provided to help you understand what is currently happening in the program. You can always see if the item is being processed now by some operation or not. Below is the list of icons used:

- an action to be executed before a deployment package installation;
- an action to be executed after a deployment package installation;

Copyright © 2001-2023 EMCO. All rights reserved.

 - an action to be executed before a deployment package uninstallation;

 - an action to be executed after a deployment package uninstallation;

 - a SAM licenses installation;

 - an action to pin an application to the Task Bar and/or Start Menu;

 - an action to unpin an application from the Task Bar and/or Start Menu;

 - an action is excluded from the build.

The system actions are also displayed in form of a table. With the help the system actions, you can pin applications to or unpin them from the Start Menu and the Task Bar. Another available system action is the SAM License Installation action. It allows you to install Software Assets Management (SAM) licenses to the Software Licensing Service (SLS) when deploying MSI packages.



App-V and MSIX/AppX packages virtualization techniques do not support custom actions, thus they can only be included into the MSI package output.

The actions for creating new actions, editing and deleting existing ones and copying/moving actions to another project are available in the **Custom Actions** view pop-up menu and on the contextual **Custom Actions** Ribbon page.

## Functions Overview

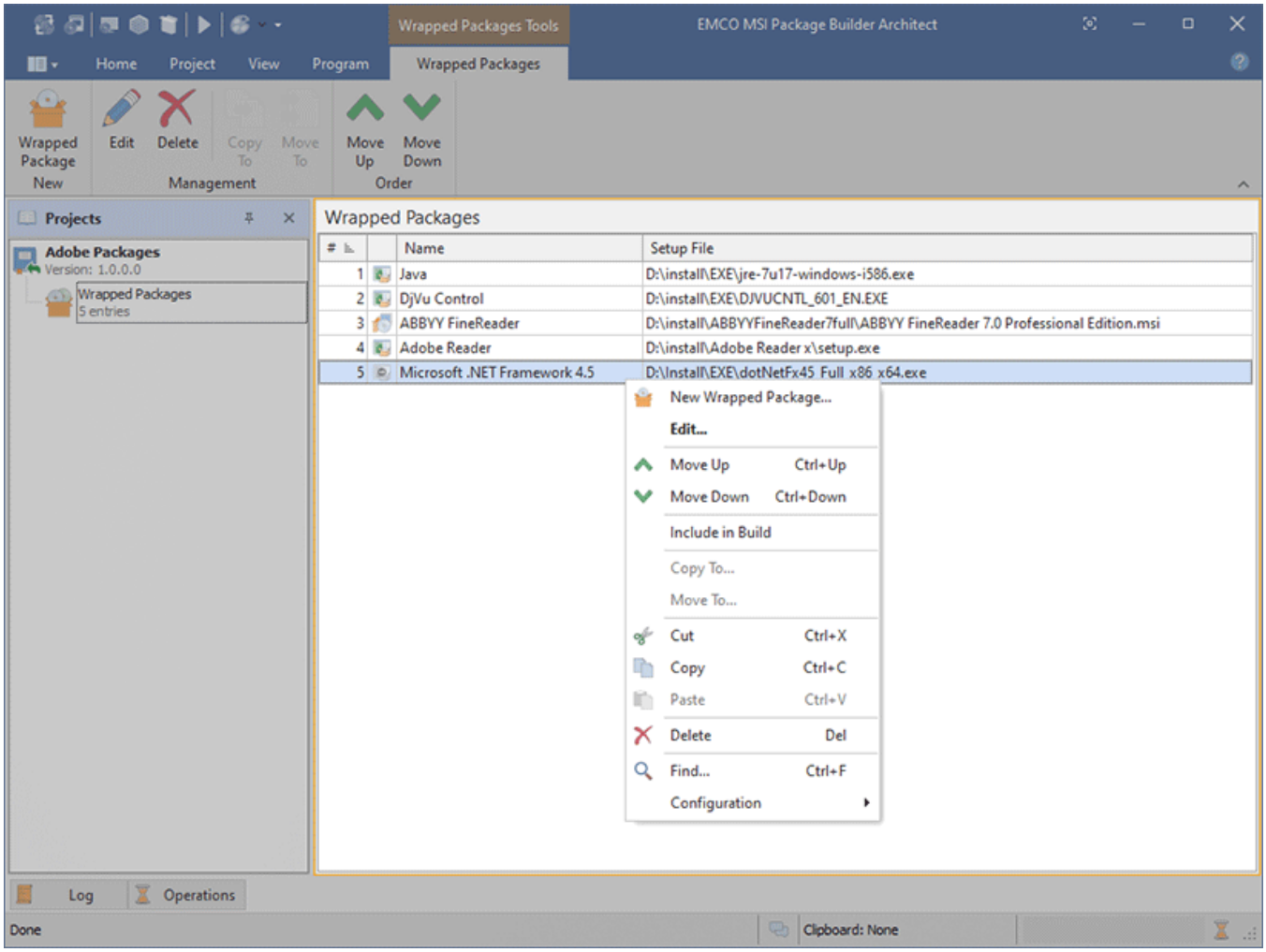
| | |
|---|---|
| **Actions Management** | From the **Custom Actions** view, you can create, edit and delete the actions to be performed before and after a generated deployment package deployment. To create a new action, you can either choose the **New Action** item from the pop-up menu, or press the **Action** button from the **New** group on the contextual **Custom Actions** Ribbon page and on the **Project** Ribbon page. To create custom actions of a specific type you can use actions available in the **Action Templates** group on the contextual Ribbon page. The **Edit** item from the pop-up menu, as well as the **Edit** button from the **Management** group on the contextual **Custom Actions** Ribbon page can be used to change the selected action, and to delete any action you can use the **Delete** items. |
| **Copy/Move** | You can easily copy and/or move the actions to be performed before and after a generated deployment package deployment from the **Custom Actions** view to another project. You can use the drag/drop and copy/paste techniques as well as the **Cut**, **Copy** and **Paste** menu items to reach the goal. It is also possible to use the **Copy To** and **Move To** items available both in the pop-up menu and on the contextual **Custom Actions** Ribbon page to perform copy/move immediately choosing a target project in a dialog. |
| **Search** | Within the **Custom Actions** view, you can execute a search for specific action using the Find item from the pop-up menu. |

For detailed information on the custom actions management process, refer to the Using Custom Actions section of this document.

Copyright © 2001-2023 EMCO. All rights reserved.

## Wrapped Packages View

The **Wrapped Packages** view is displayed within the main program area when the **Wrapped Packages** node of any project of the corresponding type is selected in the **Projects** view. This view is used to configure a set of packages that are deployed together with the generated MSI package Pic 1.




**Pic 1. The Wrapped Packages view**

The wrapped packages are displayed in form of a table, where each row represents a single package. The icon next to every package is used to describe of the package's type and state. The state icons are provided to help you understand what is currently happening in the program. You can always see if the item is being processed now by some operation or not and if there are any problems. Below is the list of icons used:

- an executable package;
- an executable package containing files that cannot be accessed;
- a Windows Installer package;
- a Windows Installer package containing files that cannot be accessed;
- a package is excluded from the build.

Copyright © 2001-2023 EMCO. All rights reserved.

The actions for adding wrapped packages, editing and deleting existing ones, changing the deployment order and others are available in the **Wrapped Packages** view pop-up menu and on the contextual **Wrapped Packages** Ribbon page.

## Functions Overview

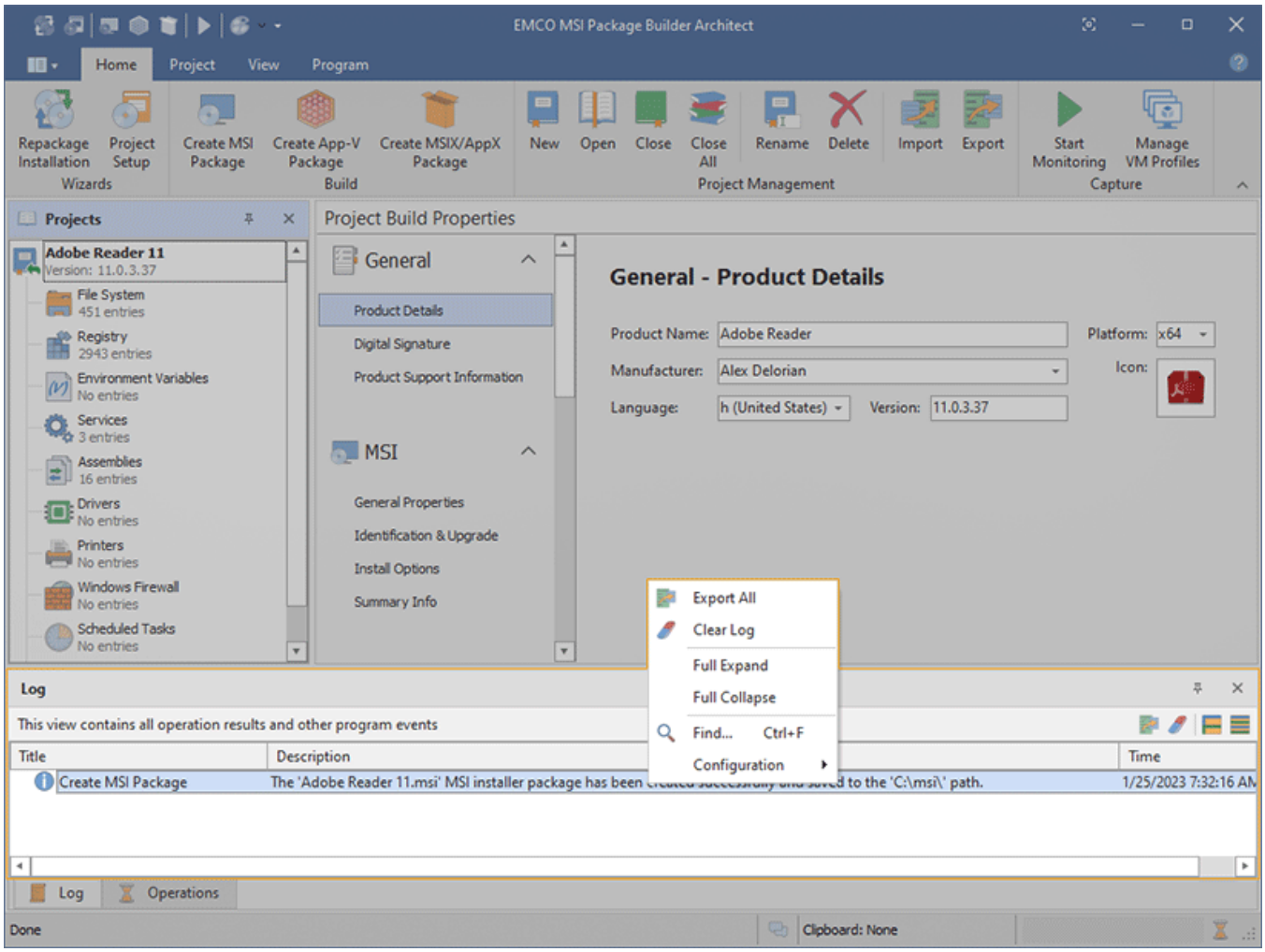
| | |
|---|---|
| **Packages Management** | From the **Wrapped Packages** view, you can create, edit and delete the installation packages to be deployed together with a generated MSI package. To create a new package, you can either choose the **New Wrapped Package** item from the pop-up menu, or press the **Wrapped Package** button from the **New** group on the contextual **Wrapped Packages** Ribbon page and on the **Project** Ribbon page. The **Edit** item from the pop-up menu, as well as the **Edit** button from the **Management** group on the contextual **Wrapped Packages** Ribbon page can be used to change the selected wrapped package, and to delete any package, you can use the **Delete** items. To change the packages deployment order, you can use the **Move Up** and **Move Down** items available in the pop-up menu and in the **Order** group from the contextual **Wrapped Packages** Ribbon page. |
| **Copy/Move** | You can easily copy and/or move the packages to be deployed together with a generated MSI package from the **Wrapped Packages** view to another project. You can use the drag/drop and copy/paste techniques as well as the **Cut**, **Copy** and **Paste** menu items to reach the goal. It is also possible to use the **Copy To** and **Move To** items available both in the pop-up menu and on the contextual **Wrapped Packages** Ribbon page to perform copy/move immediately choosing a target project in a dialog. |
| **Search** | Within the **Wrapped Packages** view, you can execute a search for specific installation package using the **Find** item from the pop-up menu. |

For detailed information on the wrapped packages and packages management process, refer to the Wrapping Existing Installations section of this document.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Log View

The **Log** view is designed to display information on the events taking place during the program execution. The larger part of this information consists of events generated by the operations and the operations results.




**Pic 1. The Log view**

The **Log** view is located by default at the bottom of the MSI Package Builder main window and displays the log in form of a tree Pic 1. The description for any logged event is by default wrapped, so that you can easily read it. If you would like to have more events visible at the same time, you can configure the **Log** view to display only one line per event by disabling the **Wrap Text** option from the **Configuration** menu.

Every event in the **Log** is assigned a severity level represented by a certain icon. The icon allows you to see if the operation has finished successfully without reading the description. The following icons are available:

- the blue icon with an 'i' character means that everything is OK;
- the brown circle icon with a cross-cut line is used to identify that the task was canceled by user or due to shutdown of the underlying system;
- the yellow icon with an exclamation mark is the warning sign: it tells you that some errors have occurred, but they are not critical. In such a case, there is no guarantee that the operation has actually succeeded;

Copyright © 2001-2023 EMCO. All rights reserved.

- the red icon with a white cross is the error sign: it means that the operation execution has failed.

Analyzing the **Log** can help you a lot in your everyday work with MSI Package Builder, because this log contains all the information on the executed operations and provides you with troubleshooting recommendations in case any problems are detected.

## Functions Overview

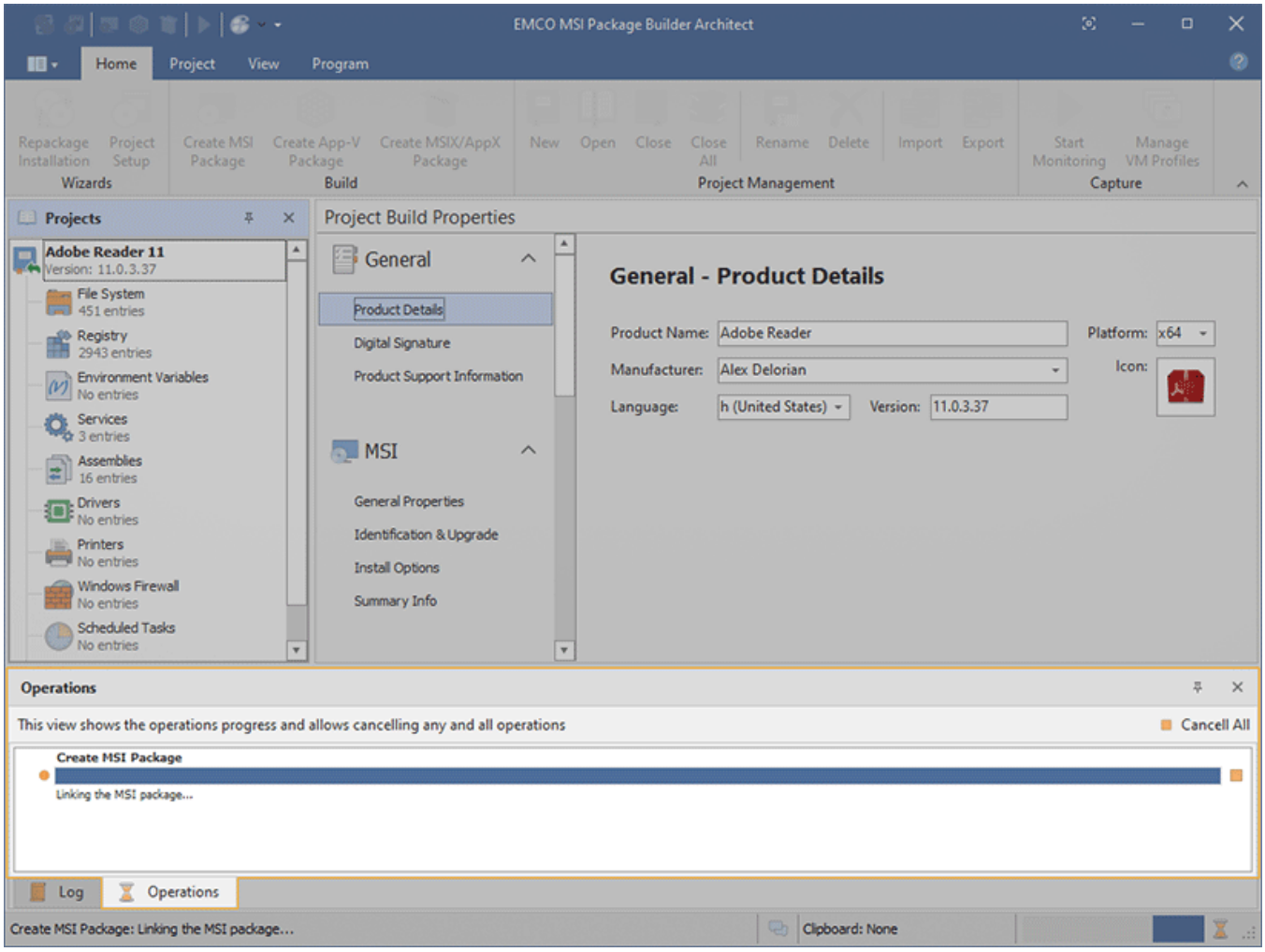
| | |
|---|---|
|  | **Export All**<br>The **Export All** button should be used to export the log to the CSV file. |
|  | **Clear**<br>The **Clear** button should be used to remove all the logged events from the program. |
|  | **Full Expand**<br>The **Full Expand** button from the **Log** view toolbar should be used to expand all nodes in the table of logged events. |
|  | **Full Collapse**<br>The **Full Collapse** button from the **Log** view toolbar should be used to collapse all nodes in the table of logged events. |

The options of clearing the log, expanding nodes in the tree of logged events and collapsing them are also available from the pop-up menu of the **Log** tree Pic 1. The layout of the **Log** view, including the visible columns, the column widths, the sorting settings and the text wrapping, is saved between sessions.

Copyright © 2001-2023 EMCO. All rights reserved.

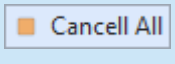
## Operations View

The **Operations** view Pic 1 shows the detailed progress of each operation being performed at the moment and allows canceling a particular operation or all running operations. By default, it is located at the bottom of the MSI Package Builder main window.




**Pic 1. The Operations view**

Progress information for every operation is shown in the pane with the progress bar, the operation information text and the **Cancel** button. The **Cancel** button is used to cancel individual running operations, whereas if the grouping operation is canceled, all the sub-operations are also canceled.




**Cancel All**

The **Cancel All** button from the **Operations** view toolbar can be used to cancel all the operations running in the application.

You can cancel all the running operations by clicking the **Cancel All** button on the toolbar of the **Operations** view.

Copyright © 2001-2023 EMCO. All rights reserved.

## Graphical User Interface features

EMCO Software provides you with a modern and intuitive graphical user interface, because we appreciate the users of our products and would like them to feel glad that they have EMCO programs installed on their PCs. Lots of resources were involved in creating this kind of an interface for you, and now we are proud we have done it. Custom DPI settings are fully supported, so that you can use EMCO programs on any display with any resolution you like. The *'Microsoft User Interface Guidelines on Layout, Icons and Sizing'* have been a powerful base for this work, and we are glad to tell you that they are fully complied with and supported. With the help of the skinning support and the Ribbon UI interface, every customer can configure the program UI to feel comfortable during each working day. EMCO also provides you with the **High Contrast** skin along with the bonus skins pack, which is an accessibility feature designed for people with vision impairment. The High Contrast color scheme can increase legibility for some users by heightening the screen contrast with alternative color combinations.

This chapter gives you a detailed description of how to fully enjoy the graphical user interface features, the skinning mechanism and the Ribbon bar features.

Copyright © 2001-2023 EMCO. All rights reserved.

## Skinning

MSI Package Builder provides you with a wide range of custom skins with unique look and feel, so that you can choose any skin you like most. If you are a fan of the Microsoft Office interface, you have no reason to complain either, since MSI Package Builder also gives you an option of choosing this type of skin. There are not only formal skins but also some informal ones.

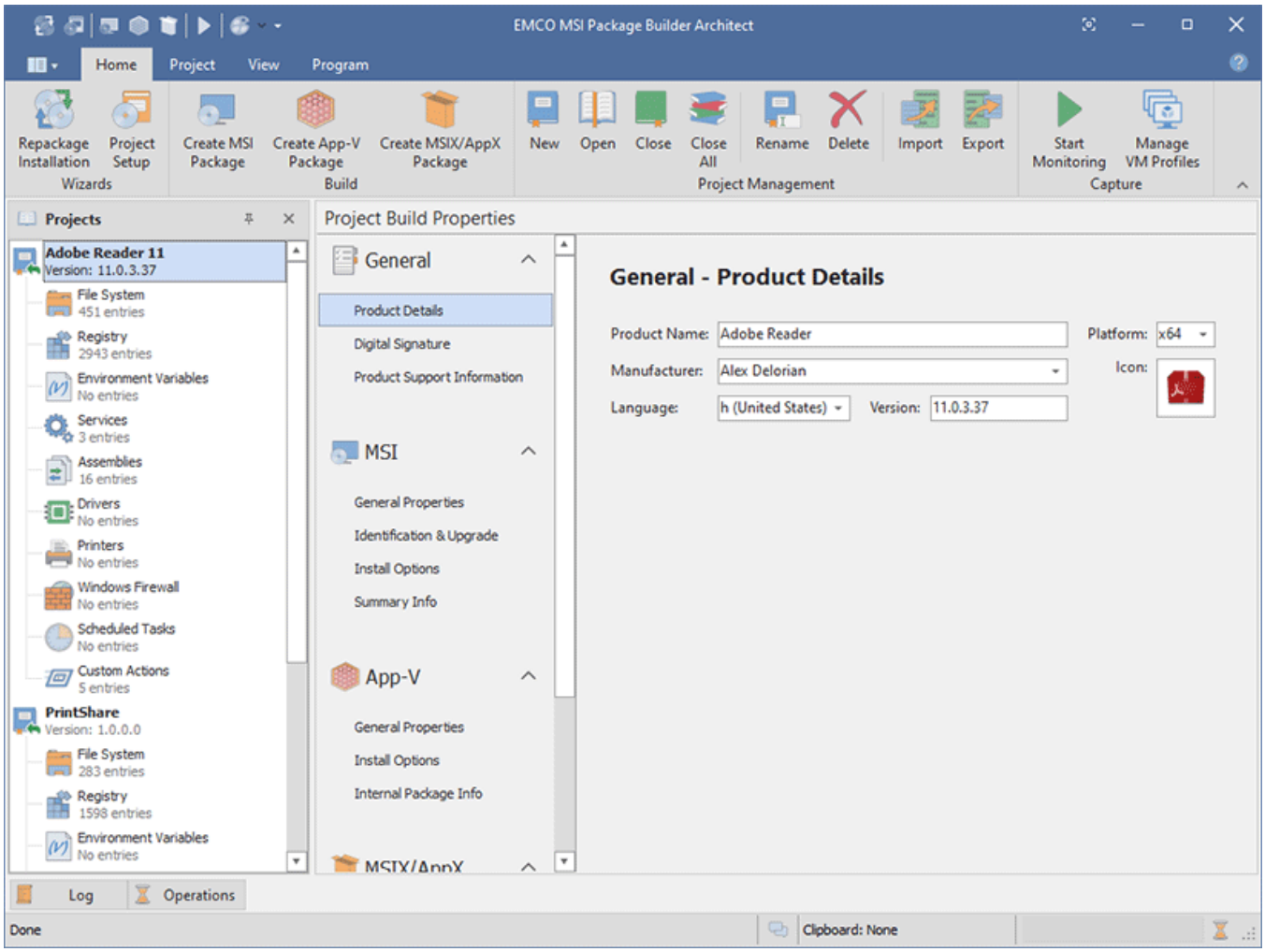
All the skins can be divided into four groups: Office Skins, Custom Design Skins, Bonus Skins and Skins for Fun. The following skins are available:

| | |
|---|---|
| Office Skins: | "Office 2019", "Office 2019 Black", "Office 2019 Dark", "Office 2019 White", "Office 2016", "Office 2016 Dark", "Office 2013", "Office 2013 Silver", "Office 2013 Black", "Office 2010 Blue", "Office 2010 Silver", "Office 2010 Black", "Office 2007 Blue", "Office 2007 Silver", "Office 2007 Black", "Office 2007 Green", "Office 2007 Pink". |
| Custom Design Skins: | "Basic", "Bezier", "Modern Style", "Dark Style", "Blue Vision", "Blue Vision 2013", "Light Vision 2013", "Dark Vision 2013", "High Contrast", "Metropolis", "Metropolis Dark", "Seven", "Seven Classic", "McSkin", "Blue", "Black", "Silver". |
| Bonus Skins: | "Lilian", "iMaginary", "Caramel", "Money Twins", "Sharp", "Sharp Plus", "Foggy", "Darkroom", "Dark Side", "Liquid Sky", "London Liquid Sky", "Stardust", "Coffee", "Blueprint", "Whiteprint". |
| Skins for Fun: | "Christmas", "Valentine", "Summer", "Springtime". |

*Copyright © 2001-2023 EMCO. All rights reserved.*

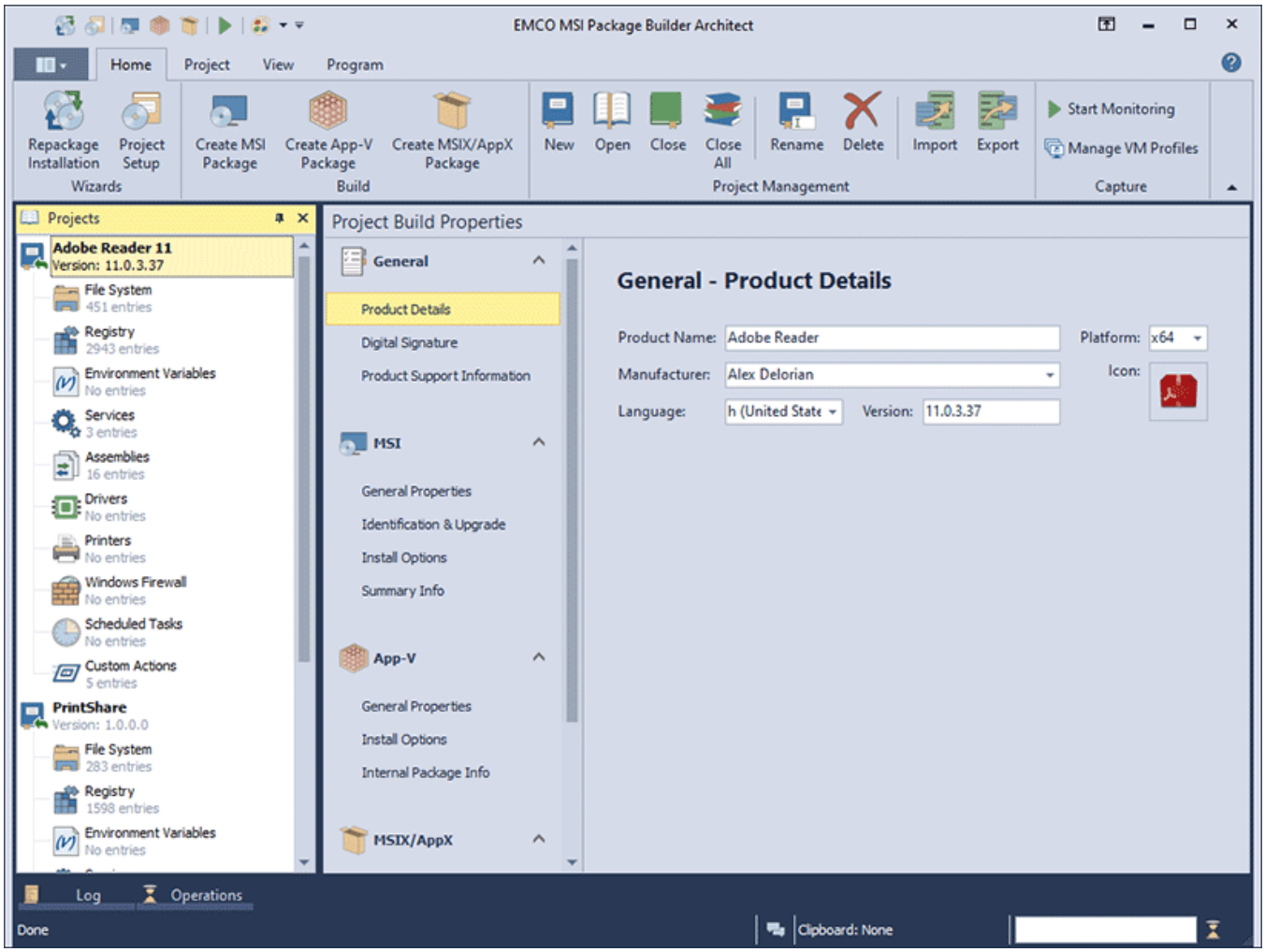
Let us take a brief look at some of the skins:

## Default skin look and feel example



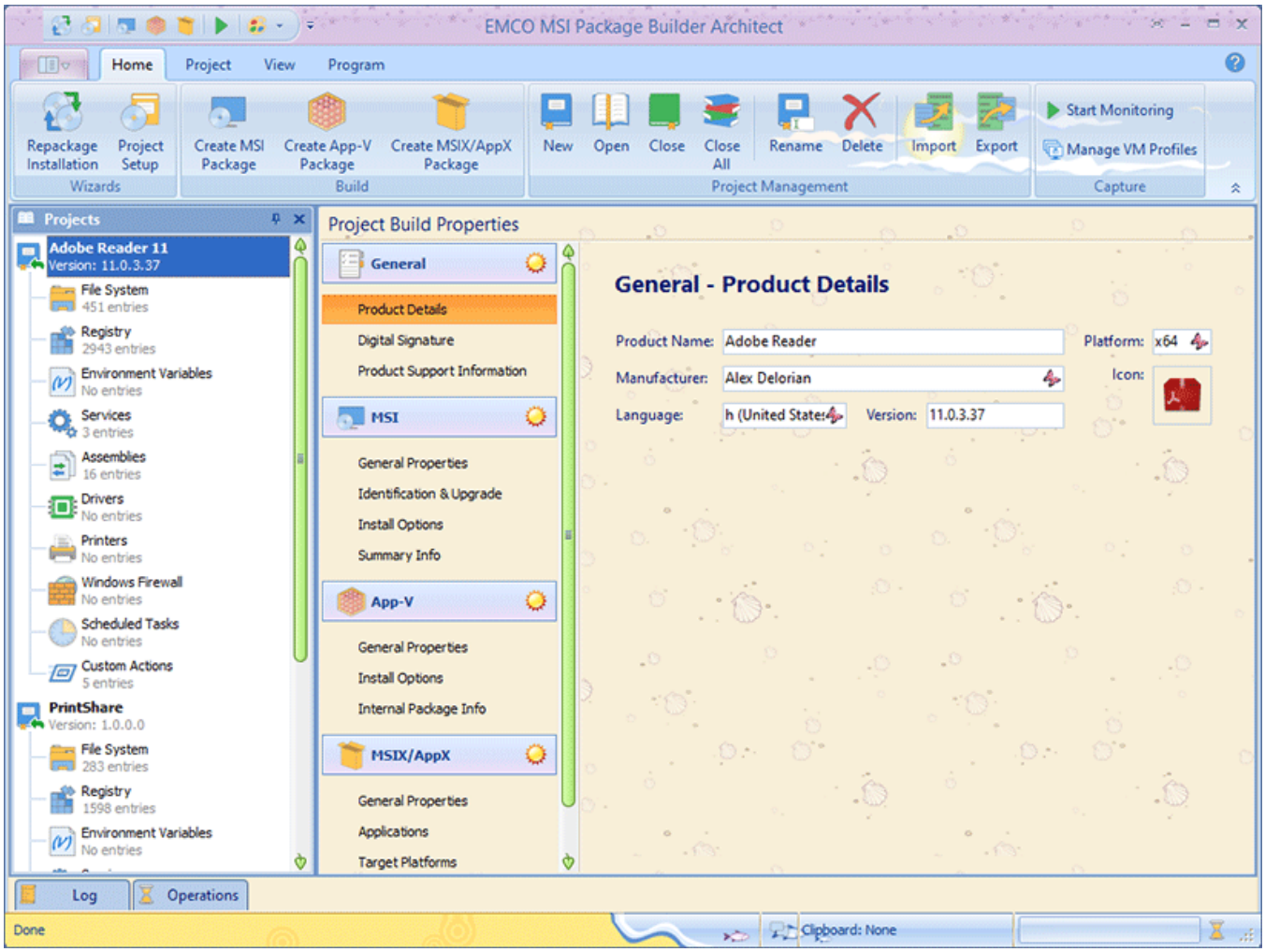

Copyright © 2001-2023 EMCO. All rights reserved.

Custom Design Skin look and feel example



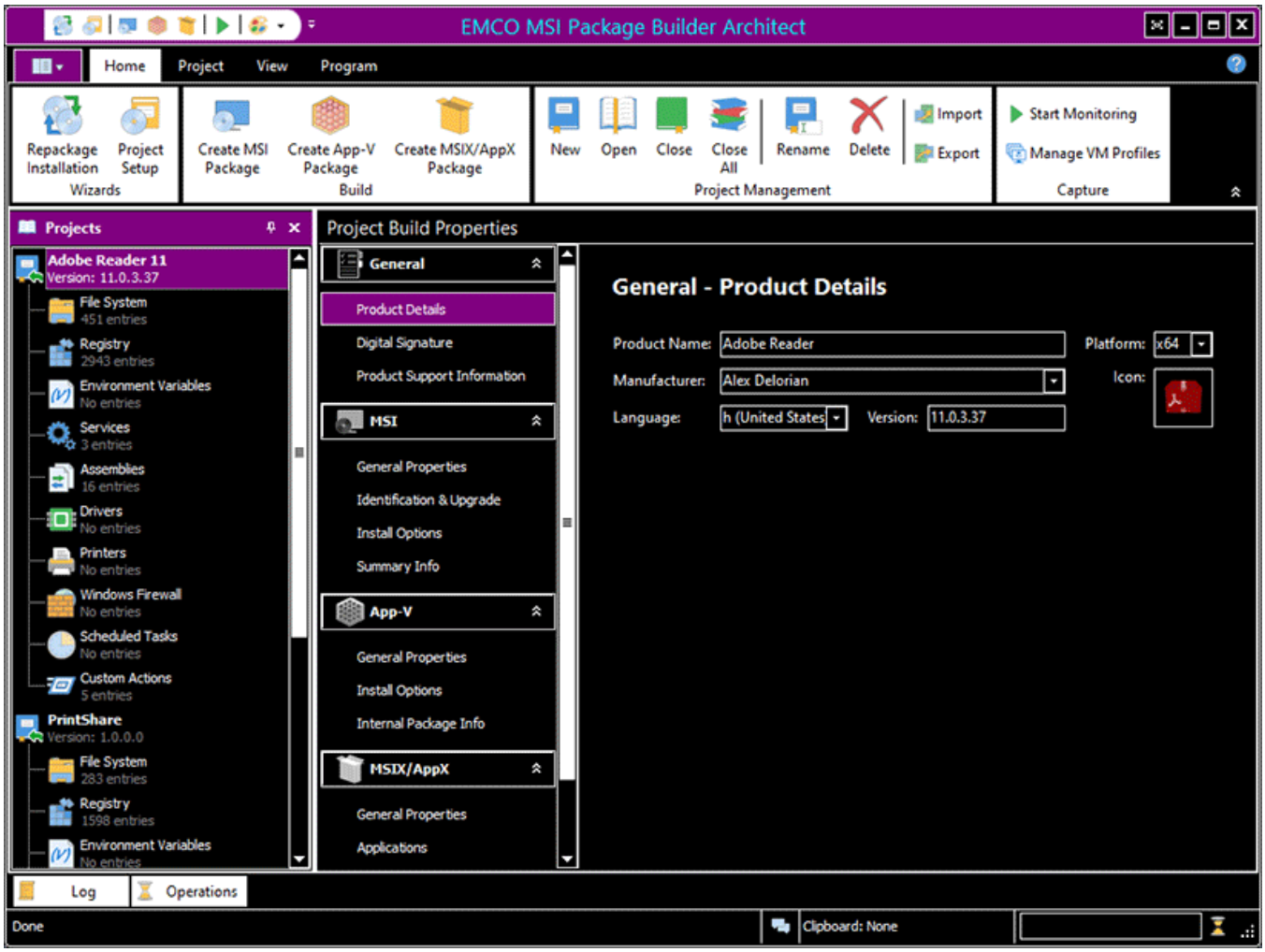

Copyright © 2001-2023 EMCO. All rights reserved.

Skins for Fun look and feel example




Copyright © 2001-2023 EMCO. All rights reserved.

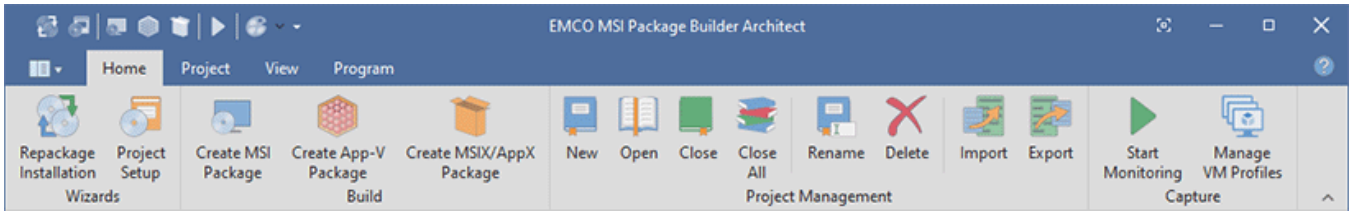
High Contrast Skin look and feel




Copyright © 2001-2023 EMCO. All rights reserved.

## Ribbon

Ribbon is a tool that presents commands organized into a set of tabs. The tabs on the Ribbon represent commands that are most relevant for each of the task areas in the program Pic 1. For example, in Office Word the tabs group commands by activities such as inserting objects like pictures and tables, doing page layout, working with preferences, doing mailings, and reviewing. The Home tab provides an easy access to the most frequently used commands. Office Excel has a similar set of tabs that make sense for spreadsheet work including tabs for working with formulas, managing data, and reviewing. Those tabs simplify access to the program features, because they organize the commands in a way that reflects the tasks people perform in those programs.



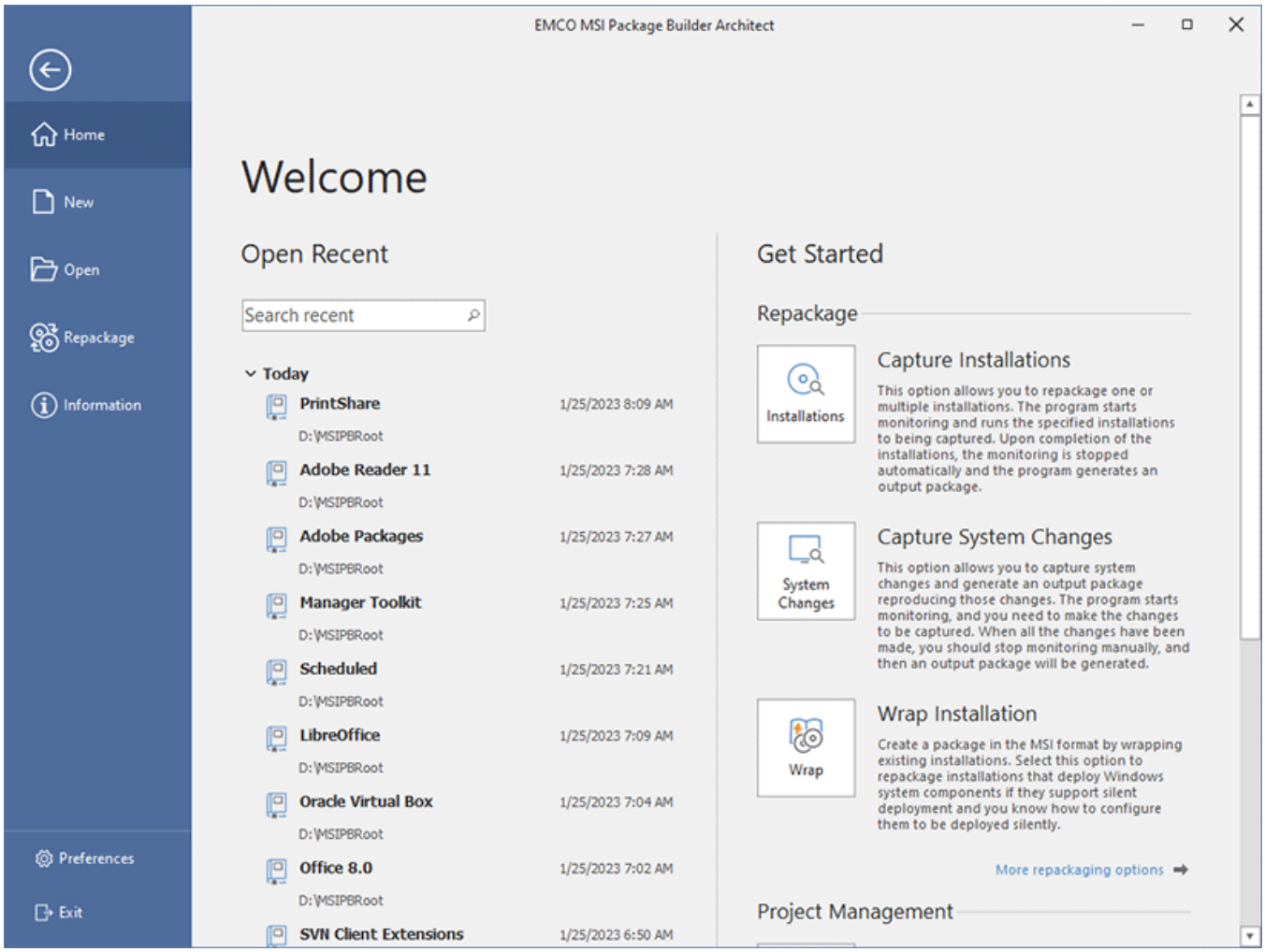

**Pic 1. The Ribbon bar**

We are delighted to let you know that we fully conform with *Microsoft® Ribbons Guidelines* and would like to introduce some Ribbon features to you. To learn more about Ribbon, the story of its development and its usability features, you may visit *'The Story of the Ribbon'* article from the MSDN blogs.

Copyright © 2001-2023 EMCO. All rights reserved.

## Application Menu

Application Menu in the program is represented as the **Backstage** view that can be used as a starting point and allows you to create a new installation package, open a recent project, configure the program settings or get information about the program. It is somewhat similar to the File menu to most of the programs and can be opened by clicking the **Application** button Pic 2 located in the top left corner of the main screen.



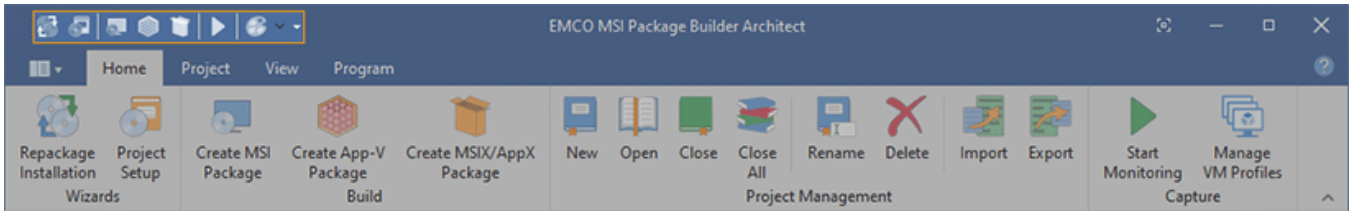

**Pic 2. Backstage view**

*Copyright © 2001-2023 EMCO. All rights reserved.*

On the left side of the **Backstage** view, you can see the buttons that allow you to create packages, open recent projects and get information about the program. On the right side, you can find the actions corresponding to the currently selected section of the **Backstage** view. The program settings can be configured by pressing the Preferences button located in the left bottom corner of the **Backstage** view.

## Quick Access Toolbar

The **Quick Access Toolbar** Pic 3 is an end-user customizable bar located near the **Application Menu** or below the Ribbon bar depending on the configuration. It can contain links to both Ribbon items and Ribbon groups.




**Pic 3. Quick Access Toolbar**

To add an action link to the quick access toolbar, right click this action and select **Add to Quick Access Toolbar** from the pop-up menu. The groups can be added in the same way, the only difference being that to add a group you should right click its caption.

## Representation and Navigation Features

The representation of the Ribbon bar can also be configured to make your work more comfortable. You can minimize Ribbon so that the tab's content is only shown when the tab is clicked on, thus extending the program workspace. Also, if it is not convenient for you to have the **Quick Access Toolbar** next to the **Application Menu**, you may place it below the Ribbon bar, so that it will look just like a simple toolbar. This configuration can be accessed from the pop-up menu of the Ribbon bar Pic 4.




**Pic 4. The Ribbon customization menu**

Navigation between the Ribbon tabs can be performed not only with a mouse click on a tab but also with the help of the mouse wheel. Just place the cursor over any tab and scroll the wheel – scrolling up will switch the tabs from right to left, and scrolling down will switch the tabs in the opposite direction.

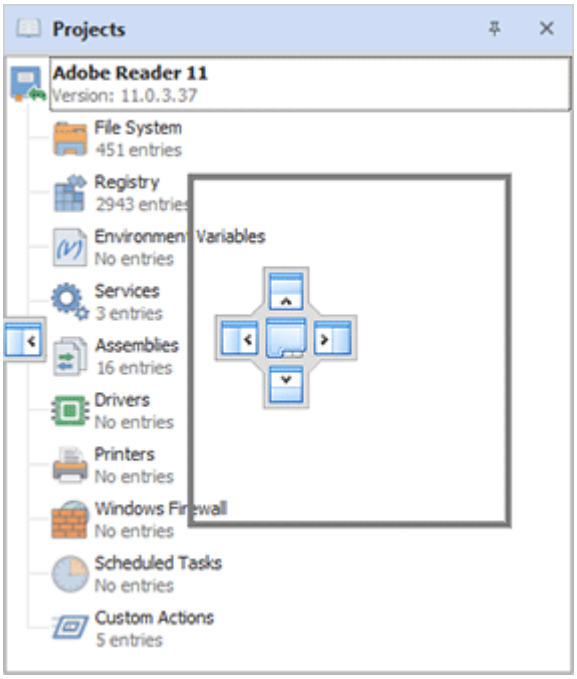
*Copyright © 2001-2023 EMCO. All rights reserved.*

## Main features of UI Elements

The graphics shell used for MSI Package Builder is aimed at providing a high level of usability to everyone. This topic covers main features of the graphical elements used in this program, and here you can find what puts EMCO GUI a step ahead of the others.

## Docking

The MSI Package Builder user interface is built using the ultimate docking technology which provides for the maximum use of the program working area. It allows docking the windows that are used less often than the main one to the sides, auto hide them or even close and then open again when required. The dock panels can be docked both to the main window and to each other, thus enabling you to build such a subsidiary window layout that makes you feel comfortable while working with MSI Package Builder Pic 1.




**Pic 1. Docking preview**

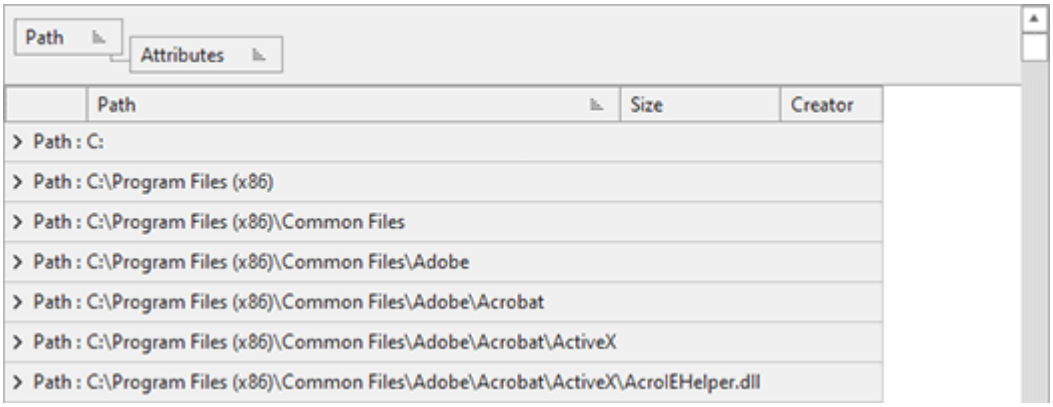
To change the position of any dock window, you should click its header and move the mouse pointer while holding the left mouse button down. Hint windows are shown to help you understand where you can drop the window dragged. When you are dragging it over another dock window, it is possible to dock both windows to each other or display them in different tabs of the same dock window. When a dock window is floating, you can expand it to full screen by either clicking the **Full Screen** button in the windows title bar or pressing **F11** on the keyboard. To exit the full screen mode, just press **F11**.

To enable the auto hide feature for a window attached to any side of a main window, click the pin button in the dock window header. Clicking the cross button results in closing of the dock window. Each view can also be closed and opened again using the checkboxes in the **Show** Ribbon group accessible from the **Application** page.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Grouping and Filtering Data

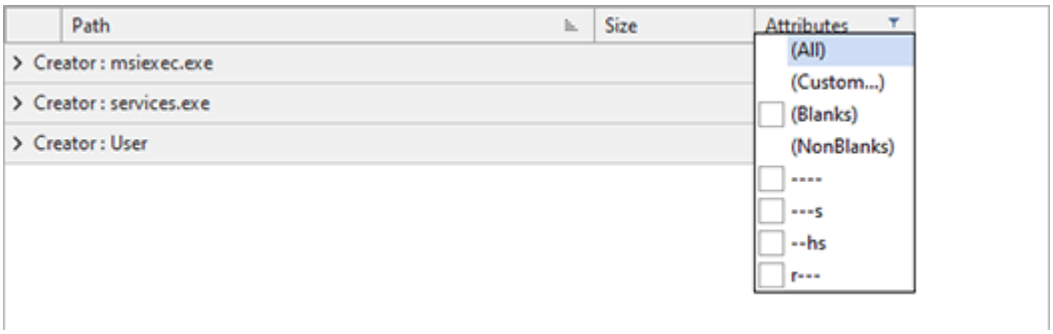
The MSI Package Builder user interface is designed so as to make its usage as flexible as possible. The tables available in every EMCO program provide you with an easy-to-use data filtration and grouping mechanism. To group data by one of the columns, you should drag its header to the grouping box displayed over the table or choose an appropriate item from the column header pop-up menu Pic 2.




**Pic 2. The grouping box of a table grouped by two columns**

To group or ungroup data by any column when **Group By Box** Pic 2 is not visible, you can have it displayed by selecting the **Show Group By Box** item from the pop-up menu of any column header.

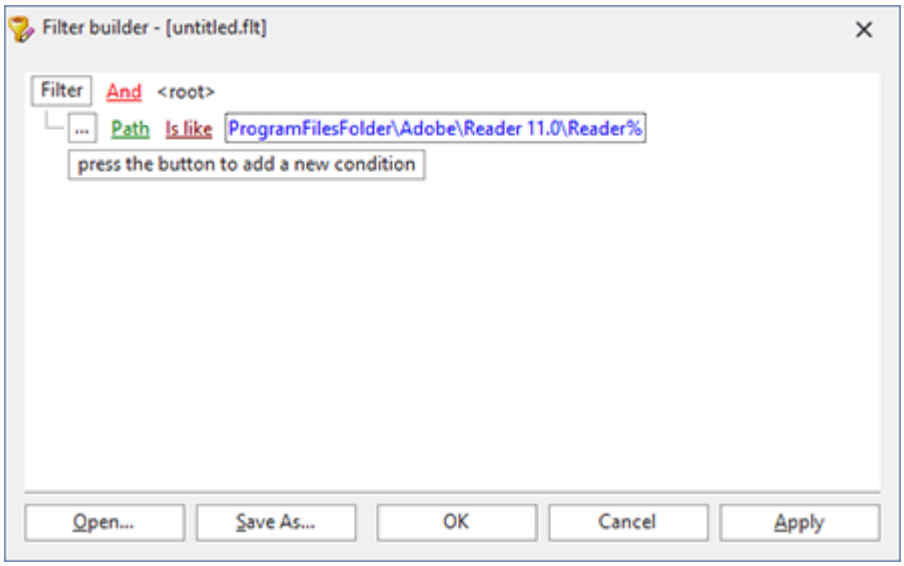
Data filtration can be performed in two ways: by using the quick filter or the filter editor. To use the quick filter feature, just click on the glyph in the right top corner of any column header. A drop-down list appears offering you to choose one of the predefined filters or select the custom one from the filtering dialog Pic 3.




**Pic 3. Accessing the quick filter abilities**

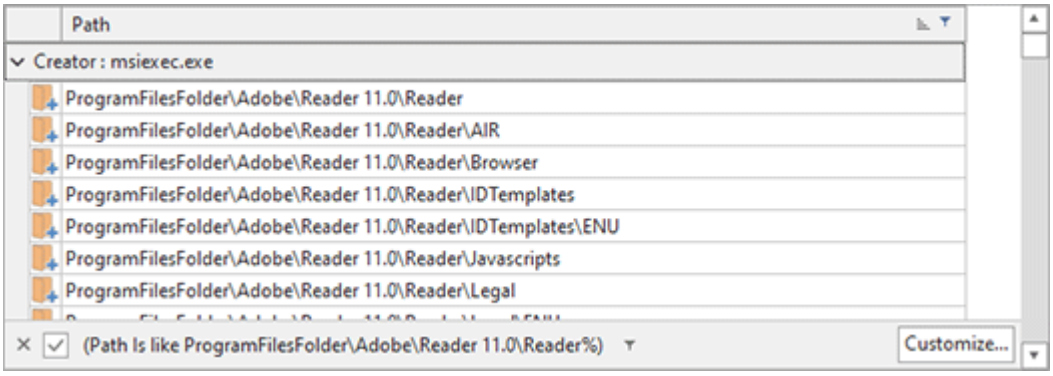
Copyright © 2001-2023 EMCO. All rights reserved.

The **Filter Editor** shipped with MSI Package Builder is easy to use and allows you to build your own complex filters quickly and easily Pic 4. To open the filter editor, choose the **Filter Editor** item from the column's pop-up menu.




**Pic 4. Using the filter editor**

You can enable and disable the currently applied filter condition using the checkbox displayed next to the filter condition in the bottom of the view, inside the filter info pane Pic 5.



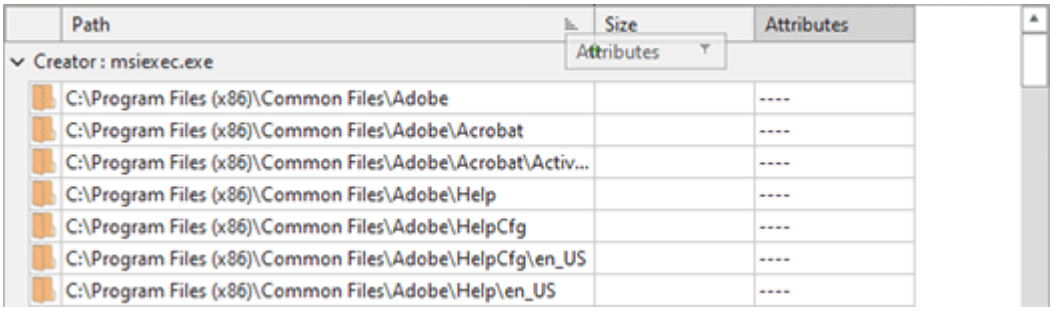

**Pic 5. The filter info pane**

To reset the currently applied filter use the button from the filter info pane, and to customize it use the **Customize** button from this pane.

Copyright © 2001-2023 EMCO. All rights reserved.

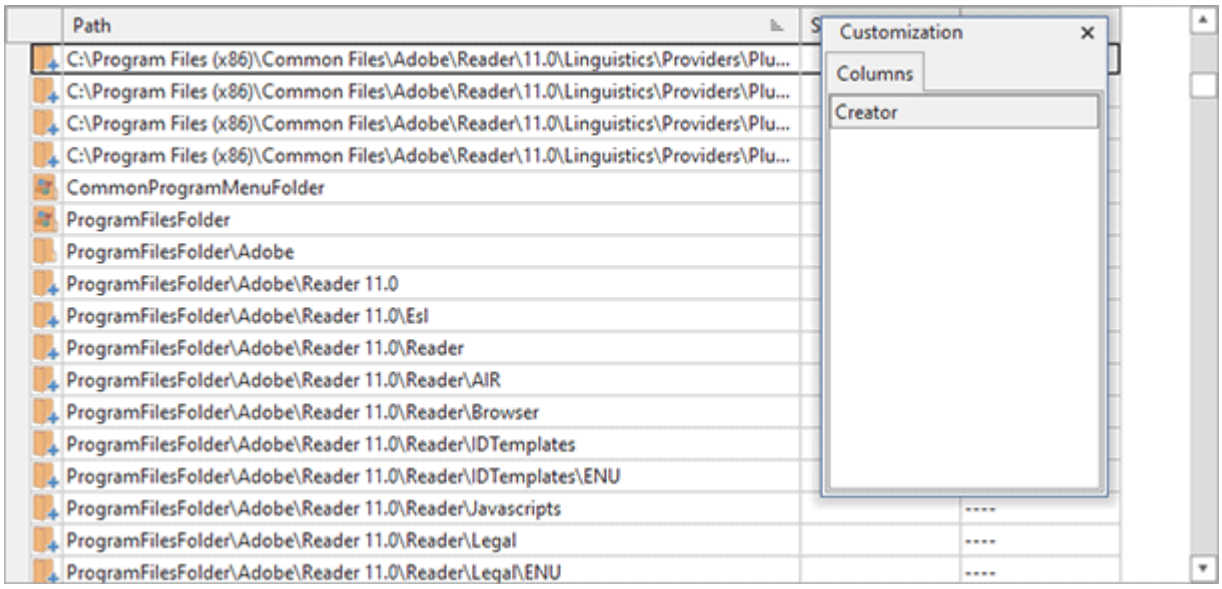
## Managing Columns in Trees and Tables

You can customize almost every table and tree in EMCO programs by moving and removing columns to make the control most informative for you. To move a column, drag it over the control's header and drop between other columns to its new position Pic 6.




**Pic 6. Moving a column**

To remove a column that is of no use for you, right click the control's header and select the **Remove This Column** item from the pop-up menu. Also, you can control columns availability using the column chooser Pic 7.




**Pic 7. Using the column chooser**

To show the column chooser, right click the control's header and select the **Column Chooser** menu item. After that, you can drag and drop columns from the header to the column chooser and backwards.

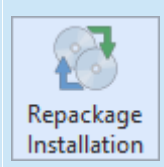
## Automatic Saving and Restoring of Windows Layout

One of the service functions of MSI Package Builder user interface is its ability to save and restore the windows layout. All the changeable parameters like the windows sizes and positions; the table columns order, sizes and positions; the grouping and filtering options; the dock windows configuration, etc. are saved between sessions. Thus, you do not need to configure the program's user interface layout every time you start this program.

Copyright © 2001-2023 EMCO. All rights reserved.

# Chapter 4: Installations Repackaging

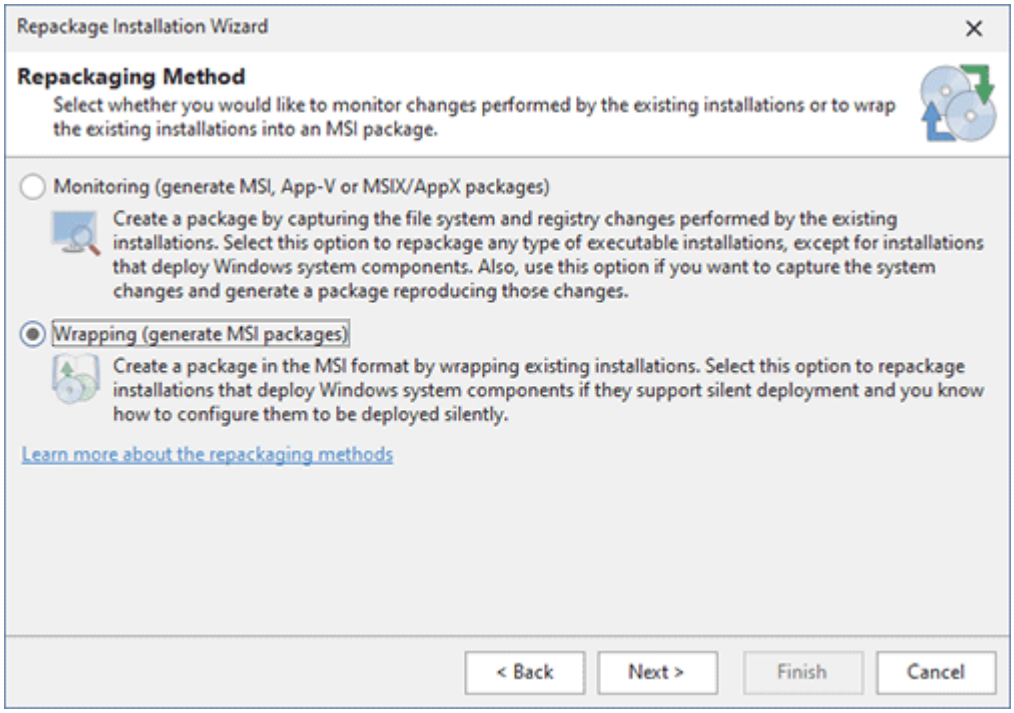
The main goal of MSI Package Builder is to deliver to you an easy-to-use installations repackaging tool. We have designed and developed two different repackaging models, which are monitoring operating system changes and wrapping existing installations. You can choose between smart repackaging, which allows you to easily create a deployment package based on existing installations by simply following the steps of the **Repackage Installation** wizard, and low-level repackaging, which allows you to control each aspect of the repackaging process.




**Repackage Installation**

The **Repackage Installation** button from the **Builder** group on the **Home** Ribbon page should be used to open the **Repackage Installation** wizard, that is designed to help you with the repackaging process.

The **Repackage Installation** wizard was designed to make the repackaging process as intuitive as it is possible. By default, it is displayed on the program startup, so you can repackage existing installation with no additional actions required. If you do not need it to be displayed on each startup, you can disable this option other on the wizard welcome page, or on the **User Interface** preference page.




**Pic 1. Choosing a repackaging mode**

Copyright © 2001-2023 EMCO. All rights reserved.

While going through the steps of the **Repackage Installation** wizard, you are choosing the repackaging mode you would like to use, configure a deployment package to be generated and a project to be created. The project containing the repackaging results is saved to the projects storage and can be used in the future to upgrade the installation package.

In this chapter we'll go through the available repackaging modes and take a closer look at each one, providing you with some tips and tricks on using the repackaging abilities shipped with MSI Package Builder.

## What's Inside

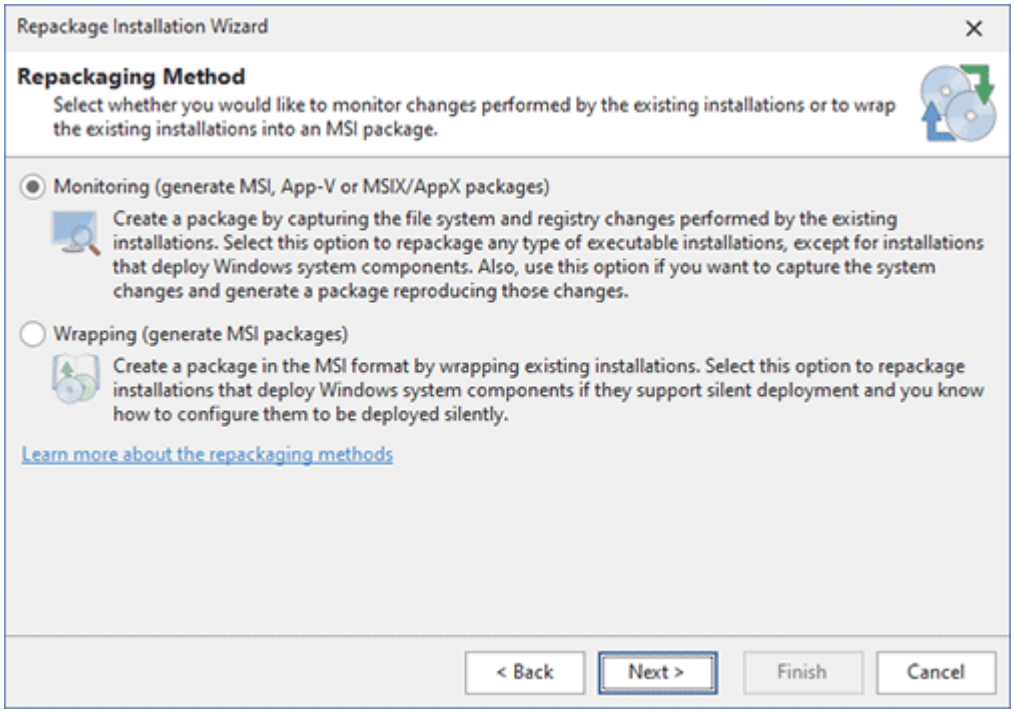
How to Choose a Repackaging Method
Capture Installations
Capture System Changes
Repackaging via Wrapping
Low-Level Repackaging
How should I configure the repackaged installations to support an upgrade?
Repackaging on Windows Sandbox
Repackaging on a Virtual Machine

Copyright © 2001-2023 EMCO. All rights reserved.

## How to Choose a Repackaging Method

MSI Package Builder is shipped with the different repackaging methods to be used for different situations. It is possible to use the **Repackage Installation** wizard to create an MSI package automatically with the help of the smart repackaging, or use the low-level repackaging to benefit from advanced repackaging features. As for smart repackaging, you can choose between installations monitoring, monitoring of changes made to the operating system and wrapping of existing installations into an MSI package. The low-level repackaging allows you to combine different repackaging techniques and to control each aspect of the repackaging process.

When you use the **Repackage Installation** wizard, you can select the monitoring or wrapping methods of installation repackaging Pic 1. Note that this step of the wizard is available in the Enterprise and Architect editions of the program, which include the installations wrapping option. The Professional edition of the program offers monitoring only, so the repackaging method selection is not available in this edition.




**Pic 1. Selecting the repackaging method**

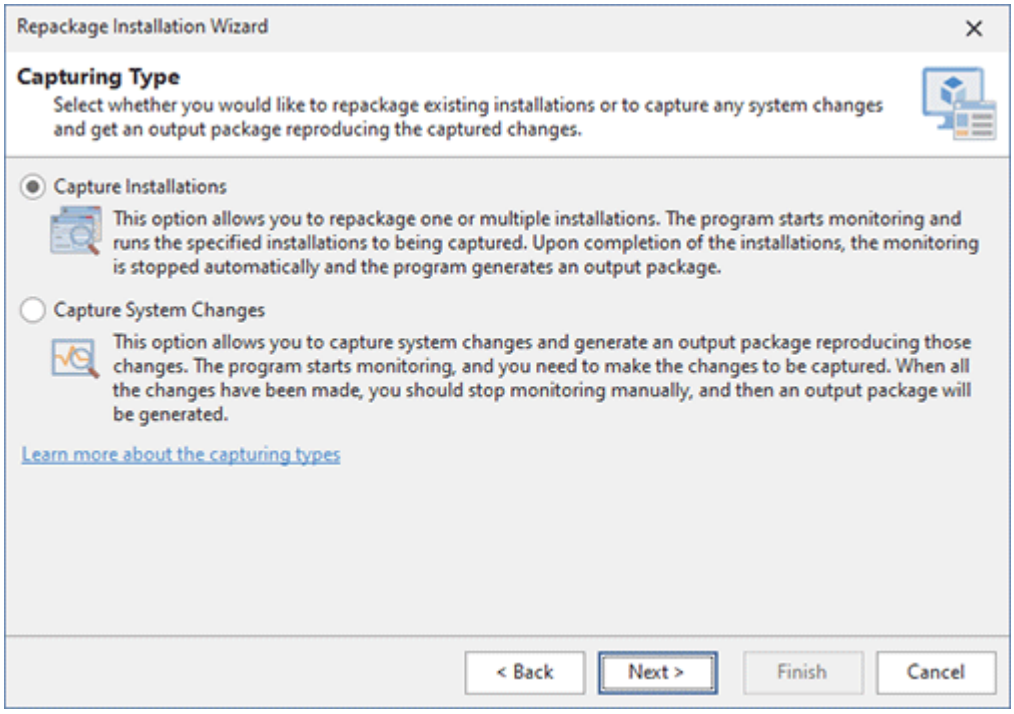
Copyright © 2001-2023 EMCO. All rights reserved.

## Installations Monitoring

Repackaging through monitoring of file system and registry changes is the easiest repackaging method, but it has a set of limitations. Using this method, the deployment package is created based on changes performed to the operating system by existing installations. You should choose this method when the repackaged installations do not support silent execution or you cannot find out the parameters to be used for silent execution. Use installations monitoring for creating App-V packages and MSIX/AppX packages as installations wrapping is not applicable for virtualization. This method can also be used when you are going to create an MSI package based on generic changes performed manually rather than those performed by existing installations.

The main prerequisite for repackaging via system changes capturing is that the monitoring should be performed in a clean environment. The main disadvantage of this mode is that the resulting deployment will not be fully cross-platform, thus the repackaging results will be valid only for the same operating system as used for monitoring.

The program supports two types of changes capturing. In the **Repackage Installation** wizard, you can select the option to capture installations or to capture system changes Pic 2 depending on your needs.




**Pic 2. Selecting the capturing type**

Copyright © 2001-2023 EMCO. All rights reserved.

**Capture Installations**

This option is designed to repackage an existing installation into a package of a different format. The program captures changes performed by the repackaged installation and uses those captured changes to generate a new package. You should use installations capturing if you need to repackage an existing installation with no additional pre/post-installation customization steps.

When the program captures an installation, it starts and stops monitoring system changes automatically. In this case, changes monitoring will stop only after the installation is fully completed, thus all the changes performed by the installation have been included into the resulting deployment package. To use this approach, an installation setup should be designed in such a manner that all the processes configuring the application and the operating system exit on installation completion. You can learn more about this type of monitoring in the Capture Installations chapter.

**Capture System Changes**

This option is designed to capture any file system and registry changes and use the captured results to generate a package. You can use this option to repackage installations that need additional pre/post-installation customization steps or to convert any changes into a package. For example, this mode allows you to make the required changes manually and generate a package reproducing those changes.

You can also use system changes capturing to repackage an installation that isn't designed to exit on installation completion. In such a case, you are responsible for stopping the monitoring process when all the required changes have been performed.

You can learn more about this type of monitoring in the Capture System Changes chapter.

## Installations Wrapping

The wrapping method of installations repackaging is recommended for experienced users for creating MSI packages. When wrapping is used, the program creates an MSI package that includes the repackaged installation file. You can use wrapping if the repackaged installation supports silent deployment and you know the command-line parameters required to deploy the installation silently.

Using wrapping is a little more complicated, but it allows you to avoid all the limitations of the monitoring technology. Installations wrapped into an MSI package will be distributed in the original state and thus deployed correctly to any operating system regardless the deployment process particularities. You can learn more about wrapping by reading the Repackaging via Wrapping chapter.

## Low-Level Repackaging

The low-level repackaging options allow you to reach the maximum flexibility in repackaging, but should be used by experienced users only. In this mode, you can change the monitored data and provide the required changes manually. This mode is the most complex one, because you are responsible for preparing the correct changes configuration for creating a deployment package. The complexity of the generic repackaging process requires that you to fully understand each aspect of every step to be performed for successful repackaging. You can learn more about this approach in the Low-Level Repackaging chapter.

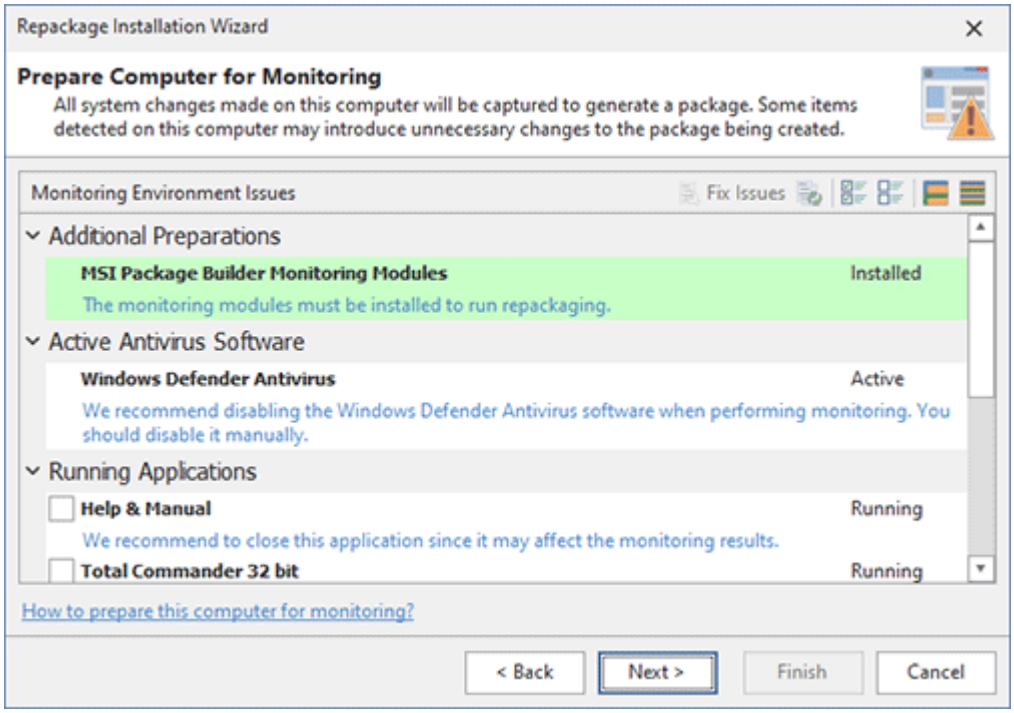
Now you have been introduced to the repackaging modes included into MSI Package Builder and should be able to choose an appropriate one during your everyday work.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Capture Installations

MSI Package Builder allows you to automatically monitor the changes performed by a single installation or a set of installations and create a deployment package based on these changes. When multiple installations are monitored, they are launched one by one in the specified order. To monitor installations, you should choose the **Capture Installations** option in the **Repackage Installation** wizard.

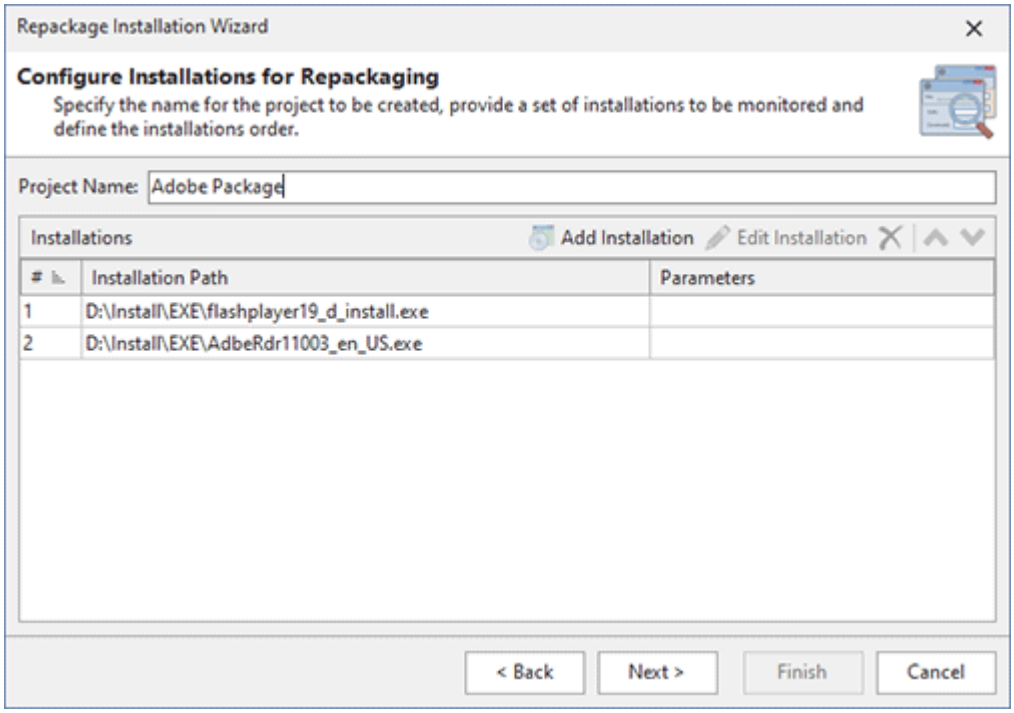
Let us take a look at the monitoring process configuration. First, you need to make sure that the monitoring environment is clean and you follow the repackaging best practices. The program can automatically check some requirements and show the list of monitoring environment issues Pic 1. You need to resolve those issues by following the displayed recommendations. For example, if any Windows applications are running, you can select them and press the **Fix Issues** button, so those applications will be closed.




**Pic 1. Monitoring environment issues**

Copyright © 2001-2023 EMCO. All rights reserved.

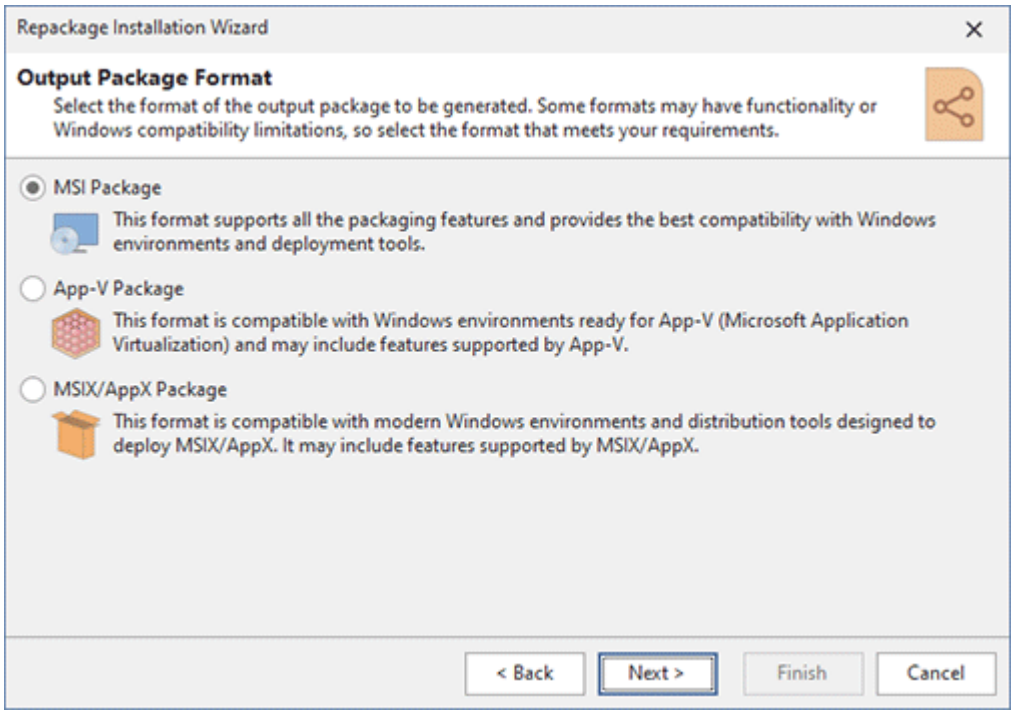
At the next step of the wizard, you should specify a name for the project to be created based on the monitored changes, provide a set of installations to be monitored and define the installation order Pic 2. You can define command-line parameters to be passed to the installer application for each installation while adding it.




**Pic 2. Providing installations to monitor**

It is possible to monitor Windows Installer packages and executable installation packages.

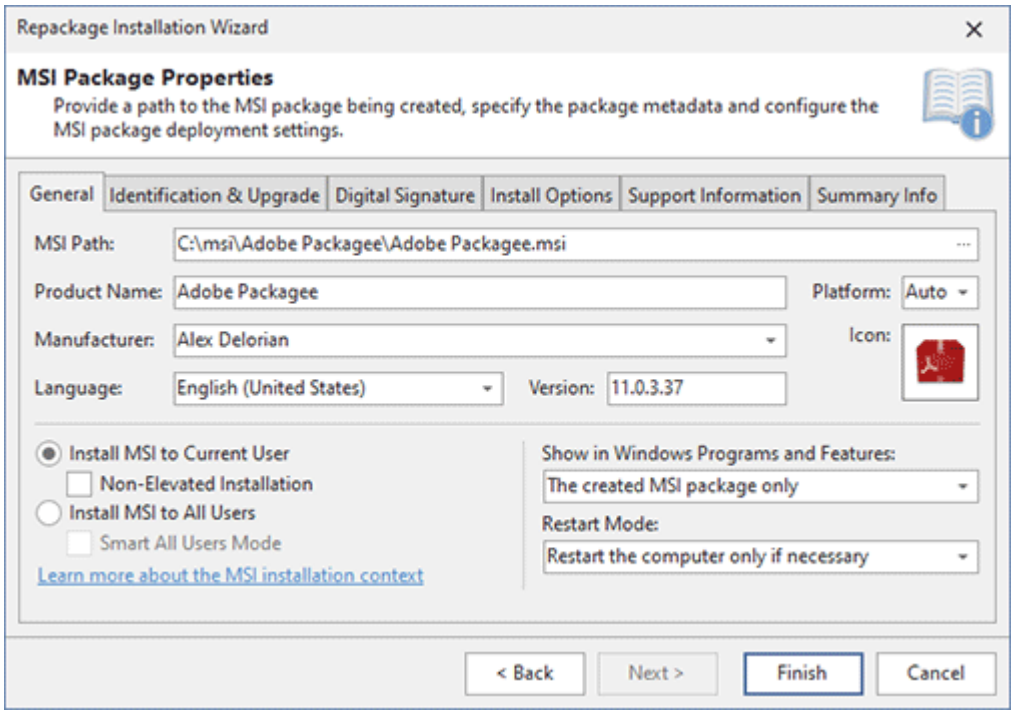
At the next step, you can choose if you would like to create an MSI package, an App-V package or an MSIX/AppX package based on the monitoring results.




**Pic 3. Choosing the package type**

Copyright © 2001-2023 EMCO. All rights reserved.

In case you have chosen to create an MSI package, at the next step you should provide a path the location to save the resulting MSI package and configure the package. Detailed information on configuring an MSI package is available in the Creating an MSI Package chapter.




**Pic 4. Configuring an MSI package**

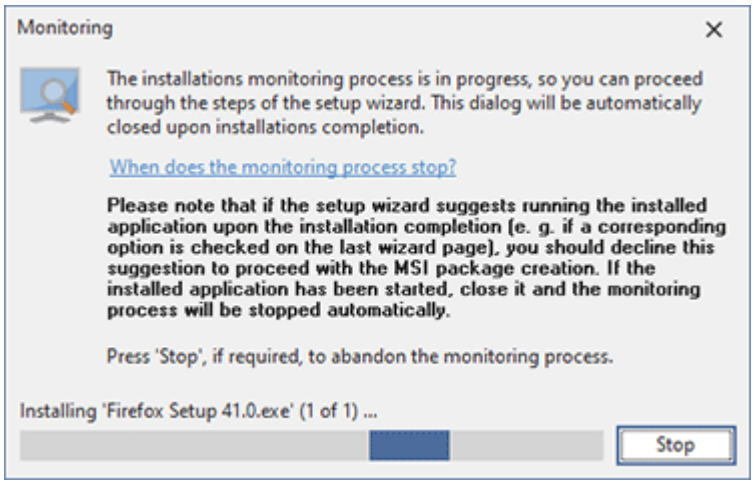
*Copyright © 2001-2023 EMCO. All rights reserved.*

If the output in form of an App-V package has been selected, you need to provide a path to the location to save the resulting App-V package and configure the package. Detailed information on configuring an App-V package is available in the Creating an App-V Package chapter. As for the output in form of an MSIX/AppX package, you need to provide a path to the location to save the package to and define the package configuration options, target platforms and digital signature settings. Refer to the Creating an MSIX/AppX Package chapter.

After you press **Finish**, the monitoring process starts. As soon as it is completed, a deployment package is created containing the monitored changes in the specified location.

## When does the monitoring process stop?

When a monitoring process is running, you can see the monitoring progress on a screen Pic 5. It displays how many installations from those defined are already complete.




**Pic 5. Monitoring progress**

As soon as all the installations have been completed, the monitoring process is stopped automatically. The installation is treated as completed when the installer process and all the processes daughterly to the installer process exit.

Please note that if the setup wizard suggests running the installed application upon the installation completion, you should decline this suggestion. Otherwise, the monitoring module will wait for the started application to exit.

Copyright © 2001-2023 EMCO. All rights reserved.

You can abandon the monitoring process using the **Stop** button at any time, although it is not recommended.

## The monitoring process does not stop. Which option should I choose?

You can find yourself in a situation where the monitoring process does not stop after you have passed through all the steps of the setup wizard. To choose further steps, it is necessary to understand what leads to this situation. Let us take a closer look at the list of possible reasons and appropriate solutions for each one.

First, the monitoring process may not be stopping because the installation is still performing the application configuration in background. Thus, it is recommended to be patient and always wait for some time after the installer application user interface has closed.

Another possible reason is that the monitored installation has launched the installed application or one of its modules. This can happen if you have not unchecked an appropriate option on the installation wizard finish page, or if the installation always performs the step of launching the installed program. Please note that those executable modules can be run in the background or minimized to tray. In such a case, you should close the launched applications and continue waiting for the monitoring to stop automatically. The monitoring module will switch to next installation, if any, or create the deployment package, if this installation is the last one from those monitored, as soon as those applications are closed. In case if it is not possible to close the launched applications, but you are sure that the installation is fully complete, you can stop the monitoring process manually and create the deployment package based on the changes currently monitored by pressing the **Stop** button and then choosing the option to proceed with creation.



It is not recommended that the monitoring process be stopped manually to proceed with the deployment package creation unless you are absolutely sure that the installation is fully completed. In case the setup is still in progress, the repackaging will be incomplete.

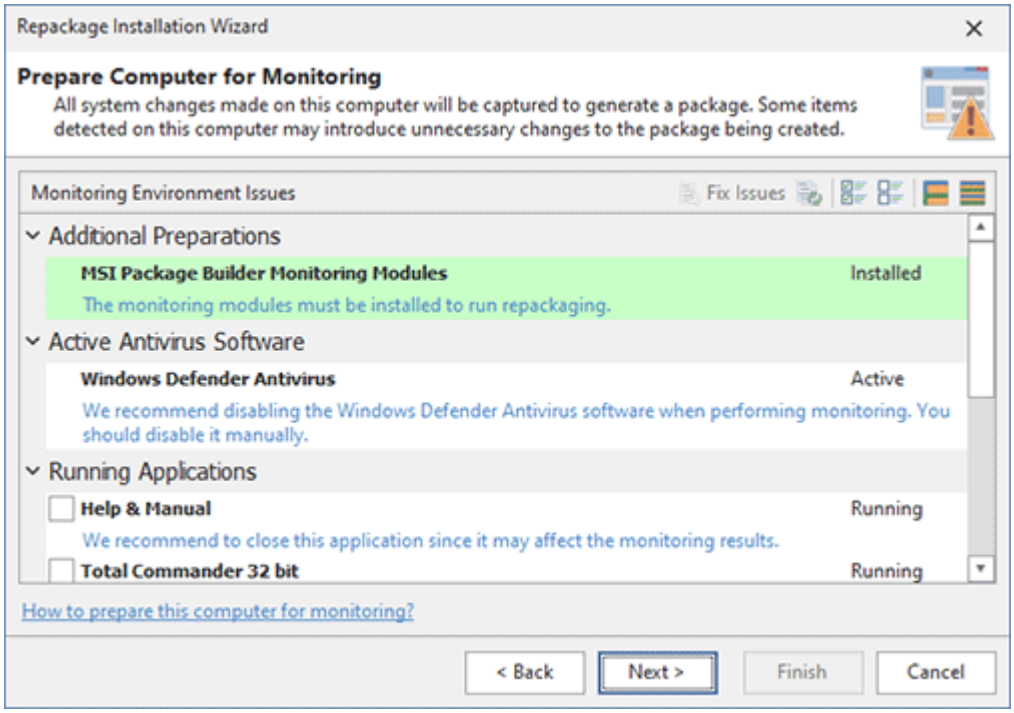
The last reason may be that the installation does not fulfill the requirements for monitoring (the process performing the installation does not exit after the setup is completed). If you are not sure that the installation process is fully completed and cannot find out the processes that are preventing the monitoring from being completed, you can abort the monitoring process by pressing the Stop button and then choosing the Abort Monitoring option. All the monitored changes will be irreparably lost. In such a case, it may be convenient for you to use repackaging via wrapping.

Copyright © 2001-2023 EMCO. All rights reserved.

## Capture System Changes

With MSI Package Builder, you can track changes made to your operating system and create a deployment package based on these changes. With the help of the monitoring technology, you can easily create an MSI, an App-V or an MSIX/AppX package from any installation you want and even from any changes you performed manually. Just start monitoring, perform the changes and stop monitoring, and the newly created project will contain all these changes. You can monitor system changes and create a deployment package on completion choosing the **Capture System Changes** option within the **Repackage Installation** wizard.

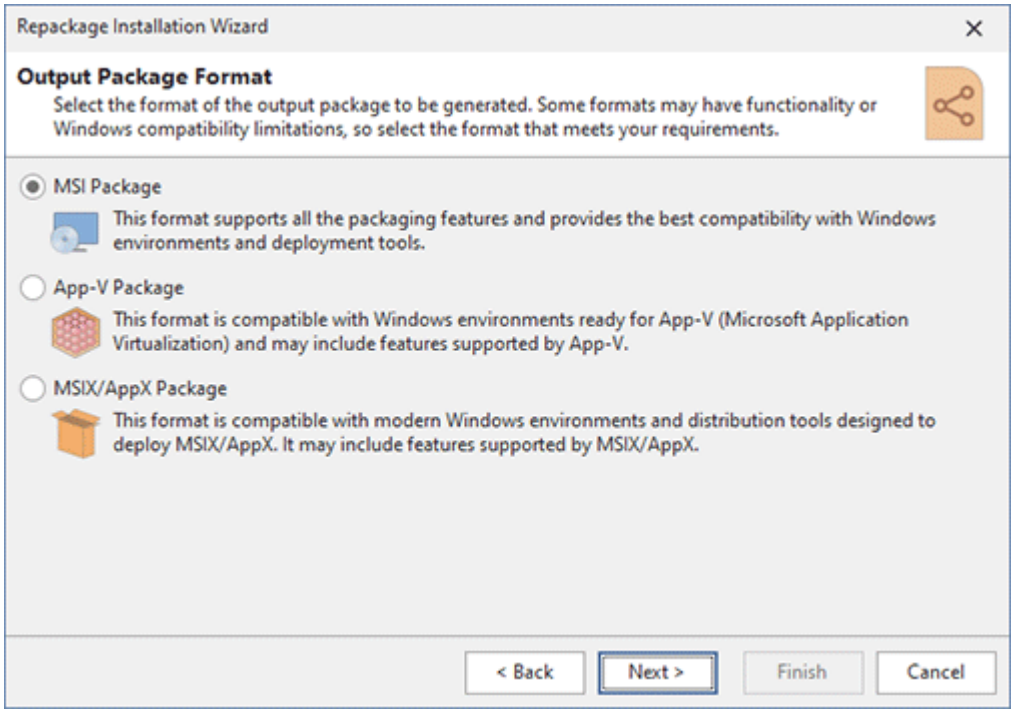
Before starting monitoring, you need to make sure that the monitoring environment is clean and you follow the repackaging best practices. The program can check some requirements automatically and show the list of monitoring environment issues Pic 1. You need to resolve these issues by following the displayed recommendations. For example, if any Windows applications are running, you can select them and press the **Fix Issues** button, so these applications will be closed.




**Pic 1. Monitoring environment issues**

Copyright © 2001-2023 EMCO. All rights reserved.

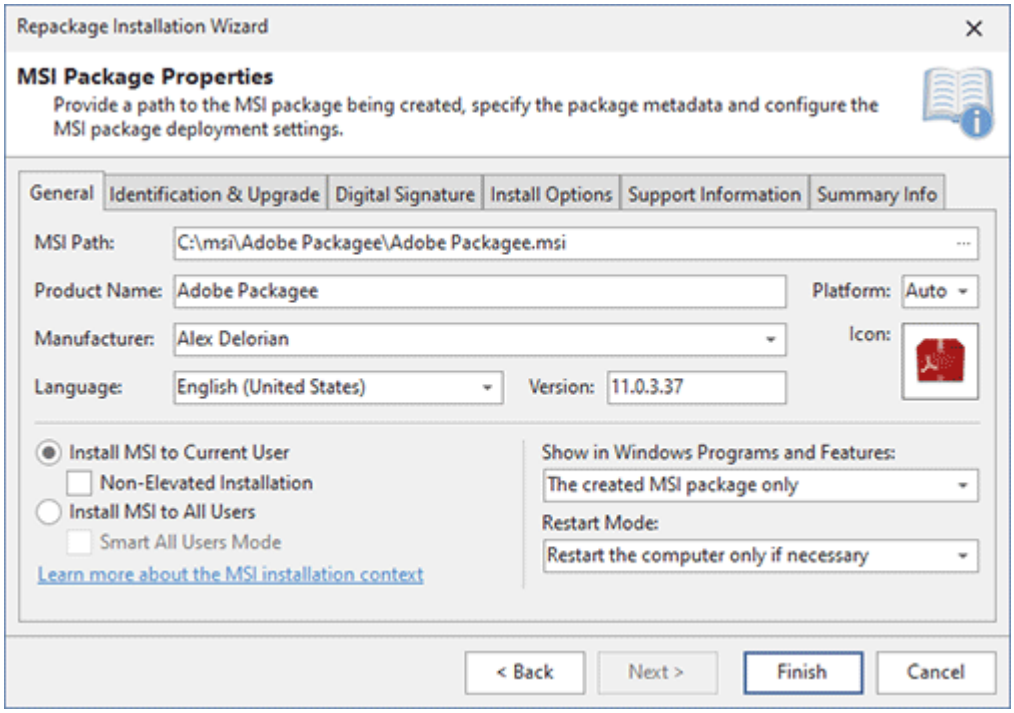
When configuring the monitoring process, you are suggested to provide a name for the project to be created as a result of monitoring. This project will contain all changes made by the running processes to the operating system during the monitoring session. At the next step, you should choose if you would like to create an MSI package, an App-V package or an MSIX/AppX package based on the monitoring results.




**Pic 2. Choosing the package type**

*Copyright © 2001-2023 EMCO. All rights reserved.*

In case you have chosen to create an MSI package, the next step is to provide a path to a location to save the resulting MSI package to and configure the package Pic 3. Detailed information on configuring an MSI package is available in the Creating an MSI Package section of this document. section of this document.




**Pic 3. Configuring an MSI package**

Copyright © 2001-2023 EMCO. All rights reserved.

If the output in form of an App-V package has been selected, you need to provide a path to save the resulting App-V package to and configure the package. Detailed information on configuring an App-V package is available in the Creating an App-V Package chapter. As for the MSIX/AppX package output, you need to provide a path to save the package to, define the package configuration options, the target platforms and the digital signature settings. Refer to the Creating an MSIX/AppX Package chapter for details.

Pressing the **Finish** button starts the monitoring process, and from that moment until the moment you stop it, all changes, except for those skipped in view of the monitoring filters configuration, are recorded and saved to the specified project. A deployment package is created at a specified location based on those changes.

## When should I stop the monitoring process?

While system changes are being monitored, the monitoring process cannot be stopped automatically because there is no trigger for that. Therefore, it is up to you to choose the moment when it should be stopped. Before stopping the monitoring process, you should make sure that all required changes have been made. It means that the installations you have monitored have exited, and all the operations have been fully completed, i.e. the required files have been fully copied, moved, created or deleted, and the required registry keys and values have been fully saved to the Windows registry, etc.

As soon as you stop the monitoring process, the project is created containing the recorded changes. If you are using the **Repackage Installation** wizard, this project is automatically prepared and a deployment package is created. For the manual capturing mode, you should prepare the created project before building a deployment package. See the Projects Preparation section for the information about the required preparation steps.

*Copyright © 2001-2023 EMCO. All rights reserved.*

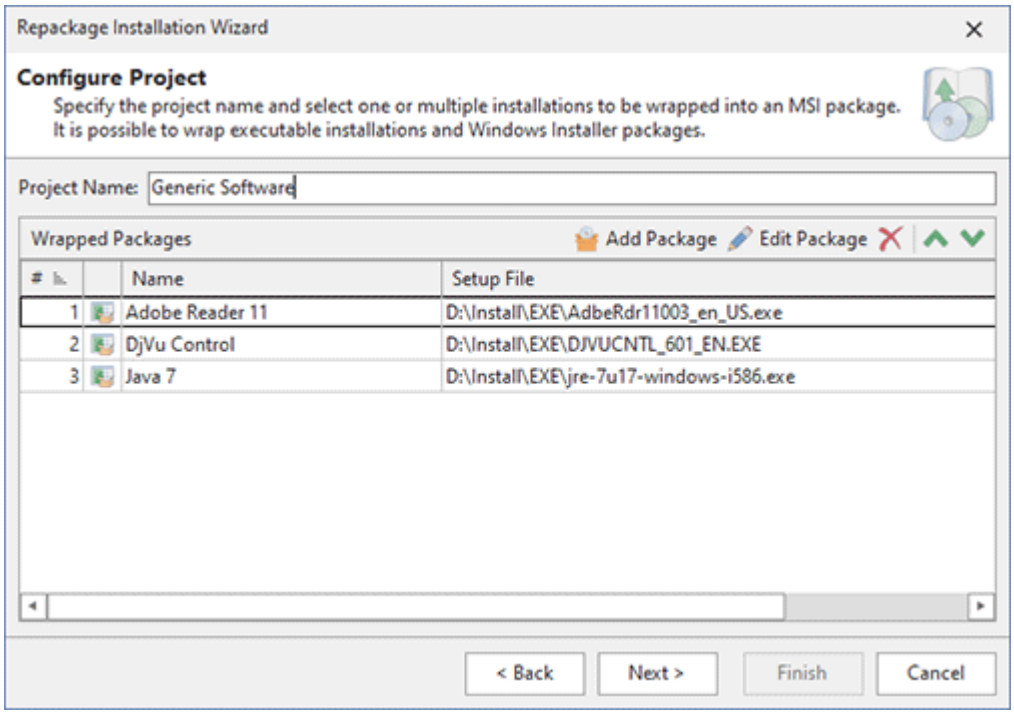
## Repackaging via Wrapping

One of the installations repackaging methods used by MSI Package Builder is wrapping existing installations into a generated MSI package. Wrapping means that those installations are distributed as a part of the MSI package and are installed, uninstalled and repaired simultaneously in the specified order.

Repackaging via wrapping can be used to build MSI packages only. App-V packages require that package content be explicitly defined before building a package, thus it is not possible to launch installations while deploying a virtual application. As the MSIX/AppX packages use an isolated sandbox environment concept for deployed contents, it is also not possible to launch installations while deploying those packages.

The advantage of this mode is that the original installation is always executed on each PC, so there should be no chance that a generated MSI package will be installed incorrectly on specific target machines. The disadvantage of this mode is that you need to know the silent installation parameters and possibly provide installation scenario files to generate an MSI package that can be used for silent deployment.

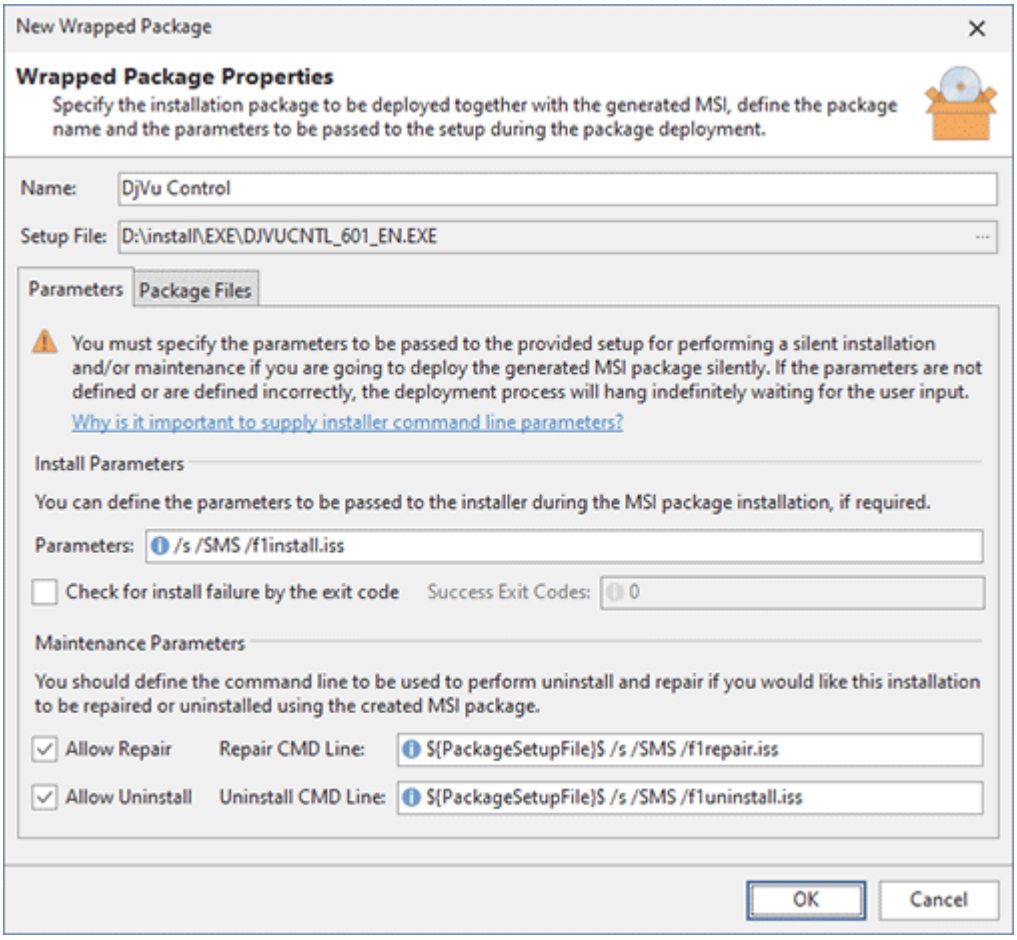
To perform repackaging via wrapping, either choose the **Wrap Installation** option in the **Repackage Installation** wizard or click the **Wrap Installation** shortcut in the **Product Actions** group on the **Welcome Screen**. In any case, you'll be able to provide a set of installations to be wrapped and their deployment order Pic 1.




**Pic 1. Repackaging via wrapping**

Copyright © 2001-2023 EMCO. All rights reserved.

MSI Package Builder allows you to wrap both Windows Installer packages and executable installation packages. For each package, you can define if it should be repaired and uninstalled while repairing and uninstalling a generated MSI package. You can also specify a list of additional files, if required, and the parameters to be passed to the installer to perform install, uninstall and repair Pic 2. The provided packages are included into a created MSI package and saved into a project with the specified name for future usage. Detailed information on configuring wrapped packages is available in the Wrapping Existing Installations chapter.




**Pic 2. Configuring MSI package**

The next step is to provide a path to the location to save a resulting MSI package to and configure the package. Detailed information on configuring an MSI package is available in the Creating an MSI Package chapter.

As soon as the package is configured, press **Finish** to proceed with its creation. The created package is saved to the specified location and is ready for deployment. The project representing the current repackaging process is also saved to the projects storage.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Low-Level Repackaging

MSI Package Builder comes with a wide range of features available for creating standalone deployment packages and repackaging existing installations. Most of repackaging tasks can be easily solved using the smart repackaging modes available in the **Repackage Installation** wizard, but the program does not put any restrictions on the ways used to for building a deployment package. All the features available in the smart mode can be used both as standalone ones and in a combination with each other while performing a low-level repackaging. Let us take a closer look on what does the low-level repackaging mean and how is it possible to benefit from it.



The low-level repackaging concept should be used by experienced users only. While using a low-level repackaging you are responsible for performing each and every step required for the successful repackaging, so it is required that you fully understand those steps.

The advantage of the low-level repackaging is the fact that you are not limited by any single repackaging technique and can combine those techniques to reach the desired goal. For example, you can monitor the installations you want to repackage and then either provide additional changes to be performed to a target Machine during an MSI package deployment, or correct the monitoring results, if required.

When using the feature of capturing changes made to the underlying operating system, you have to prepare the created project before creating a deployment package. Preparation consists of several important steps. First, MSI Package Builder should copy all file system resources to a project storage. If there are any problems during the copying process, the problematic files and folders are marked as missing links. The problems with missing links must be resolved before proceeding with the package generation. For a deployment package to be correctly installed on different operating systems with different environment configurations, it is required to replace all paths that represent the system folders with a special system folder object. All those steps are described in detail within the Projects Preparation section of this document.

When providing the required changes manually, you just create a project to contain those changes, and then provide the changes to the project. Detailed information on providing and modifying the changes is available in the Installation Projects section of this document.

If you want to use wrapping, the packages to be deployed together with an MSI package, those are wrapped packages, are provided in the **Wrapped Packages** view in the same manner as when performing repackaging via wrapping. For detailed information on wrapping existing installations, refer to the Wrapping Existing Installations section of this document.

As soon as all required changes are provided or the wrapped packages are configured, depending on the project type, you can proceed with a deployment package creation. To learn more about the options available while creating deployment packages, see the Creating an MSI Package, the Creating an App-V Package and the Creating an MSIX/AppX Package sections of this document.

In general, the low-level repackaging is a set of steps to be performed for creating a deployment package that include all required changes and actions. It is really flexible and easy-to-use for those, who fully understand the steps required for performing successful repackaging. Using the available abilities it is possible to perform repackaging of any complexity, but you must always be particularly accurate while using the low-level repackaging in your everyday work.

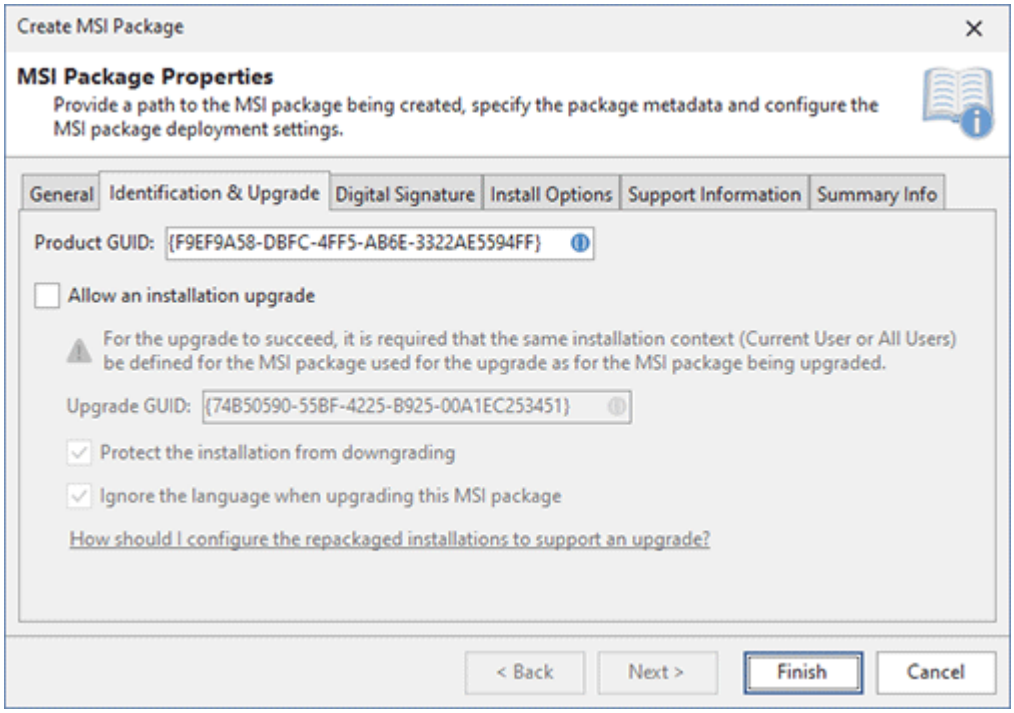
Copyright © 2001-2023 EMCO. All rights reserved.

## How should I configure the repackaged installations to support an upgrade?

MSI Package Builder allows you to perform installation repackaging fast and easy. But a single instance repackaging is not enough in a real life. It is obvious, that when a new version of the repackaged software is released, you are going to create a new MSI package, containing this version. In this chapter, we will show you how to create an MSI package for repackaging a new version of any application.

The information below must be read by everyone who is going to repackage the upgraded installations and then deploy a resulting MSI package to Machines where the previous version of the package is installed. If the conditions described below are not met, the upgrade won't be performed correctly and the **Programs and Features** section of the **Control Panel** will display both versions of the repackaged installation.

For MSI packages to be deployed and upgraded correctly, they should be properly identified. For each MSI package there is a **Product GUID**, that uniquely identifies the product, and an **Upgrade GUID** that is used for matching upgrades. Windows Installer maintains its database and handles its consistency, depending on these identifiers. These identifiers are defined on a project scope while creating and an MSI package or in the **Project Details** view.




**Pic 1. Identifying the generated MSI package**

The **Product GUID** value for the different versions of the installations being repackaged should differ, and the **Upgrade GUID** value should match. So, for example, if we are using the GUIDs displayed on Pic 1 while repackaging the *Abode Reader 10.1.0* installation, the same **Upgrade GUID** in combination with another **Product GUID**, must be used for the *Abode Reader 10.3.1* installation.

If you have lost the project used to create a previous version of the MSI package, thus cannot find GUIDs used to identify that project, you can import that package. You'll find the required GUIDs in the **Project Details** view when the package created on a basis of the import result is selected.

Copyright © 2001-2023 EMCO. All rights reserved.

It is also required that at least one of the first three version figures of the MSI package used for upgrade differs from the version of the previously generated package. For example, if we have used the *1.0.0* version of the MSI package while repackaging the *Abode Reader 10.1.0* installation, then we can use *1.1.0* version for the *Abode Reader 10.3.1*, but we cannot use *1.0.0.3*.



If you are going to repackage only a single installation into an MSI package, but not a suit of applications, it is recommended that the application version be used as the MSI package version. It will make the configuration process easier.

For the successful upgrade it is also required that both MSI packages should have the same installation context (Current User/All Users) defined.

Before performing a mass deployment of the upgraded package, it is strongly recommended that the upgrade process be checked for correct work on a test PC. First, install a previous version of an MSI package, and then install an upgraded version. If you have configured the packages correctly, you'll be able to see that the new version of the application is installed and there is a single entry in the **Programs and Features** section of the **Control Panel** representing this package.

Following the above-stated recommendations, you'll be able to create MSI packages that fully support upgrade, thus be able to upgrade the applications installed by the repackaged setups.

## Repackaging on Windows Sandbox

The program allows you to use Windows Sandbox environment to perform monitoring. Sandbox is a lightweight Windows desktop environment to run applications in isolation. Software installed inside the Windows Sandbox environment remains "sandboxed" and runs separately from the host machine.

Windows Sandbox provides a clean environment for monitoring that follows the repackaging best practices. Sandbox has the following features:

- **Is a part of Windows**. Sandbox is a Windows feature available starting from Windows 10 and above.
- **Clean environment**. Every time Sandbox is started, it runs a clean, brand-new Windows installation.
- **Isolation**. Sandbox uses virtualization, so all changes made in Sandbox are isolated from the host that runs Sandbox.
- **Automatic cleanup**. When you close Sandbox, all changes made in Sandbox are discarded.

## How to Enable Sandbox

A machine running Sandbox should meet the following requirements:

- Run Windows 10 build 18305 or later Windows versions. Sandbox isn’t available on the Windows Home edition.
- A machine should have AMD64 or ARM64 architecture.
- A machine should have enabled hardware virtualization capabilities.
- 4GB of RAM minimum.
- 1GB of free disk space.
- 2 CPU cores minimum.

Copyright © 2001-2023 EMCO. All rights reserved.

To enable Sandbox you need to enable hardware virtualization capabilities in BIOS settings. After enabling virtualization you need to open **Windows Program and Features** and click **Turn Windows features on or off**. In the list of features you need to find and enable Windows Sandbox and reboot the machine.

## Sandbox Usage

When you use Sandbox for monitoring the program runs Sandbox automatically, copies the specified installation file to Sandbox to repackage it there. You need to follow installation steps on Sandbox. At the end of installation the program extracts monitoring results from Sandbox and generates a package.



Sandbox environment isn't an identical analog of the regular OS. Some Windows features aren't available on Sandbox and it may influence software installations. Some installations fail to deploy on Sandbox, so that they cannot be repackaged in Sandbox environment. If a repackaged installation cannot be deployed in Sandbox, you can try to repackage it on a virtual machine, for example.

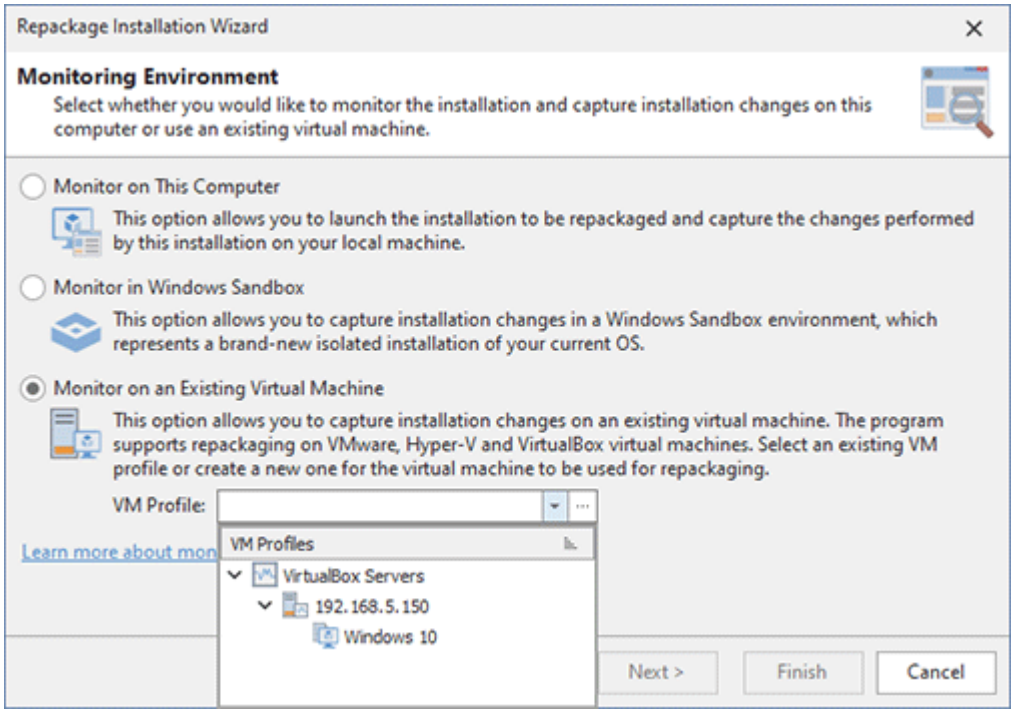
Using Sandbox for monitoring is a good option because you don't need to prepare environment, it provides a clean environment out of the box, but Sandbox has functional limitations and some installations fail to deploy in Sandbox. Using a virtual machine for repackaging is an alternative option that doesn't have Sandbox limitations, but allows using clean environment quickly and easily. This option is described in the next chapter.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Repackaging on a Virtual Machine

The program allows you to perform monitoring on an existing virtual machine connected to remotely. In this case EMCO MSI Package Builder is installed on your local machine, and it connects to a Hyper-V, VMware or VirtualBox virtual machine to perform monitoring on it. At the end of monitoring, all captured changes are copied to the local machine and a package is generated locally. Monitoring on a remote virtual machine is recommended instead of monitoring on a local machine because it allows avoiding some extra work. For example, you don't need to install EMCO MSI Package Builder on every machine used for monitoring, so you can easily switch among repackaging environments. Besides, the program can automatically revert virtual machines to a specified snapshot with a clean VM state at the end of monitoring, so you don't need to do it manually.

To perform monitoring on an existing virtual machine, you should select a corresponding option in the **Repackage Installation** wizard Pic 1, where you need to choose the VM profile used for monitoring. This profile includes the VM configuration stored in **VM Profiles Manager**.




**Pic 1. Select a VM as the repackaging environment**

You can learn how to configure virtual machines to be used for monitoring in the Requirements for Virtual Machines chapter. To use a VM in the program, you need to create a profile for it. This topic is explained in the Managing VM Profiles chapter. A practical example of using a VM is provided in the How to Use a Virtual Machine for Repackaging chapter.

### What's Inside

Requirements for Virtual Machines
Managing VM Profiles
How to Use a Virtual Machine for Repackaging

Copyright © 2001-2023 EMCO. All rights reserved.

### Requirements for Virtual Machines

The program can work with virtual machines hosted on Hyper-V, VMware and VirtualBox servers. There are specific requirements for virtualization servers, virtual machines and OS configurations on VMs, which are listed below. Make sure that your environment is configured to satisfy these requirements.

## Virtualization Server Requirements

Virtualization servers that host virtual machines must meet the following requirements:

**Hyper-V:**

- The program supports local and remote Hyper-V servers.

**VirtualBox:**

- To work with virtual machines hosted locally (on localhost), make sure VirtualBox version 6.0 or above is running on the local machine before starting MSI Package Builder.
- To work with virtual machines hosted remotely, use a server with VirtualBox version 6.1.

**VMware:**

- To work with virtual machines hosted locally (on localhost), make sure VMware Workstation (version 12 or above) is running on the local machine before starting MSI Package Builder. For MSI Package Builder to discover a VM, the VM should be powered on before creating a profile for it in the program.
- To work with virtual machines hosted remotely, you need to use the VMware ESXi server (version 6.0 or above) with a license that allows remote connections.

To work with the local VMware virtual machines using VMware Workstation, you need to make sure that VMware VIX is configured properly. VMware VIX is an API that allows manipulating files within a guest operating system. VMware VIX configuration is available in the *vixwrapper-product-config.txt* file located in the VMware Workstation installation folder. This file translates the product version specifications into the appropriate VIX implementations, so you have to specify correct settings there. For example, for all VMware Workstation builds of version 15 you have to specify *Workstation-15.0.0* as VIX implementation.

```
# Workstation 15.5.0
ws        19  vmdb  15.5.0 Workstation-15.0.0
player    19  vmdb  15.5.0 Workstation-15.0.0
# Workstation 15.5.5
ws        19  vmdb  15.5.5 Workstation-15.0.0
player    19  vmdb  15.5.5 Workstation-15.0.0
# Workstation 15.5.6
ws        19  vmdb  15.5.6 Workstation-15.0.0
player    19  vmdb  15.5.6 Workstation-15.0.0
```



VMware Workstation or VirtualBox may conflict with Hyper-V installed on the same machine. If you need to run VMware Workstation or VirtualBox on the machine with Hyper-V, you need to disable Hyper-V first. You can run the following command to disable Hyper-V and restart the PC to apply the changes.

*bcdedit /set hypervisorlaunchtype off*

Copyright © 2001-2023 EMCO. All rights reserved.

## Virtual Machines Requirements

Virtual machines and their configuration should meet the following requirements:

**For all types of virtual machines:**

- The virtual machine can run Windows Vista SP2 or a higher OS.
- In the VM profile, specify a Windows account (by providing the username/password) that has the Local Administrator permissions on the VM.
- Before starting monitoring, log in on the virtual machine with the user account specified in the VM profile. If a snapshot (a checkpoint on Hyper-V) is selected in the VM profile, it is recommended to use an "online" snapshot (checkpoint), i.e. the one with turned-on Windows and logged-in user.



It's recommended to use "online" snapshots (or checkpoints on Hyper-V) to optimize the repackaging process. In such snapshots, Windows is running and the user is logged in, so you don't need to start Windows and log in to Windows manually every time you make repackaging. To create an "online" snapshot, make sure you have configured the virtual machine to follow the repackaging best practices. Then start the virtual machine, log in to Windows and make a snapshot when you are logged in.

**Hyper-V:**

- Use checkpoints of the standard type only. Other types of checkpoints aren't supported.
- Enable remote access in Windows on the virtual machine configured in the VM profile (see the explanation below).

**VirtualBox:**

- For a VM hosted locally, click 'Yes' on the Windows UAC prompt displayed on the virtual machine to allow monitoring.
- For a VM hosted remotely, enable remote access in Windows on the virtual machine configured in the VM profile (see the explanation below).

**VMware:**

- VMware Tools must be installed on the guest operating system.
- Click 'Yes' on the Windows UAC prompt displayed on the virtual machine to allow monitoring.

Copyright © 2001-2023 EMCO. All rights reserved.

## How to Configure Windows Running on a Hyper-V or VirtualBox VM to Enable Remote Access

When the program uses a VM for monitoring, it copies an installation file to the VM before monitoring and retrieves the capturing results after monitoring. Virtualization servers provide different levels of support for communicating with VMs. To work with a Hyper-V server or with a remote VirtualBox server, you need to perform some extra configuration steps. You need to enable remote access in Windows on the VMs so that the program can communicate with those VMs.

You need to enable remote access in Windows for every VM you plan to use for repackaging that are hosted on a local or remote Hyper-V server or on a remote VirtualBox server. Follow the instructions below to enable remote access.

**Enable Remote Access**

Make sure that **NetBIOS over TCP/IP** is enabled on the network adapter of the virtual machine in Windows. This option is configured in the **WINS settings** of the TCP/IP protocol configuration. In the **NetBIOS settings** group, you should either choose the **Enable NetBIOS over TCP/IP** value or leave the **Default** value if a static IP is used or the DHCP server in your domain is configured to enable NetBIOS.

In addition, you need to enable **File and Printer Sharing**. You can do it in the **Network and Sharing Center** of Windows on the virtual machine. Open the connection properties and turn on **File and Printer Sharing**.

**Enable Remote Access on Windows Running in a Workgroup**

If a virtual machine runs Windows connected to a corporate domain, its remote access is already enabled after joining the domain, so no additional settings are required. If a virtual machine runs Windows connected to a workgroup, you need to enable remote access for it. To enable remote access, you should disable the UAC remote restrictions by creating the **LocalAccountTokenFilterPolicy** DWORD value and setting it to *1* in the following registry key:

*HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System*

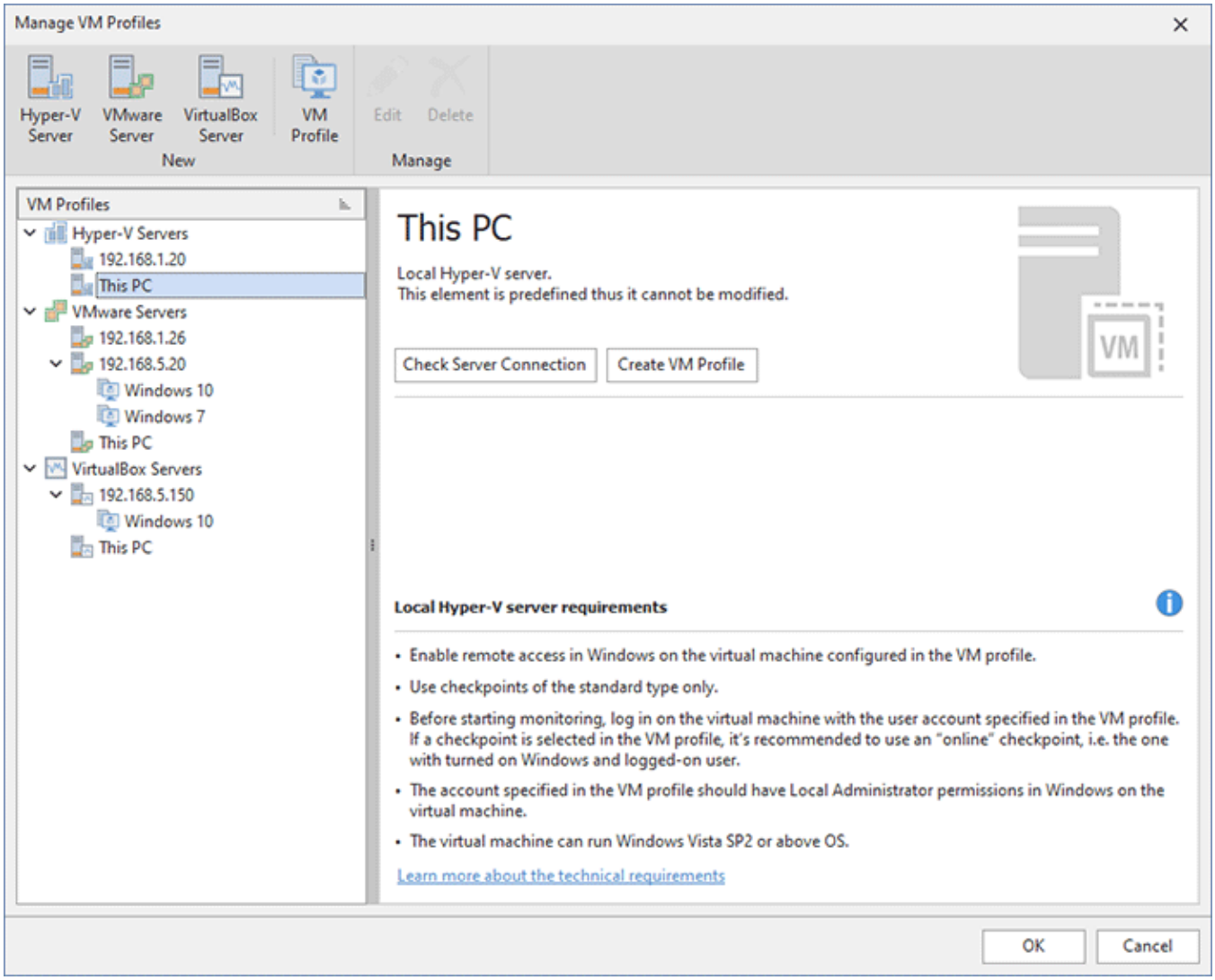
You can use the Microsoft support article as the reference.

To make sure your virtualization server and virtual machine are configured properly, you need to use the button in **VM Profile Manager** to test the VM profile settings. If you get an error, it means that the VM has problems, so you need to check the requirements and try again.

Copyright © 2001-2023 EMCO. All rights reserved.

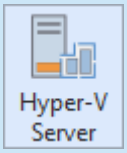
## Managing VM Profiles

The program stores configurations of virtual machines used for monitoring in VM profiles. To configure VMs, you need to click the **Manage VM Profiles** button located in the **Capture** group of the **Home** tab on Ribbon. **VM Profiles Manager** is used to configure Hyper-V, VMware and VirtualBox servers and virtual machines hosted on those servers that can be used by the program for repackaging. The profiles manager dialog consists of the toolbar with the main actions located at the top, the list of VM servers and VM profiles located on the left, and the information area located on the right Pic 1.



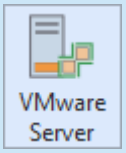

**Pic 1. VM profiles manager**

The profiles manager has the following actions displayed on the toolbar.




**Hyper-V Server**

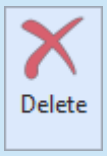
This button allows adding a new Hyper-V server that hosts virtual machines.



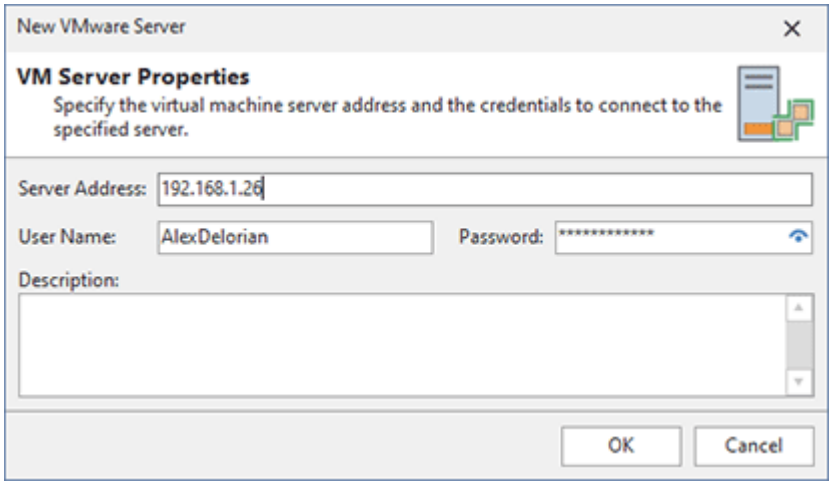

**VMware Server**

This button allows adding a new VMware server that hosts virtual machines.

Copyright © 2001-2023 EMCO. All rights reserved.

| | |
|---|---|
|  | **VirtualBox Server**<br>This button allows adding a new VirtualBox server that hosts virtual machines. |
|  | **VM Profile**<br>This button allows adding a new profile with the virtual machine configuration. |
|  | **Edit**<br>This button allows changing properties of the selected item. |
|  | **Delete**<br>This button allows deleting the selected items. |

In **VM Profiles Manager**, you can add Hyper-V, VMware and VirtualBox servers that host virtual machines that you plan to use in the program for repackaging. Note that the program has predefined configurations for local servers displayed in the **VM Profiles** tree, so that you can create new VM profiles on any of them if you plan to work with virtual machines hosted on the local PC. To add a new remote server, you need to click the **Hyper-V Server**, **VMware Server** or **VirtualBox Server** buttons on the toolbar or select the corresponding options in the context menu of the **VM Profiles** tree. In the displayed configuration dialog, you need to specify settings for connecting to a VM server Pic 2.

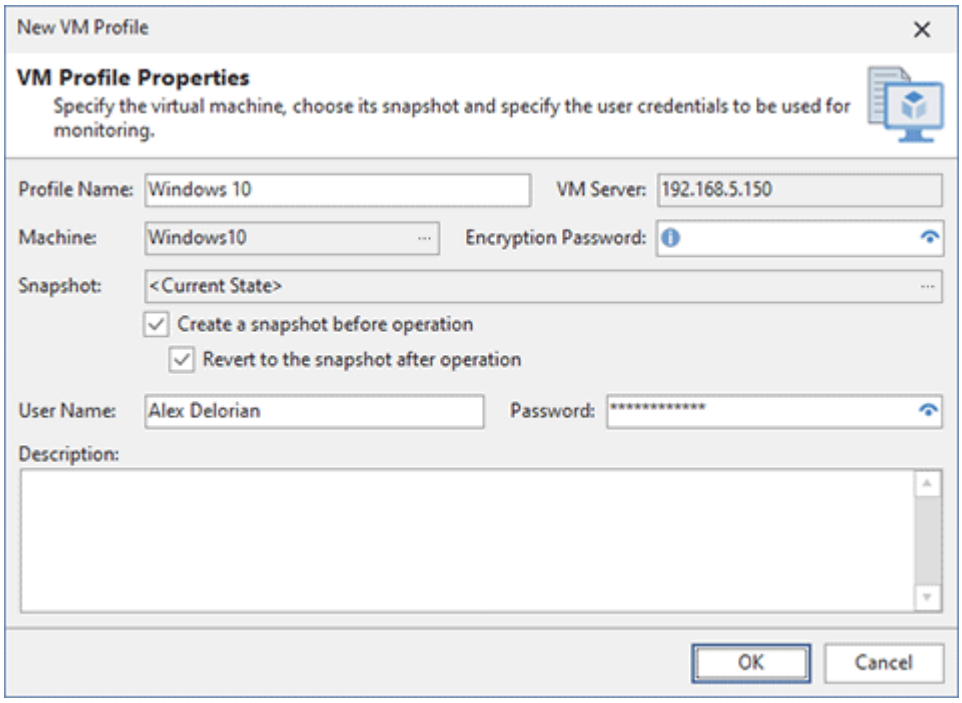



**Pic 2. Configuring a VM server**

Copyright © 2001-2023 EMCO. All rights reserved.

Server configurations for all types of servers are identical. You need to specify the **Server Address**, **Username** and **Password**. After being added, a server appears in the VM Profiles tree, and you can select it there. Press the **Check Server Connection** button to make sure the server is accessible.

After configuring a server, you can configure a virtual machine by creating its VM profile. On the server, select the option of creating a new VM profile by choosing the corresponding option in the context menu or on the toolbar. As a result, the VM configuration dialog appears Pic 3. It is identical for all types of virtual machines and includes the same options.




**Pic 3. Configuring a VM profile**

Copyright © 2001-2023 EMCO. All rights reserved.

In the displayed dialog, you need to specify the **Profile Name**. This name will be displayed in the **VM Profiles** tree, and you can see it in the **Repackage Installation** wizard when you select a VM profile for repackaging. The server can host multiple virtual machines, so you need to press  to choose the required one. After selecting a **Machine**, you need to configure the **Snapshot** that will be used for monitoring. Press  to select the required snapshot. If you would like to use the current actual state of the VM, you can select the *<Current State>* option.

Depending on the selected snapshot, you can configure the additional options. If *<Current State>* is selected, you can enable the **Create a snapshot before operation** option, so the program will automatically create a new snapshot before monitoring. You can also select the **Revert to the snapshot after operation** option, and the program will automatically revert the VM to the created snapshot at the end of monitoring so that you always use a clean VM for monitoring. If you have selected an existing snapshot, the program allows you to revert the VM to this snapshot automatically at the end of monitoring by selecting the **Revert to the snapshot after operation** option.

When a new VM profile is created, you can select it in the **VM Profiles** tree and press the **Check Profile** button to test the VM. If the VM settings are configured properly, you can see a dialog that displays the discovered environment issues for the VM. Make sure that the VM is configured properly to follow the repackaging best practices. If for some reasons there is no connection to the VM, you will see an error message and you need to troubleshoot the problem.



The program has specific requirements for VM servers and virtual machines depending on the used virtualization provider, the location of the server (a local PC or a remote server), and the virtual machine and OS configuration. It's recommended to review the Requirements for Virtual Machines chapter to make sure that your VM configuration satisfies those requirements.

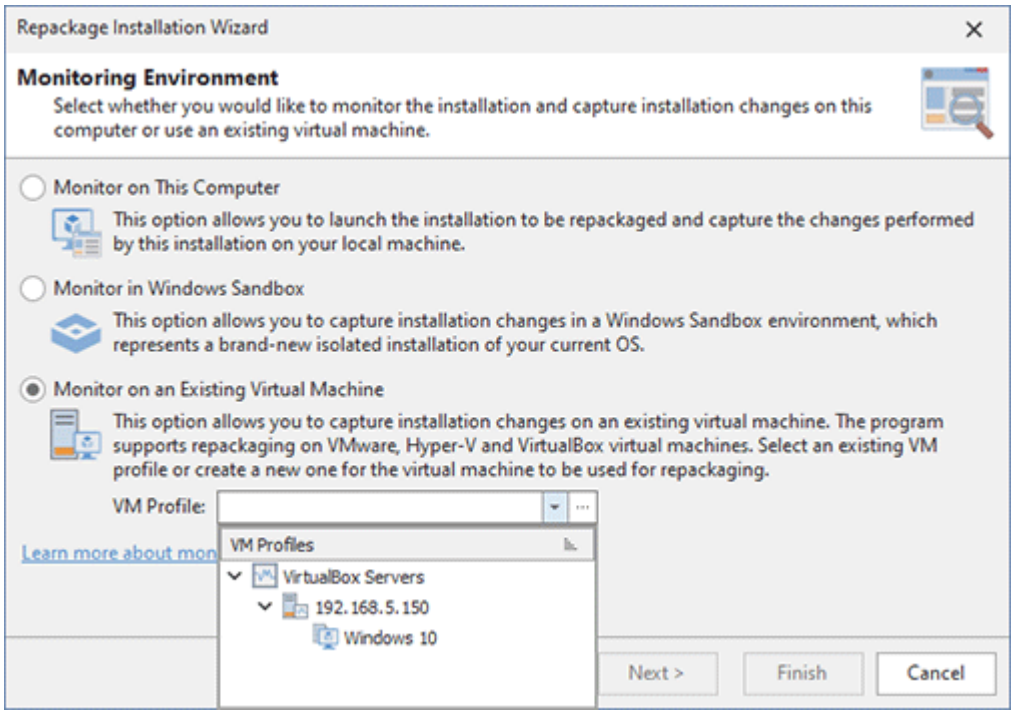
Once a VM is properly configured and tested, you can use it for repackaging. Select the VM profile in the **Repackage Installation** wizard and follow the repackaging process described in the How Monitoring on a Virtual Machine Works chapter.

Copyright © 2001-2023 EMCO. All rights reserved.

## How to Use a Virtual Machine for Repackaging

Repackaging on a remote virtual machine is similar to repackaging on the local machine, except for a few differences. The step-by-step process is explained in the demo chapter. Here you can learn about the differences caused by monitoring on a remote virtual machine.

To use an existing remote VM for monitoring, you need to select the corresponding option in the **Repackage Installation** wizard and select the required VM profile Pic 1. If you don't have a profile yet, you can create it right at this step by clicking ... button. You can learn how to configure a VM profile in the Managing VM Profiles chapter.

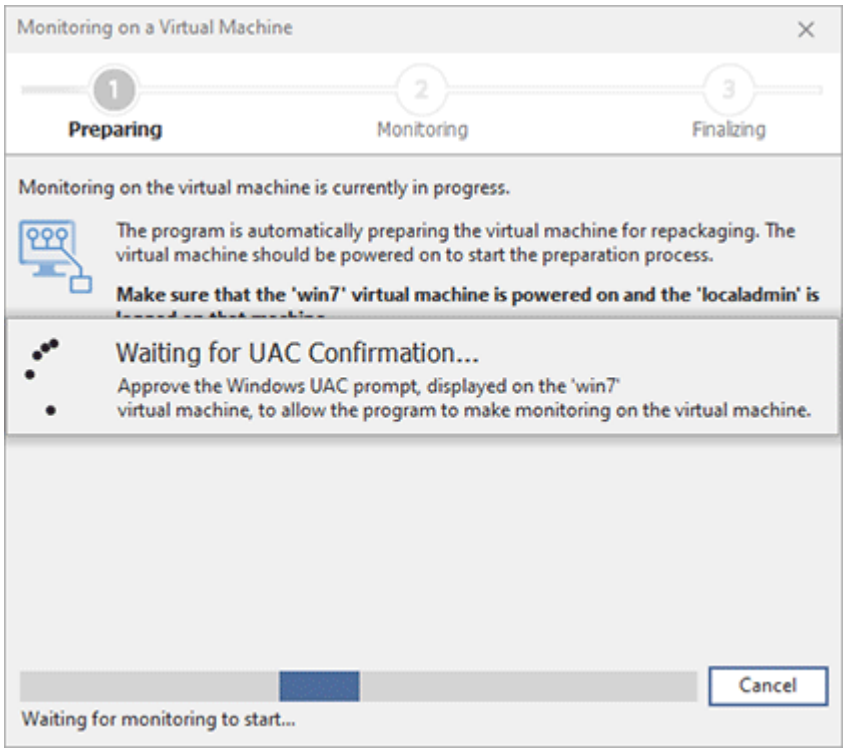



**Pic 1. Selecting a VM for monitoring**

Copyright © 2001-2023 EMCO. All rights reserved.

All other steps of the **Repackage Installation** wizard are the same as for monitoring on the local machine, so you can select the required monitoring type to capture an installation or system changes. You need to provide a path to the installation to be monitored and configure the options of the generated package.

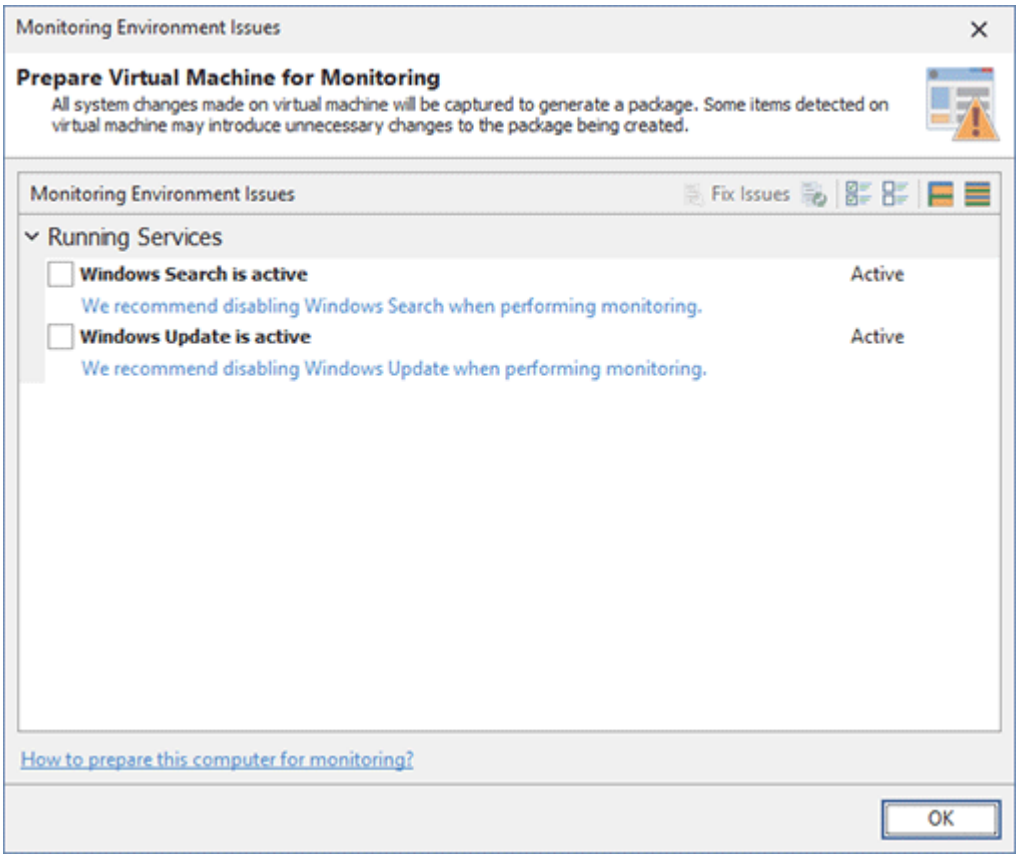
When all the settings have been provided and you click the **Finish** button in the wizard, the program interacts with the VM. Before starting monitoring, you need to log in on the virtual machine with the user account specified in the VM profile. At this step, the program needs to interact with the OS running on the VM, so if you use a VMware virtual machine, you need to click 'Yes' on the Windows UAC prompt displayed on the VM to allow such an interaction. The corresponding instructions are displayed by the program Pic 2.




**Pic 2. Information on the preparation steps displayed by the program**

*Copyright © 2001-2023 EMCO. All rights reserved.*

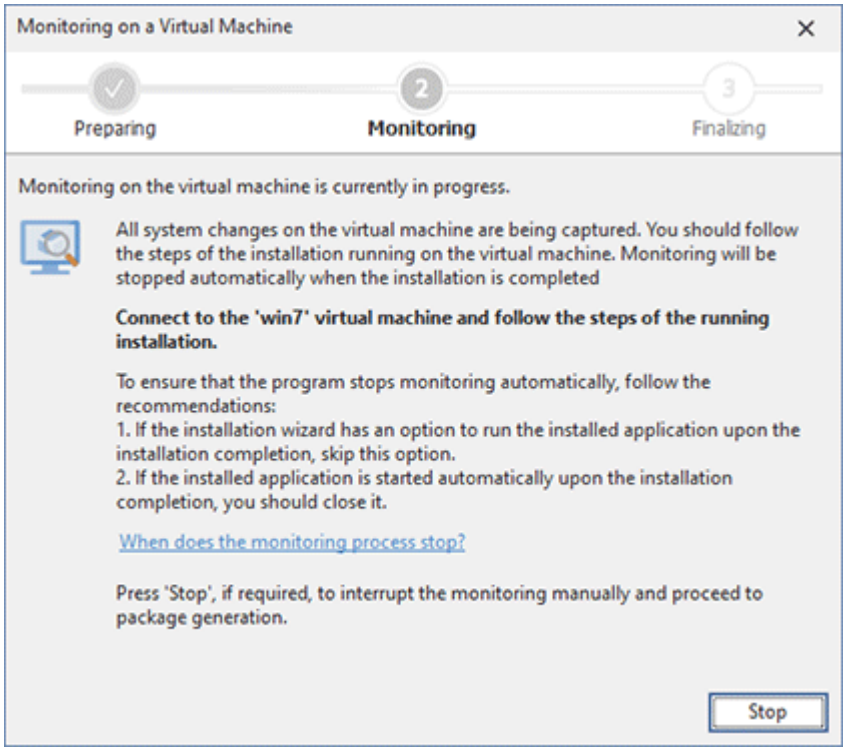
In scope of VM preparation for monitoring, the program needs to check if there are any monitoring issues on the VM. The corresponding dialog is displayed, and you can fix the issues if any have been detected Pic 3.




**Pic 3. Information about the monitoring issues on the VM**

*Copyright © 2001-2023 EMCO. All rights reserved.*

Once the monitoring issues have been resolved, the program automatically prepares the VM for monitoring. The installation file is copied to the VM, and the monitoring modules are checked and installed on the VM, if required. All these steps are performed automatically. You don't need to make any special preparations on the VM. Once everything is done, monitoring starts automatically, and the corresponding dialog with instructions is displayed Pic 4.

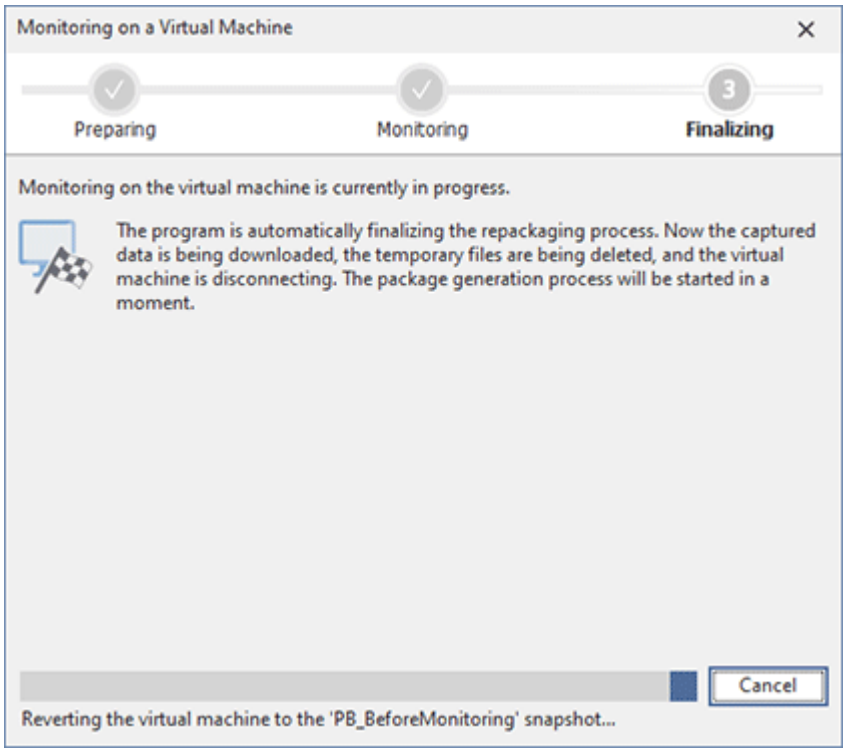



**Pic 4. Monitoring is in progress**

Copyright © 2001-2023 EMCO. All rights reserved.

At this point, you can open the VM and apply the required changes to be monitored. If you configured the program to capture an installation and specified the installation file, this installation should be automatically started on the VM, and you need to follow the installation steps. Monitoring will be stopped automatically once the installation is completed. If you selected the option of capturing system changes, you can apply those changes. When all the changes have been applied, you can go back to the program and press the button to stop monitoring.

Once monitoring is stopped (automatically or manually depending on the selected option), the program exports the captured data from the VM to be saved in the projects storage and runs the package generation. You can see the progress on the screen Pic 5.




**Pic 5. Final steps of monitoring**

If you choose the option in the VM profile to revert the VM to a selected snapshot, it is reverted automatically at the end of monitoring, so that next time you need to repeat repackaging, you can use a ready snapshot with a clean VM state.

The program generates a package using the monitoring results captured on the VM, and you can see the generated package on the file system. It's recommended to test this package before deployment in the corporate network. Follow the instructions available in the Package Testing and Troubleshooting chapter.

## How to Optimize Installation Monitoring on a VM

When you use an external VM for monitoring, the program needs to copy an installation to be monitored to the VM. If the installation file is large, it can take some time to copy it to the VM. To reduce the monitoring preparation time, you can select an installation file stored on a network share rather than selecting a local file. In this case, the program doesn't copy the installation to the VM, but runs the deployment from the network share. To use this approach, just make sure the network share allows access for the Windows account used on the VM for monitoring.

Copyright © 2001-2023 EMCO. All rights reserved.

# Chapter 5: Installation Projects

MSI Package Builder is a project based program – this means that all data is grouped by the projects, that are displayed in the **Projects** view. The interesting feature of the projects structure is that you can close the project if it is not needed right now then reopen it again. This allows you to keep the **Projects** view in an actual state during the everyday work.

Each project contains the information on file system, registry and other modifications – the deployment package based on the project will perform these modifications during the installation on the computer. The project stores all configuration options, required to create a deployment package between sessions. That makes it easier to manage updates. So, for example, if you have upgrade enabled in the project settings, all you need to create an updated package is to define the required modifications, change the version and provide another product GUID – everything else is already configured.

In this chapter we will take a closer look on the project structure, will show you how you can define the changes, specify actions to be performed during the package deployment, provide installations to be deployed together with the package and will guide you through the process of manual MSI package creation. This part is for those who want to fully understand the concept of MSI Package Builder and be able to use all available advanced features.

## What's Inside

Projects Management
Projects Preparation
File System Modifications
Registry Modifications
Environment Variables Modifications
Services Modifications
Side-by-side Assemblies Deployment
Drivers Deployment
Printers Deployment
Windows Firewall Modifications
Scheduled Tasks Deployment
Using Custom Actions
Wrapping Existing Installations
Capturing Changes
Creating an MSI Package
MSI Package Language
MSI Installation Context
Signing Packages
Exporting a Project
Importing a Project

Copyright © 2001-2023 EMCO. All rights reserved.

## Projects Management

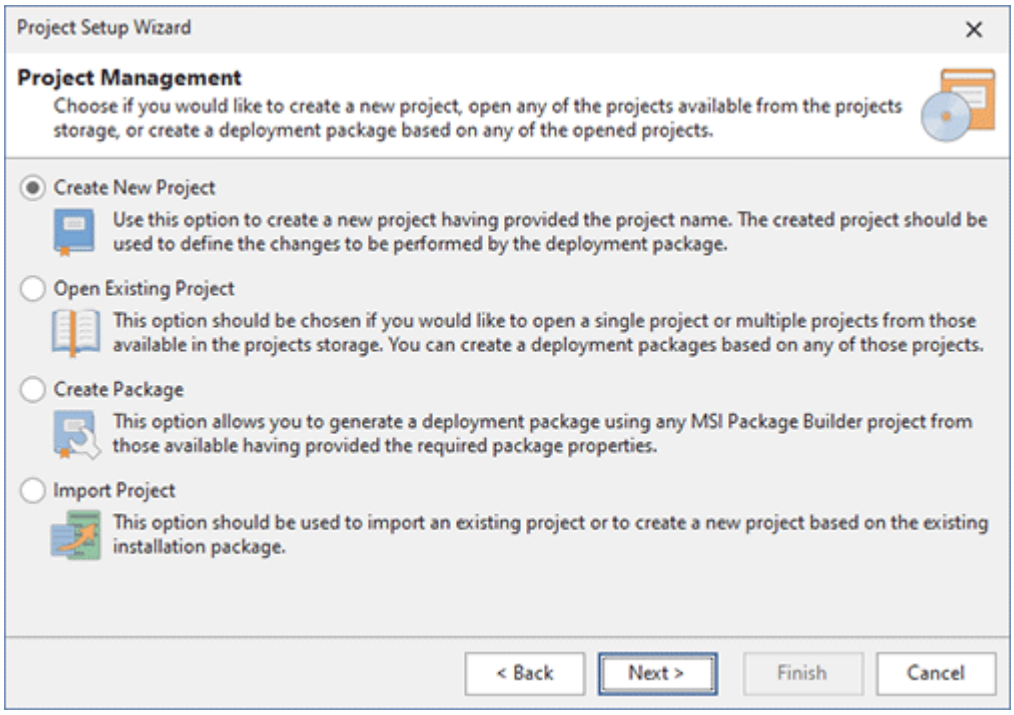
MSI Package Builder is a project based program, so a project is a main structure unit. While repackaging installations the required projects are created automatically, but it is also possible to create projects manually to be used for generating deployment packages. The projects that are not currently in use are closed, thus are not available in the **Projects** view. On each program start up, all projects are closed, and then you can open any project from those available to start working with it. Let us take a closer look on the projects management process, including creating, renaming, closing, opening and deleting projects.




**Project Setup**

The **Project Setup** button from the **Builder** group on the **Home** Ribbon page opens the project setup wizard, that can be used as an entry point both for managing MSI Package Builder projects and creating deployment packages on the basis of those projects.

To make it easier to create new projects (both empty and bases on an existing deployment packages), open existing projects, and create a deployment package based on an opened project we have designed a **Project Setup Wizard** Pic 1. To open this wizard, just press the **Project Setup** button from the **Builder** group on the **Home** Ribbon page.




**Pic 1. The Project Setup Wizard**

Copyright © 2001-2023 EMCO. All rights reserved.

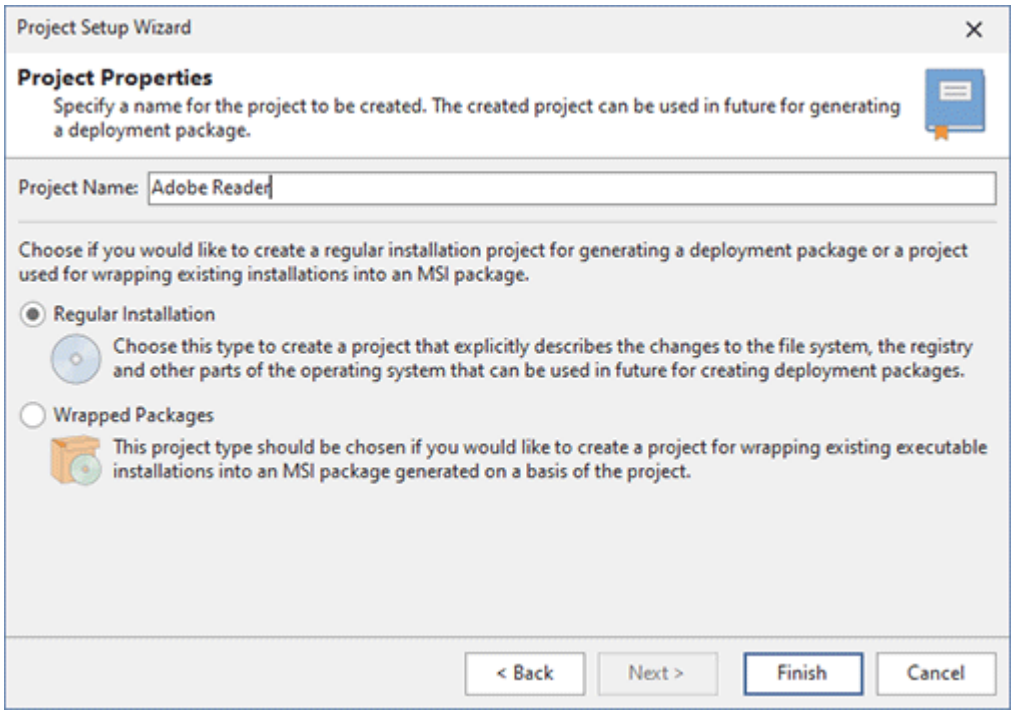
The first page of the wizard is used to introduce its abilities to you, and on the next page Pic 1 you are to choose the action you would like to perform. It is possible to choose between the following options: **Create New Project**, **Open Existing Projects**, **Create Package** and **Import Package**. The projects creation and opening process will be described later in this chapter. The **Create Package** option allows you to generate a deployment package using any project from those currently opened. For detailed information on the MSI package creation, refer to the Creating an MSI Package section of this document, for the App-V packages creation process, you can refer to the Creating an App-V Package section, and, as for the MSIX/AppX packages creation process, refer the Creating an MSIX/AppX Package section. The **Import Package** option should be used if you would like to create a new project, containing all the changes to be performed by an existing deployment package created with MSI Package Builder. This can be useful, if you need to change the package you have generated earlier but the original project is no longer available. Detailed information on the package import is available in the **Importing Package** section of this document.




**New**

The **New** button from the **Project Management** group on the **Home** Ribbon page should be used to create a new project.

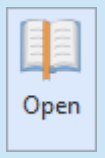
To create a new project in MSI Package Builder you can choose the **New Project** item from the **Projects** view pop-up menu or press the **New** button from the **Project Management** group on the **Home** Ribbon page. Alternatively, you can click the **New Project** link in the **Recent Projects** group on the **Welcome Screen**. The **New Project** dialog will appear on the screen Pic 2. The same pane is also displayed if you choose the **Create New Project** option in the **Project Setup Wizard**.




**Pic 2. Creating a new project**

Copyright © 2001-2023 EMCO. All rights reserved.

You are suggested to specify a name for the project to be created and the project type. You should pay enough attention to a project name; it will help you to identify the project in future. As for the project type, you are suggested to choose between the regular repackaging project and the project for wrapping existing installations into a single MSI package.




**Open**

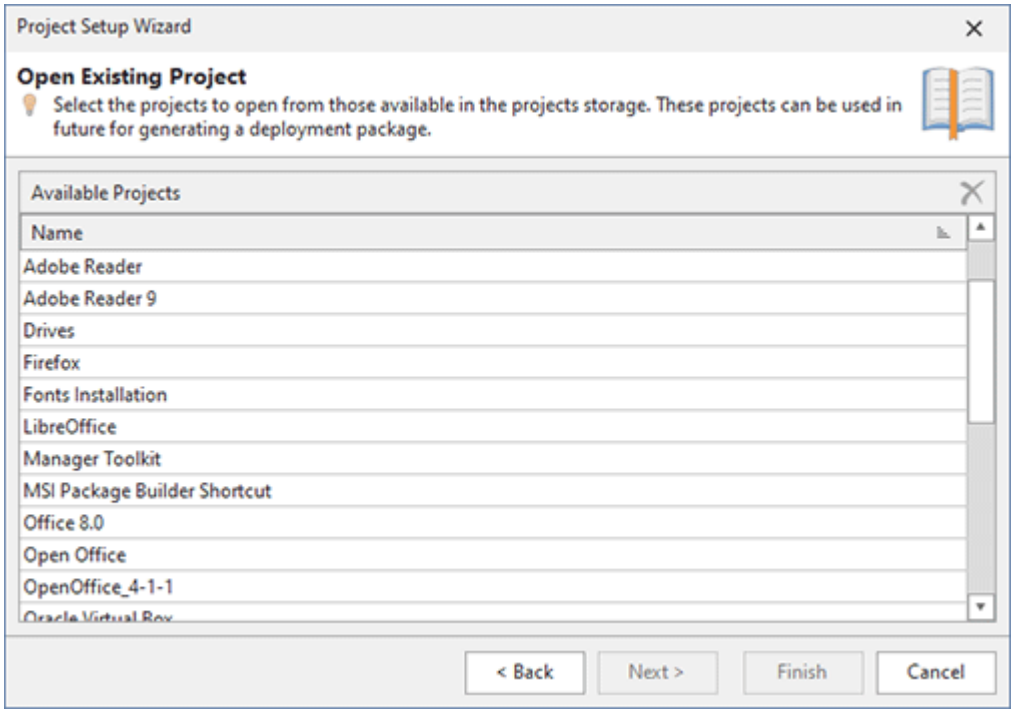
The **Open** button from the **Project Management** group on the **Home** Ribbon page allows you to open projects available in the projects storage.

When you close MSI Package Builder all the projects you were working with are automatically closed too. It is also possible to close projects manually – this will be described further. If you want to continue working with those projects that were closed, you should open them again.



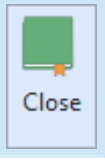
The links to the projects you have used recently are available in the **Application Menu** and in the **Recent Projects** group of the **Welcome Screen**. To open any of those projects, simply click on a link.

To open projects, choose the **Open Project** item from the **Projects** view pop-up menu or press the **Open** button from the **Project Management** group on the **Home** Ribbon page. Alternatively, you can click the **Open Project** link in the **Recent Projects** group on the **Welcome Screen**. The **Open Project** dialog will appear on the screen Pic 3. The same pane is also displayed if you choose the **Open Existing Projects** option in the **Project Setup** wizard.




**Pic 3. Opening existing projects**

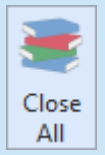
You can select both single and multiple projects in the **Available Projects** table to be opened at the same time. It is also possible to delete a project from the projects storage, if it is no longer needed by using the **Delete Projects** button from the toolbar.




**Close**

The **Close** button from the **Project Management** group on the **Home** Ribbon page should be used to close the selected projects, thus remove them from the Projects view.

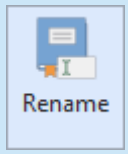
Copyright © 2001-2023 EMCO. All rights reserved.




**Close All**

The **Close All** button from the **Project Management** group on the **Home** ribbon page allows you to close all currently opened projects, thus remove them from the Projects view.

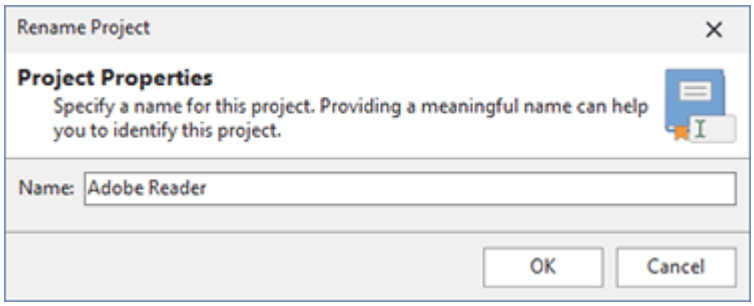
MSI Package Builder provides you with an ability of always keeping the **Projects** view in an actual state. You can always close the projects you are not going to work with right now and they will be removed from the view, but they still exist so you can reopen any of them to continue working with them anytime you want. To close the project select it in the **Projects** view and choose the **Close** item from the pop-up menu, or press the **Close** button from the **Project Management** group on the **Home** ribbon page. It is also possible to close all the currently opened projects at once using the **Close All** button.




**Rename**

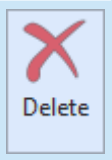
The **Rename** button from the **Project Management** group on the **Home** Ribbon page should be used to provide a new name for the selected project.

You can face a situation when the name you have given to a project is no longer actual or you have mistakenly provided a wrong name during the project creation. It is not a problem – with MSI Package Builder you can rename the project easily any time you want. Just double click the project, or select it and press the **Rename** button from the **Project Management** group on the **Home** Ribbon page. The **Rename Project** dialog will appear on the screen, where you can provide a new name for the project Pic 4.




**Pic 4. Renaming a project**

The important limitation for a project name is that it must be unique. Thus, it is not possible to rename the project, if the project with the same name as specified already exists (even if it is now closed).




**Delete**

The **Delete** button from the **Project Management** group on the **Home** Ribbon page allows you to delete the selected projects from the projects storage.

The projects created in MSI Package Builder can require much space on the hard drive, but it is always possible to delete the project from the storage if it is no longer needed. To delete the project select it in the **Projects** view and choose the **Delete** item from the pop-up menu or press the **Delete** button from the **Project Management** group on the **Home** Ribbon page.



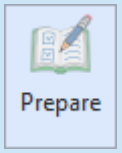
After you have deleted a project it is not possible to restore it any more. Make sure that the project is no longer needed before deleting it.

Now you are introduced to the project management features and should be able to organize your MSI Package Builder projects the way that will fit your needs best.

Copyright © 2001-2023 EMCO. All rights reserved.

## Projects Preparation

As the result of the monitoring file system, registry and other changes the MSI Package Builder project is created and filled with those changes. If the monitoring was performed via **Repackage Installation** wizard, all the preparation steps are performed automatically, so this chapter is interesting only to those users who are using advanced changes monitoring through the capturing abilities of MSI Package Builder.

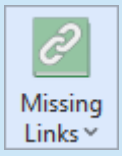



**Prepare**

The **Prepare** button from the **Links** group on the **Project** Ribbon page should be used to prepare the selected projects.

After monitoring for changes is completed, the files created and modified during the monitoring process are not available in the projects structure. The projects containing those files are marked with a red arrow overlay (), which means they are unprepared. All external files should be copied to the project structure before creating a deployment package. This is performed using the **Prepare** button from the **Links** group on the **Project** Ribbon page, when the project is selected.

As soon as the preparing operation is completed, you can see if it has actually succeeded by taking a look at the overlay arrow icon. If the arrow is green (), than the whole projects has been prepared successfully, and if it is yellow (), than there have been problems during the preparation process. In most cases, this means that some files were missing when the preparation was executed. Detailed information on errors is available in the **Log** view.

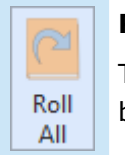



**Missing Links**

The **Missing Links** drop-down button from the **Links** group on the **Project** Ribbon page allows you to choose the method for resolving the problems with missing files in the selected projects. You can either remove those files, or try to check if they became available.

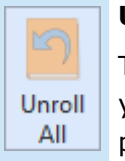
If the file was not accessible during the preparation process, thus not copied to the project storage, it is marked as missing. For the easier identification of the missing links a broken chain overlay icon ( ) is displayed in the left top corner of the items in the **File System** view. The problems with missing links should be resolved before creating a deployment package. There are two possible scenarios of solving the missing links problem: repair and remove.

Before choosing a resolution mode, you should analyze the problems. If the missing files are temporary or not required to be created by the installer during a deployment of a generated package, they can be painlessly removed. Otherwise, you should provide access to those files and repair the missing links. You should use the **Missing Links** drop-down button from the **Links** group on the **Project** Ribbon page to resolve problems with missing links.




**Roll All**

The **Roll All** button from the **System Folders** group on the **Project** Ribbon page should be used to replace all system folders with their definitions in the selected projects.




**Unroll All**

The **Unroll All** button from the **System Folders** group on the **Project** Ribbon page allows you to expand the system folder definitions to their absolute local paths in the selected projects.

Copyright © 2001-2023 EMCO. All rights reserved.

For a deployment package to be installed correctly on different operating systems with different environment configurations, it is required to replace all paths that represent the system folders (such as *My Documents*) with a special system folder object. This object will be expanded to the full path on the destination PC during the package deployment. You can perform this step for all folders in a project at once, using the **Roll All** button from the **System Folders** group on the **Project** Ribbon page.

After performing those steps, the packages are fully prepared to be used for creating a deployment package.

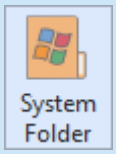
*Copyright © 2001-2023 EMCO. All rights reserved.*

## File System Modifications

The file system modifications are the changes to be performed by a generated package during its deployment to the file system on a target PC. Those changes can include files and folders creation or deletion, file content modification and shortcuts creation. The file system modifications are displayed in the **File System** view when the **File System** node of a project is selected in the **Projects** view. The changes performed during the monitoring process are added to the file system modifications automatically and can be edited only after the project is prepared. Let us take a closer look at configuring file system modifications.

### System Folders and Drives

Each file system modification is defined with a full path of the resource, so to add a file you should also define the whole file path. Each path is routed either with a logical drive or with a system folder object. The logical drive simply represents the drive letter, and the system folder object is used to define one of the available system folders. You can review a list of system folders in the **System Folder Definition Placeholders** section of this document.




**System Folder**

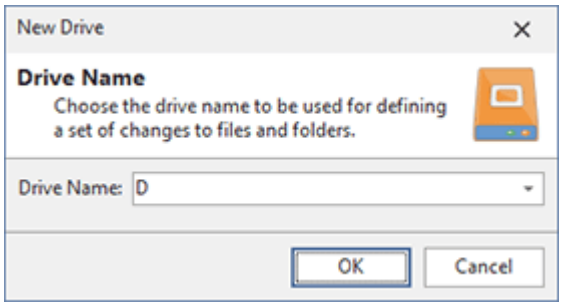
The **System Folder** button from the **New** group on the **Project** Ribbon page and on the contextual **File System** Ribbon page should be used to add a new system folder object to the currently configured project.




**Drive**

The **Drive** button from the **New** group on the **Project** Ribbon page and on the contextual **File System** Ribbon page should be used to add a new logical drive to the currently configured project.

To add a new logical drive definition, you should either choose the **New Drive** item from the pop-up menu, or click the **Drive** button from the **New** group on the regular **Project** and contextual **File System** Ribbon pages. Within the **New Drive** dialog, you are suggested to choose a drive letter Pic 1.

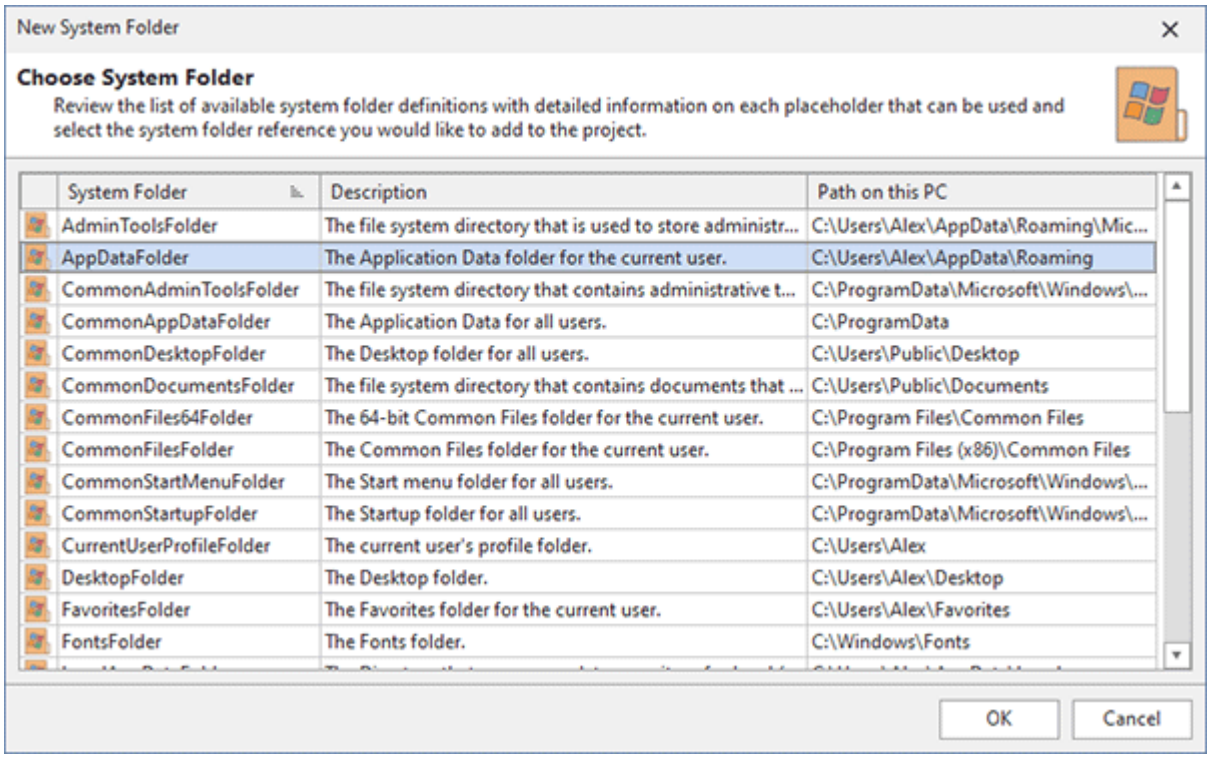



**Pic 1. Adding a logical drive definition to a project**

Copyright © 2001-2023 EMCO. All rights reserved.

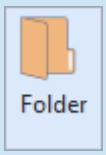
As soon as the drive is added, you can define modifications to files and folders on this drive.

Mostly the programs are installed to specific folders, and the shortcuts are created within the specific folders, that are operating system specific. To reach the same behavior you should use the system folders. To add a new system folder definition to the file system modifications, either choose the **New System Folder** menu item, or click the **System Folder** button from the **New** group on the regular **Project** and contextual **File System** Ribbon pages. Within the **New System Folder** dialog, you are suggested to choose a system folder to add to the project Pic 2.




**Pic 2. Adding a system folder to a project**

As soon as the system folder is added, you can define modifications to files and folders in this folder. Each system folder is expanded to a full path during a generated package deployment.



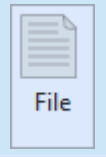

**Folder**

The **Folder** button from the **New** group on the contextual **File System** Ribbon page should be used to create a new folder modification within the selected path.




**Sync Folder**

The **Sync Folder** button from the **New** group on the contextual **File System** Ribbon page can be used to create a new folder, which content is synchronized with the selected folder on the file system.




**File**

The **File** button from the **New** group on the contextual **File System** Ribbon page should be used to create a new file modification within the selected path.




**Shortcut**

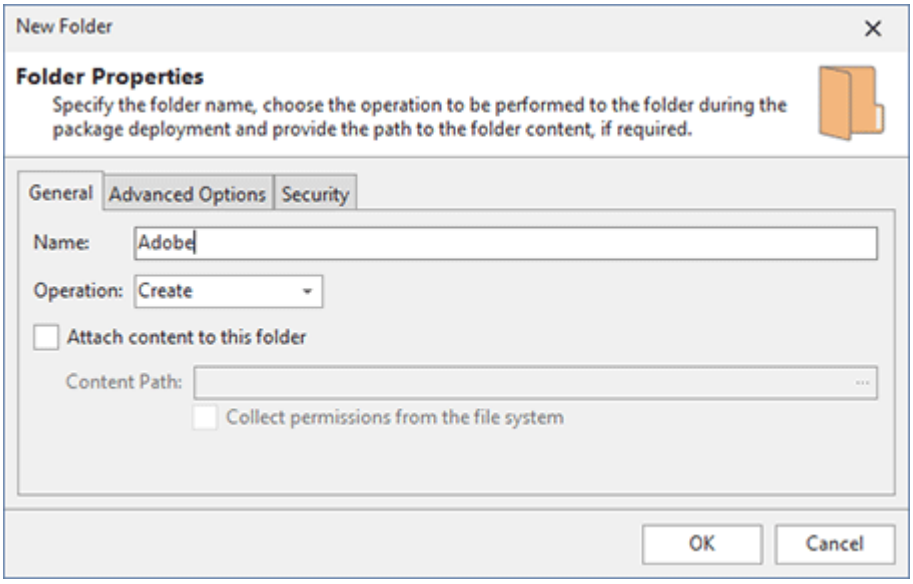
The **Shortcut** button from the **New** group on the contextual **File System** Ribbon page allows you to create a new shortcut to any target from this project in the selected path.

Copyright © 2001-2023 EMCO. All rights reserved.

MSI Package Builder allows you to define which files and folders should be created, modified or deleted during a deployment package installation.

## Folders

To add a new folder modification, select the drive, folder or system folder to add modification to, and either choose the **New > Folder** item from the pop-up menu, or press the **Folder** button from the **New** group on the contextual **File System** Ribbon page. The **New Folder** dialog will appear on the screen to configure the folder modification Pic 3.




**Pic 3. Adding a folder modification to a project**

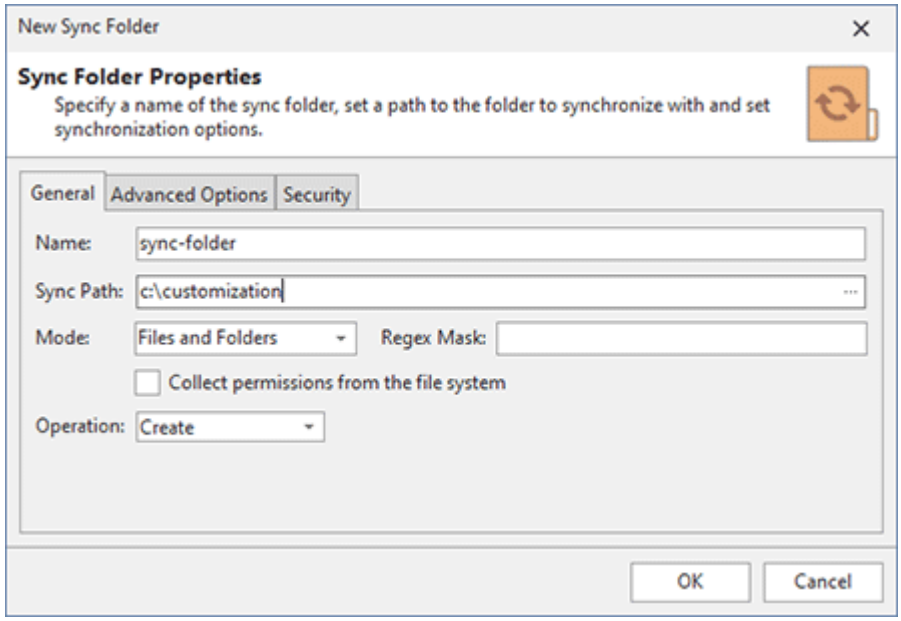
Copyright © 2001-2023 EMCO. All rights reserved.

While configuring a folder modification, you should specify a folder name and an operation to be performed over this folder during a deployment package installation. You can choose between the **Create**, **Modify** and **Delete** operations. For the folder creation and modification, you can attach content to this folder, using the file system import. Just check the **Attach content to this folder** option and provide a path to the folder to import its content. If you need to set file system attributes to be applied to the folder, you can configure these attributes on the **Advanced Options** tab. On the **Security** tab, you can specify the permissions to be explicitly assigned to this folder.

## Sync Folders

When you add a folder and prepare the project, the folder contents is copied into the project storage, as described in the **Project Preparation** chapter. The project storage is used to generate an output package. So when you add a regular folder, its content is copied to the storage when the project is prepared, but after preparation folder changes aren't reflected in the generated package.

A sync folder allows you to add a folder to the project and let the program use the folder content to generate a package. Any external changes in the folder contents will be reflected in the package generated. To add a new sync folder choose the **New > Sync Folder** item from the pop-up menu, or press the **Sync Folder** button from the **New** group on the contextual **File System** Ribbon page. The **New Sync Folder** dialog will appear on the screen to configure the folder modification Pic 4.




**Pic 4. Adding a sync folder modification to a project**

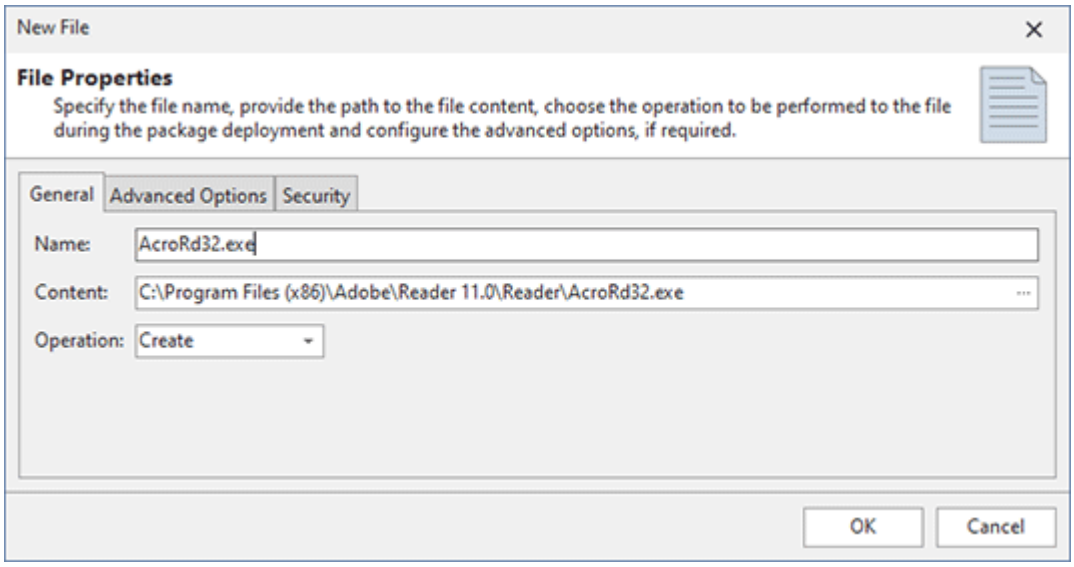
Copyright © 2001-2023 EMCO. All rights reserved.

A sync folder has the same parameters as a regular folder, but additionally you need to specify a **Sync Path** to the folder used to provide content to the generated package. The **Mode** option allows you to retrieve files only, or files and folders that are stored in the sync folder. The **Regex Mask** option allows you to specify a regular expression to retrieve folder entries. Only resources matching these options will be added to the generated package.

Once a sync folder is added to the project, you can see its content retrieved from the file system according with the configured options. If the sync folder content is changed on the file system, you can refresh it in the project by choosing the **Synchronize** option in the context menu for the sync folder.

## Files

To add a new file modification, select the drive, folder or system folder to add modification to, and either choose the **New > File** item from the pop-up menu, or press the **File** button from the **New** group on the contextual **File System** Ribbon page. The **New File** dialog will appear on the screen to configure the file modification Pic 5.




**Pic 5. Adding a file modification to a project**

While configuring a file modification, you should specify a file name and an operation to be performed over this file during a deployment package installation. You can choose between **Create**, **Modify** and **Delete** operations. For the file creation and modification, you should define file content in the **Content** field, providing a path to an existing file to get content from.

While specifying files and folders modifications you can provide a relative path to the **Name** field – the missing path elements will be automatically created in the file system modifications.

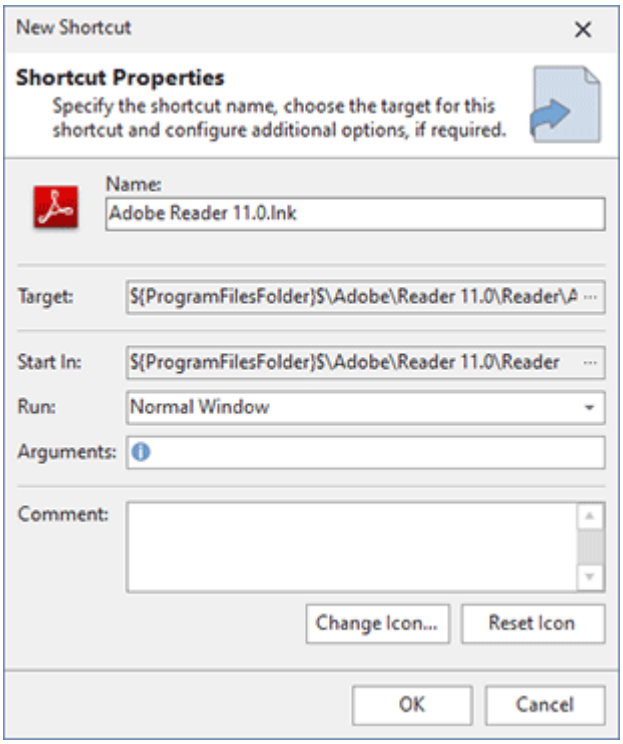
It is also possible to override file system attributes to be used for the file and specify if this file should be treated as persistent, thus not removed during a deployment package uninstall process. These options are available on the **Advanced Options** tab. On the **Security** tab, you can specify the permissions to be explicitly assigned to this file.

The advanced options are provided for the experienced users. Please do not modify these options until you are positively sure that this modification is required for the correct package deployment.

Copyright © 2001-2023 EMCO. All rights reserved.

## Shortcuts

MSI Package Builder allows you to create shortcuts to any item defined in the file system modifications during a deployment package installation. To add a shortcut definition, select the path you would like to create shortcut in and choose the **New > Shortcut** item from the pop-up menu. Alternatively, you can use the **Shortcut** button from the **New** group on the contextual **File System** ribbon page. The **New Shortcut** dialog will appear on the screen to let you configure the shortcut Pic 6.




**Pic 6. Adding a shortcut to a project**

While adding a shortcut, you should provide a name for the shortcut to be created to the **Name** field and choose the shortcut target in the **Target** field. Optionally you can configure a set of additional shortcut settings. You can choose a working directory for a shortcut, choose a required window state, provide command-line arguments to be passed to the shortcut target and define a comment for the shortcut. The **Arguments** field supports the property definition placeholders, so you can use the standard MSI properties while specifying the arguments. See the Property Definition Placeholders section of this document for the list of available placeholders.

## Modifying File System Resources

The file system modifications can be changed, deleted and transferred to different location.




**Edit**

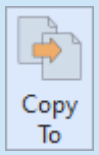
The **Edit** button from the **Management** group on the contextual **File System** Ribbon page should be used to change the selected file system modification.




**Delete**

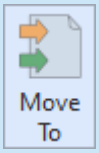
The **Delete** button from the **Management** group on the contextual **File System** Ribbon page allows you to delete the selected file system modification from the project.

Copyright © 2001-2023 EMCO. All rights reserved.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **File System** Ribbon page should be used to copy the selected file system modifications to another location.



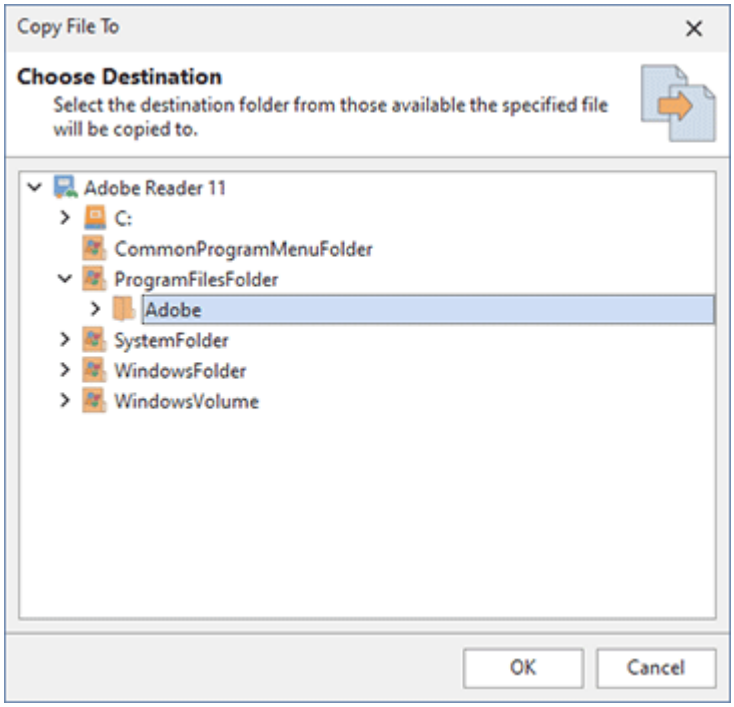

**Move To**

The **Move To** button from the **Management** group on the contextual **File System** Ribbon page allows you to move the selected file system modifications to another location.

To change the modification, select it in the **File System** view and either choose the **Edit** item in the pop-up menu or click the **Edit** button on the contextual **File System** Ribbon page. While editing a file system modification, it is possible to configure the same options as during its creation. To delete specific modifications from the project, select them and choose the **Delete** item from the pop-up menu or click the **Delete** button on the contextual **File System** Ribbon page.

If the project contains some files or folders that you don't want to include into the generated package, you can use the context menu option **Exclude from Build**. Excluded resources are displayed in the project using the faded icons. Use the context menu option **Include in Build** for excluded resource to add them into the generated package. The **Exclude from Build** option allows you to make experiments with the set of resources included into the resulted package with no deleting resources from the project.

The well-known drag/drop and copy/paste techniques are fully supported for copying and moving file system modifications within a single project and between projects. You can also use the **Copy To** and **Move To** buttons from the **Management** group on the contextual **File System** Ribbon page to transfer the selected file system modifications to a different location. While using these buttons, you are suggested to select a target location in a dialog and confirm your selection Pic 7.




**Pic 7. Copying file system modifications**

Copyright © 2001-2023 EMCO. All rights reserved.

When a container (drive, system folder or folder) is transferred to another location, all files and folders within the container are transferred as well. Also all files and folders within a container are deleted irreparably when a container is deleted, so be careful while transferring and deleting file system modifications.

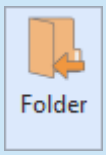
## Importing Existing Files and Folders

MSI Package Builder allows you to ease the file system modifications definition by using an ability of importing the local file system objects to the project. You can import both files and folders with or without their contents. While importing file system elements, a folder structure is created and files contents are attached automatically. To perform import, you can choose the **Import > Folder** or **Import > File** items from the pop-up menu. Alternatively, you can use the **File System**, **Folder** and **File** buttons from the **Import** groups on the regular **Project** and contextual **File System** Ribbon pages.



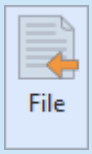

**File System**

The **File System** drop-down button from the **Import** group on the **Project** Ribbon page should be used to add files and folders from the file system to the selected project.




**Folder**

The **Folder** button from the **Import** group on the contextual **File System** Ribbon page allows you to add a folder from the file system with or without its content to the selected project.




**File**

The **File** button from the **Import** group on the contextual **File System** Ribbon page allows you to add a file from the file system to the selected project.

While importing a folder, you are suggested to provide the path to the folder to be added to the selected project and choose if the folder content should also be imported. You can import files and folders from the specified folder, only files from the specified folder or ignore the folder content. As for importing a file, you should provide the path to the file to be added to the selected project. The file content will be attached automatically.

## Rolling and Unrolling System Folders

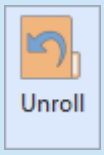
If a deployed file system resource should be stored in one of the Windows system folders, a path to this resource can be different on different machines. To create installations that can be deployed successfully on different machines, it's recommended to replace absolute deployment paths by system folders in the project. When such an installation is deployed, system folders are automatically resolved to absolute paths suitable for the system, where the installation is deployed. When you use monitoring the program automatically rolls system folders by replacing system paths, but you can also roll system folder manually for a project or for a specific folder.




**Roll**

The **Roll** button from the **System Folders** group on the contextual **File System** Ribbon page should be used to replace the selected system folders with their definitions in the selected project.

Copyright © 2001-2023 EMCO. All rights reserved.




**Unroll**

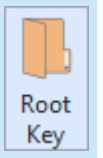
The **Unroll** button from the **System Folders** group on the contextual **File System** Ribbon page allows you to expand the selected system folders to their absolute local paths in the selected project.

Together with rolling and unrolling all system folders in a project, that is a recommended way, it is possible to roll and unroll only the specific folders. To roll a folder select it and choose the **Roll** item from the pop-up menu, and to perform unroll you should choose the **Unroll** item. Alternatively, you can use the **Roll** and **Unroll** buttons from the **System Folders** group on the contextual **File System** Ribbon page. Rolling and unrolling only specific folders is an advanced feature of MSI Package Builder and should be used very carefully.

Now you are introduced to the features used to manage the file system modifications with MSI Package Builder and should be able to define the modifications to be included into a deployment package without any misunderstanding.

## Registry Modifications

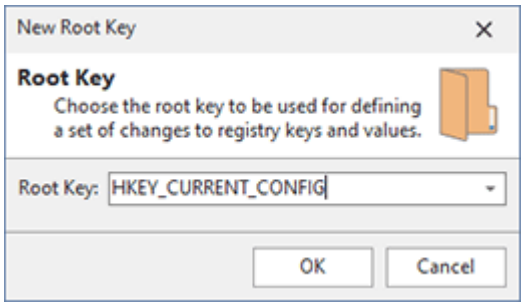
The registry modifications are the changes to be performed by a generated package during its deployment to the registry on a target PC. Those changes can include keys and values creation or deletion and values modification. The registry modifications are displayed in the **Registry** view when the **Registry** node of a project is selected in the **Projects** view. The changes performed during the monitoring process are added to the registry modifications automatically. Let us take a close look at configuring registry modifications.




**Root Key**

The **Root Key** button from the **New** group on the **Project** Ribbon page and on the contextual **Registry** Ribbon page should be used to add a new root key to the currently configured project.

Each registry modification is defined with a full path of the resource, so to add a value you should also define the complete key path. Each path is routed with a root registry key, also known as a registry hive. To add a new root key, you should either choose the **New Root Key** item from the pop-up menu, or click the **Root Key** button from the **New** group on the regular **Project** and contextual **Registry** Ribbon pages. Within the **New Root Key** dialog, you are suggested to choose a registry hive Pic 1.




**Pic 1. Adding a root key to a project**

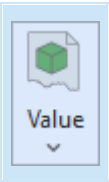
As soon as the root key is added, you can define modifications to keys and values in this key.




**Key**

The **Key** button from the **New** group on the contextual **Registry** Ribbon page should be used to create a new registry key modification within the selected registry key.

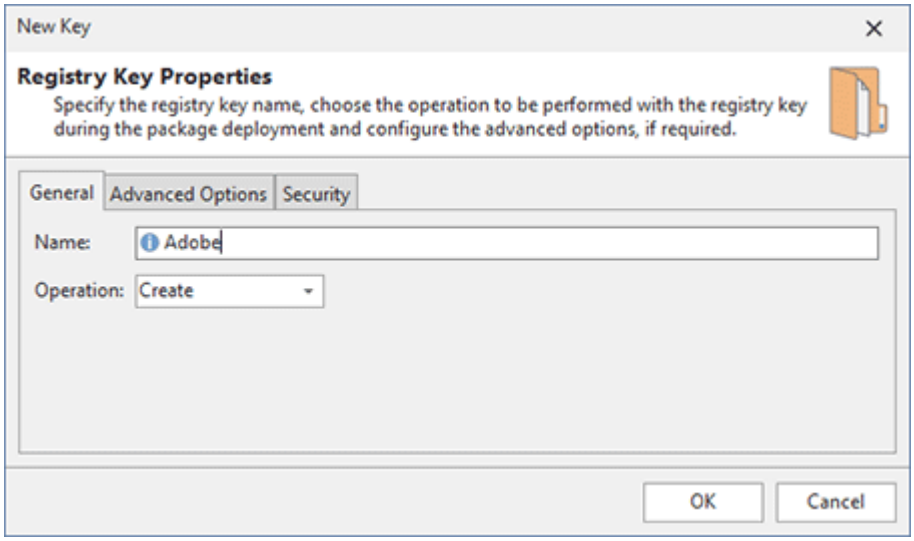
Copyright © 2001-2023 EMCO. All rights reserved.




**Value**

The **Value** drop-down button from the **New** group on the contextual **Registry** Ribbon page should be used to create a new registry value modification within the selected registry key.

MSI Package Builder allows you to define which registry keys and values should be created, modified or deleted during a deployment package installation. To add a new registry key modification, select the root key or key to add modification to, and either choose the **New > Key** item from the pop-up menu, or press the **Key** button from the **New** group on the contextual **Registry** Ribbon page. The **New Key** dialog will appear on the screen to configure the registry key modification Pic 2.




**Pic 2. Adding a registry key modification to a project**

While configuring a registry key modification, you should specify a key name and an operation to be performed over this key during a deployment package installation. You can choose between the **Create**, **Modify** and **Delete** operations.

While specifying registry modifications you can provide a path to the **Name** field, and the missing path elements will be created in the registry modifications automatically.

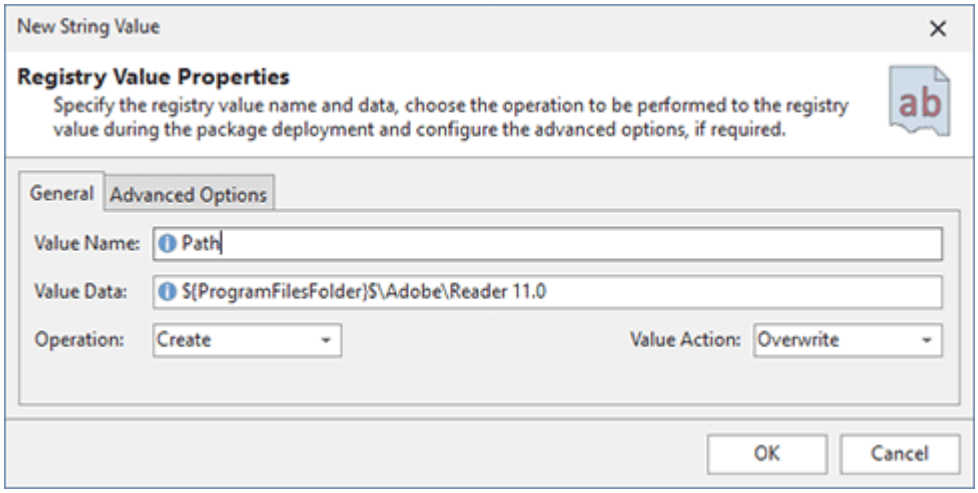
The **Name** field supports the property definition placeholders, so you can use the standard MSI properties while specifying the arguments. See the Property Definition Placeholders section of this document for the list of available placeholders.

It is also possible to specify if the registry key should be treated as persistent, thus not removed during a deployment package uninstall process, on the **Advanced Options** tab. On the **Security** tab, you can specify the permissions to be explicitly assigned to this key.

The advanced options are provided for the experienced users. Please do not modify these options until you are positively sure that this modification is required for the correct package deployment.

Copyright © 2001-2023 EMCO. All rights reserved.

To add a new registry value modification, select the key you want to add modification to and choose the appropriate value type from the **New** item in the pop-up menu or from the drop-down of the **Value** button from the **New** group on the contextual **Registry** Ribbon page. The **New Value** dialog will appear on the screen Pic 3. This dialog is almost the same for all value types; it differs only with a value data editor.




**Pic 3. Adding a registry value modification to a project**

While configuring a registry value modification, you should specify a value name and an operation to be performed over this value during a package deployment. You can choose between **Create**, **Modify** and **Delete** operations. For the value creation and modification, you can define a data to be written to this value. For the string, expandable string and multi-string values, in case if the value already exists, you can choose if it should be overridden by the specified data or the specified data should be appended/prepended to the existing.

To define a modification to a default registry value, you can select it and choose the **Edit** item from the pop-up menu. If you want to edit it as a type, different from REG_SZ, you should use the Change Type item from the pop-up menu.

It is also possible to specify if the registry value should be treated as persistent, thus not removed during a deployment package uninstall process, within the **Advanced Options** group.

The advanced options are provided for experienced users. Please do not modify these options until you are positively sure that this modification is required for the correct package deployment.

The **Value Name** and **Value Data** fields support the property definition placeholders, so you can use the standard MSI properties while specifying the arguments. See the Property Definition Placeholders section of this document for the list of available placeholders.




**Edit**

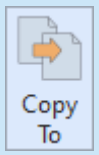
The **Edit** button from the **Management** group on the contextual **Registry** Ribbon page should be used to change the selected registry modification.




**Delete**

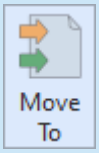
The **Delete** button from the **Management** group on the contextual **Registry** Ribbon page allows you to delete the selected registry modification from the selected project.

Copyright © 2001-2023 EMCO. All rights reserved.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **Registry** Ribbon page should be used to copy the selected registry modifications to another location.



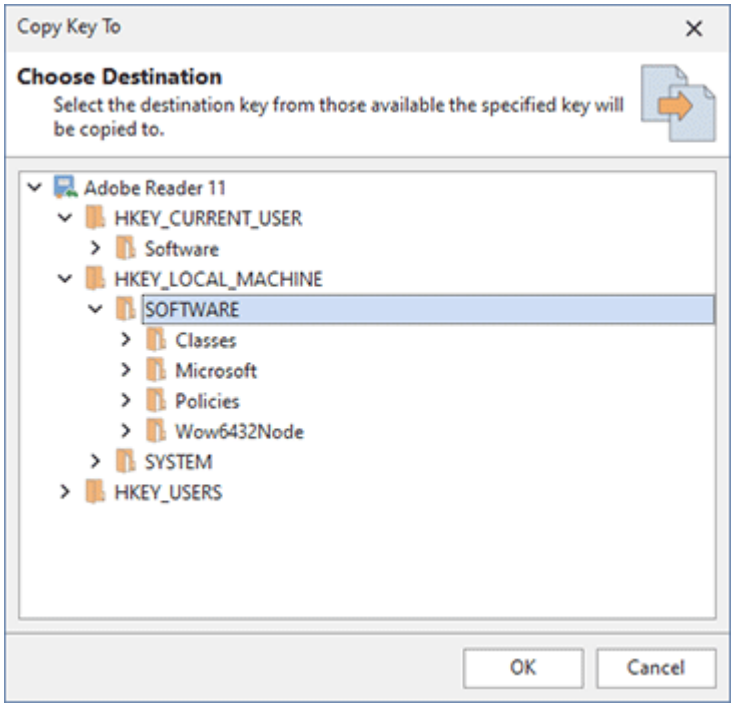

**Move To**

The **Move To** button from the **Management** group on the contextual **Registry** Ribbon page allows you to move the selected registry modifications to another location.

The registry modifications can be changed, deleted and transferred to different location. To change the modification, select it in the **Registry** view and either choose the **Edit** item in the pop-up menu or click the **Edit** button on the contextual **Registry** Ribbon page. While editing a registry modification, it is possible to configure the same options as during its creation. To delete specific modifications from the project, select them and choose the **Delete** item from the pop-up menu or click the **Delete** button on the contextual **Registry** Ribbon page.

If the project contains some registry entries that you don't want to include into the generated package, you can use the context menu option **Exclude from Build**. Excluded resources are displayed in the project using the faded icons. Use the context menu option **Include in Build** for excluded resource to add them into the generated package. The **Exclude from Build** option allows you to make experiments with the set of resources included into the resulted package with no deleting resources from the project.

The well-known drag/drop and copy/paste techniques are fully supported for copying and moving registry modifications within a single project and between projects. You can also use the **Copy To** and **Move To** buttons from the **Management** group on the contextual **Registry** Ribbon page to transfer the selected registry modifications to a different location. While using these buttons, you are suggested to select a target location in a dialog and confirm your selection Pic 4.



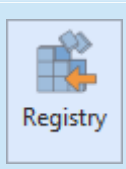

**Pic 4. Copying registry modifications**

Copyright © 2001-2023 EMCO. All rights reserved.

When a registry key is transferred to another location, all keys and values within the key are transferred as well. Also all keys and values within a registry key are deleted irreparably when a registry key is deleted, so be careful while transferring and deleting registry keys.

## Importing Changes from Registration Entries (.reg) File

MSI Package Builder allows you to ease the registry modifications definition by using an ability of importing changes from a registration entries (.reg) file. To perform import, you can choose the **Import** item from the pop-up menu. Alternatively, you can use the **Registry** button from the **Import** groups on the regular **Project** and contextual **Registry** Ribbon pages.




**Registry**

The **Registry** button from the **Import** group on the regular **Project** and contextual **Registry** Ribbon pages allows you to add the changes defined in the specific registration entries (.reg) file to the selected project.

While importing changes, you are suggested to provide the path to the file to use as a definition for the changes to be included into a project.

Now you are introduced to the features used to manage the registry modifications with MSI Package Builder and should be able to define the modifications to be included into a deployment package without any misunderstanding.

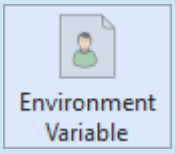
*Copyright © 2001-2023 EMCO. All rights reserved.*

## Environment Variables Modifications

With MSI Package Builder you can easily add to a project any system and user environment variables so that they will be registered in or unregistered from the operating system when the deployment package is installed on the computer.

MSIX/AppX packages do not support configuring the environment variables during deployment, so, when generated, those changes are included into MSI and App-V deployment packages only.

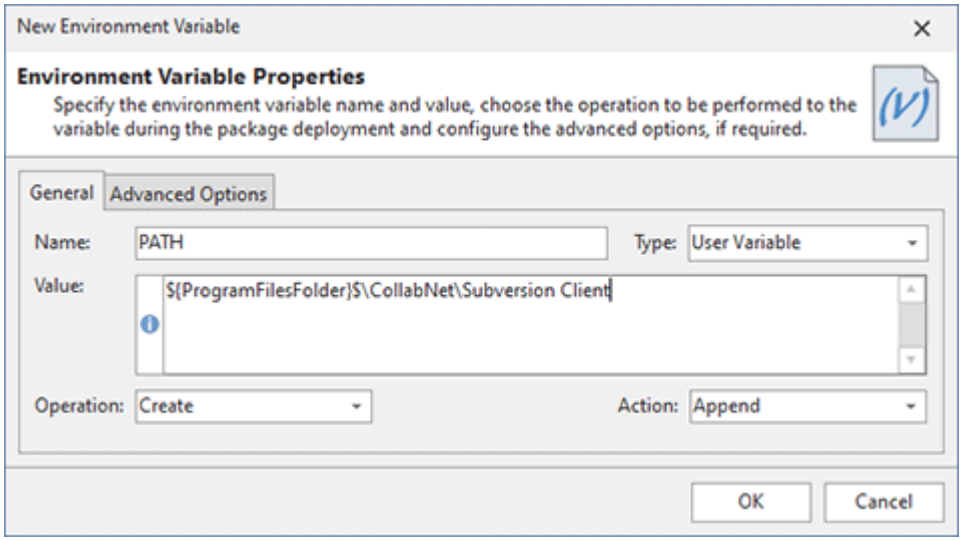
The environment variables modifications are displayed in the **Environment Variables** view when the **Environment Variables** node of a project is selected in the **Projects** view. Let us take a close look at configuring environment variables modifications.




**Environment Variable**

The **Environment Variable** button from the **New** group on the regular **Project** and contextual **Environment Variables** Ribbon pages should be used to add a new environment variable modification to the currently configured project.

MSI Package Builder allows you to add both modifications to system and user environment variables. To add a modification choose the **New Environment Variable** item from the pop-up menu or click the **Environment Variable** button from the **New** group on regular **Project** and contextual **Environment Variables** Ribbon pages. The **New Environment Variable** dialog will appear on the screen Pic 1.




**Pic 1. Configuring an environment variable modification**

Copyright © 2001-2023 EMCO. All rights reserved.

During the configuration process, you are suggested to specify the environment variable name and the operation to be performed over this variable during a deployment package installation. You can choose between the **Create**, **Create if does not exist**, **Remove** and **Remove if value equals** operations. For creating an environment variable you should provide a variable value to the **Value** field. If variable already exists during a generated package installation, you can choose if its value should be overridden with the defined one, or the defined value should be appended/prepended to the existing one. The **System Variable** checkbox is used to identify if you are going to modify system or user environment variable.

It is also possible to specify if the environment variable should be treated as persistent, thus not removed during a deployment package uninstall process, within the **Advanced Options** group.

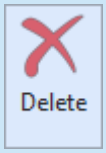
The advanced options are provided for the experienced users. Please do not modify these options until you are positively sure that this modification is required for the correct package deployment.

The **Value** field supports the property definition placeholders, so you can use the standard MSI properties while specifying the value for an environment variable. See the Property Definition Placeholders section of this document for the list of available placeholders.



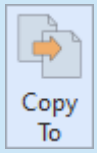

**Edit**

The **Edit** button from the **Management** group on the contextual **Environment Variables** Ribbon page should be used to change the selected environment variables modification.



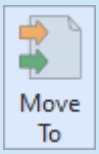

**Delete**

The **Delete** button from the **Management** group on the contextual **Environment Variables** Ribbon page allows you to delete the selected environment variables modifications from the selected project.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **Environment Variables** Ribbon page should be used to copy the selected environment variables modifications to another location.




**Move To**

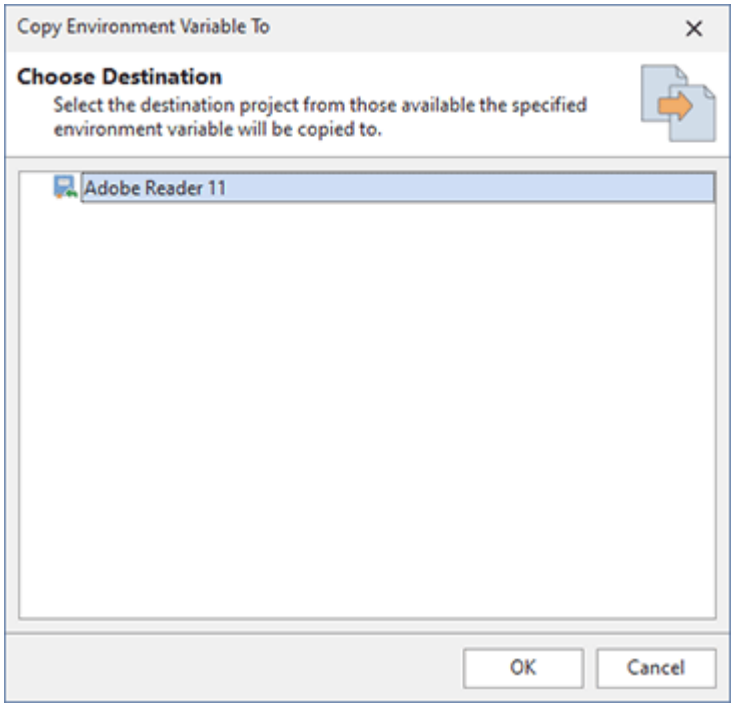
The **Move To** button from the **Management** group on the contextual **Environment Variables** Ribbon page allows you to move the selected environment variables modifications to another location.

The environment variables modifications can be changed, deleted and transferred to different location. To change the modification, select it in the **Environment Variables** view and double-click it. Alternatively, you can either choose the **Edit** item in the pop-up menu or click the **Edit** button on the contextual **Environment Variables** Ribbon page. While editing an environment variables modification, it is possible to configure the same options as during its creation. To delete specific modifications from the project, select them and choose the **Delete** item from the pop-up menu or click the **Delete** button on the contextual **Environment Variables** Ribbon page.

If the project contains some environment variables that you don't want to include into the generated package, you can use the context menu option **Exclude from Build.** Excluded resources are displayed in the project using the faded icons. Use the context menu option **Include in Build** for excluded resource to add them into the generated package. The **Exclude from Build** option allows you to make experiments with the set of resources included into the resulted package with no deleting resources from the project.

Copyright © 2001-2023 EMCO. All rights reserved.

The well-known drag/drop and copy/paste techniques are fully supported for copying and moving environment variables modifications between projects. You can also use the **Copy To** and **Move To** buttons from the **Management** group on the contextual **Environment Variables** Ribbon page to transfer the selected environment variables modifications to a different location. While using these buttons, you are suggested to select a target location in a dialog and confirm your selection Pic 2.




**Pic 2. Copying environment variables modifications**

Now you are introduced to the features used to manage the environment variables modifications with MSI Package Builder and should be able to define the user and system variables modifications to be included into a deployment package without any misunderstanding.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Services Modifications

The services modifications are the changes to be performed by a generated package during its deployment to the installed services on a target PC. Those changes can include services creation, deletion and control.

Configuring services during deployment is not supported by App-V and MSIX/AppX packages, so, when generated, those changes are included into the MSI deployment packages only.

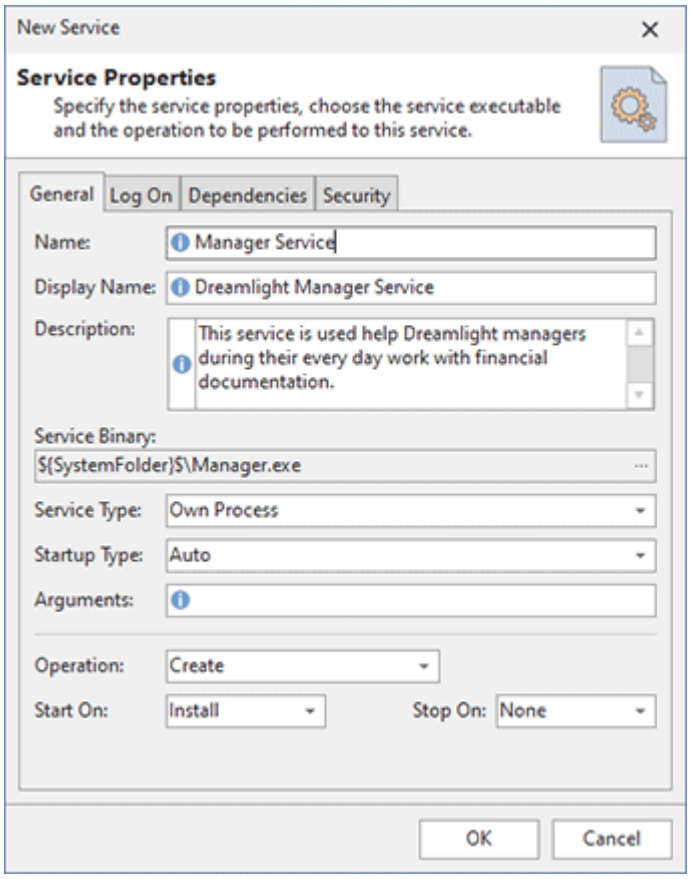
The services modifications are displayed in the **Services** view when the **Services** node of a project is selected in the **Projects** view. The changes performed during the monitoring process are added to the services modifications automatically. Let us take a close look at configuring service modifications.




**Service**

The **Service** button from the **New** group on the regular **Project** and contextual **Services** Ribbon pages should be used to add a new service modification to the currently configured project.

To add a new service modification, you should either choose the **New Service** item from the pop-up menu, or press the **Service** button from the **New** group on the regular **Project** and contextual **Services** Ribbon pages. Within the **New Service** dialog, you are suggested to configure the service modification in detail Pic 1.




**Pic 1. Adding a service to be created and started during a deployment package installation**

Copyright © 2001-2023 EMCO. All rights reserved.

While adding a service modification, you should first specify the service name and the operation to be performed over the service. You can choose between the **Create**, **Delete**, **Control** and **Restart** operations. The scope of the settings available for configuring depends on the selected operation. Let us describe each operation in detail.

The **Create** operation is used if the service is a part of the installation. So, it is created together with the generated deployment package installation. For the **Create** operation, the scope of options is the widest. Together with the service name, used to install the service, you must also provide a display name used for the service and a path to the file from the current project to be used as the service executable on the **General** tab. It is also recommended that a service description be provided, which is displayed in a services section of the computer management console for each installed service. The other options available for service creation on the **General** tab are the execution options; those are the service type, startup type and the arguments to be passed to the service on startup.

On the **Log On** tab for the service being created, you can choose it should log on as a local system, or as a specific user. For the local system account, it is possible to define if the service should interact with desktop. On this tab, you can also configure an error control level used for this service. You can choose from the following:

- **Normal** - the startup program logs the error in the event log but continues the startup operation.
- **Critical** - 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** - the startup program ignores the error and continues the startup operation.

The **Dependencies** tab is also used when configuring the service creation and allows defining the services this service depends on. The service name is provided to the **Dependency** field and you can add, edit and remove dependencies on-line using the corresponding buttons built into this field.

On the **Security** tab, you can specify the permissions to be explicitly assigned to this service.

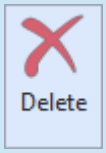
While specifying a service name, display name, description, startup arguments and dependencies you can use the property definition placeholders, standing for the standard MSI properties. See the Property Definition Placeholders section of this document for the list of available placeholders.

For the other operations, except **Create**, the **Log On**, **Dependencies** and **Security** tabs are not used and only the **Name** field from the top part of the **General** tab is used. As for the **Delete** operation, the service name is the only field to be specified. The service with the matching name will be deleted from a target PC during a generated deployment package installation. For the **Control** operation, after providing a service name, you should define triggers to start and stop the service on. You can choose between **Install**, **Uninstall**, **Both** and **None** options to start and stop the service on. As for the **Restart** operation, you should choose if it should be restarted on install, uninstall or both events.




**Edit**

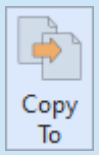
The **Edit** button from the **Management** group on the contextual **Services** Ribbon page should be used to change the selected services modification.




**Delete**

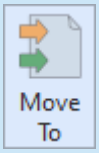
The **Delete** button from the **Management** group on the contextual **Services** Ribbon page allows you to delete the selected services modifications from the selected project.

Copyright © 2001-2023 EMCO. All rights reserved.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **Services** Ribbon page should be used to copy the selected services modifications to another location.



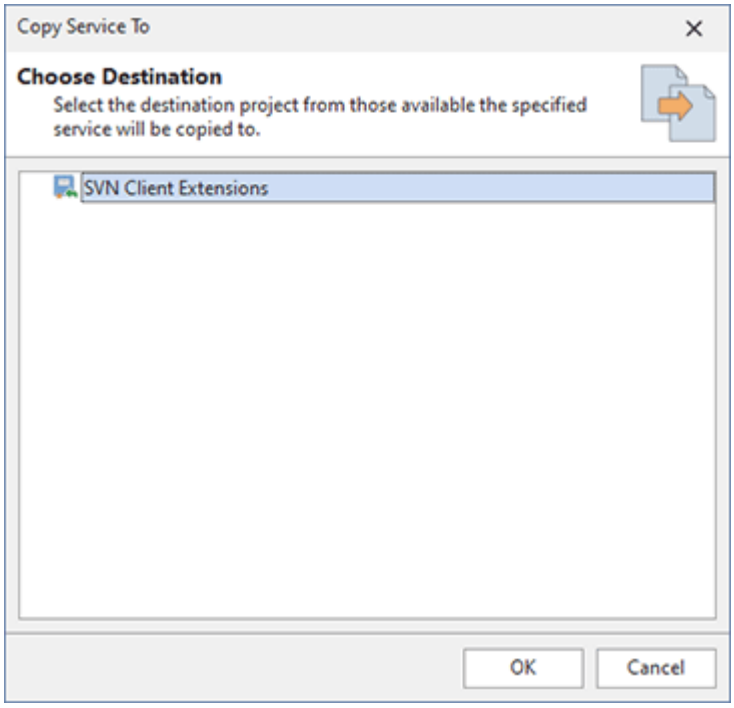

**Move To**

The **Move To** button from the **Management** group on the contextual **Services** Ribbon page allows you to move the selected services modifications to another location.

The services modifications can be changed, deleted and transferred to different location. To change the modification, select it in the **Services** view and double-click it. Alternatively, you can either choose the **Edit** item in the pop-up menu or click the **Edit** button on the contextual **Services** Ribbon page. While editing a service modification, it is possible to configure the same options as during its creation. To delete specific modifications from the project, select them and choose the **Delete** item from the pop-up menu or click the **Delete** button on the contextual **Services** Ribbon page.

If the project contains some services that you don't want to include into the generated package, you can use the context menu option **Exclude from Build**. Excluded resources are displayed in the project using the faded icons. Use the context menu option **Include in Build** for excluded resource to add them into the generated package. The **Exclude from Build** option allows you to make experiments with the set of resources included into the resulted package with no deleting resources from the project.

The well-known drag/drop and copy/paste techniques are fully supported for copying and moving services modifications between projects. You can also use the **Copy To** and **Move To** buttons from the **Management** group on the contextual **Services** Ribbon page to transfer the selected services modifications to a different location. While using these buttons, you are suggested to select a target location in a dialog and confirm your selection Pic 2.




**Pic 2. Copying services modifications**

Copyright © 2001-2023 EMCO. All rights reserved.

Now you are introduced to the features used to manage the services modifications with MSI Package Builder and should be able to create and control services by a generated deployment package without any misunderstanding.

Copyright © 2001-2023 EMCO. All rights reserved.

## Side-by-side Assemblies Deployment

MSI Package Builder provides you with an ability to deploy shared libraries. A shared library is a file that is intended to be shared by executable files and further shared objects files. Modules used by a program are loaded from individual shared objects into memory at load time or run time. Side-by-side assemblies can safely share assemblies among multiple applications and can offset the negative effects of assembly sharing. Such effects, known as 'DLL hell', include version conflicts, missing DLLs, duplicate DLLs, and incorrect or missing registration. Instead of a single version of an assembly that assumes backward compatibility with all applications, side-by-side assembly sharing enables multiple versions of an assembly to run simultaneously on the system. As for .NET assemblies, the side-by-side concept is implemented with a help of the *Global Assembly Cache*. It case of Win32 assemblies, in side-by-side, Windows stores multiple versions of a DLL in the *WinSxS* folder of the Windows directory. SxS is also the technological basis for registration-free COM activation.

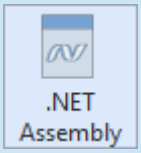
MSIX/AppX packages do not support side-by-side assemblies deployment, so, when generated, those changes are included into MSI and App-V deployment packages only.

For the side-by-side assemblies to be deployed, they are specified on a project level and displayed in the **Assemblies** view when the **Assemblies** node of a project is selected in the **Projects** view. The assemblies installed during the monitoring process are added to the project automatically and can be edited only after the projects containing those changes are prepared. Let us take a closer look at configuring side-by-side assemblies.




**Win32 Assembly**

The **Win32 Assembly** button from the **New** group on the **Project** Ribbon page and on the contextual **Assemblies** Ribbon page should be used to add a new Win32 assembly to the currently configured project.

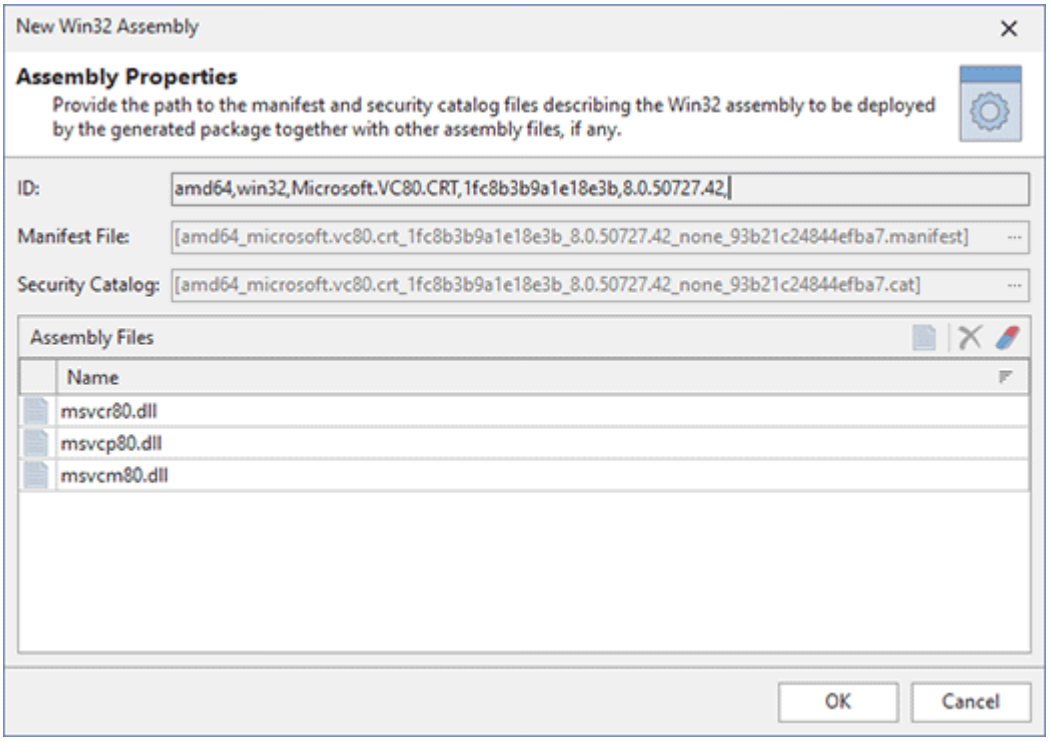



**.NET Assembly**

The **.NET Assembly** button from the **New** group on the **Project** Ribbon page and on the contextual **Assemblies** Ribbon page should be used to add a new .NET assembly to the currently configured project.

Copyright © 2001-2023 EMCO. All rights reserved.

To add a new assembly to a project, you can choose the **New Win32 Assembly** and **New .NET Assembly** items from the **Assemblies** view pop-up menu. Alternatively, you can use the **Win32 Assembly** and **.NET Assembly** button from the **New** group on the **Project** and **Assemblies** Ribbon pages. The dialog will appear on the screen to let you configure the assembly to be added Pic 1.




**Pic 1. Configuring a Win32 Assembly**

While configuring a Win32 assembly you should provide a path to the assembly manifest and security catalog files. If it is required, you should also add assembly files to the **Assembly Files** table. As for a .NET assembly, it is required that a path to the assembly file be provided. Optionally, it is possible to add configuration files, required for the assembly installation, to the **Assembly Files** table.



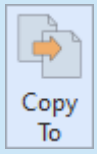

**Edit**

The **Edit** button from the **Management** group on the contextual **Assemblies** Ribbon page should be used to change the selected assembly.



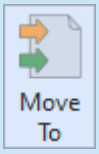

**Delete**

The **Delete** button from the **Management** group on the contextual **Assemblies** Ribbon page allows you to delete the selected assemblies from the selected project.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **Assemblies** Ribbon page should be used to copy the selected assemblies to another project.




**Move To**

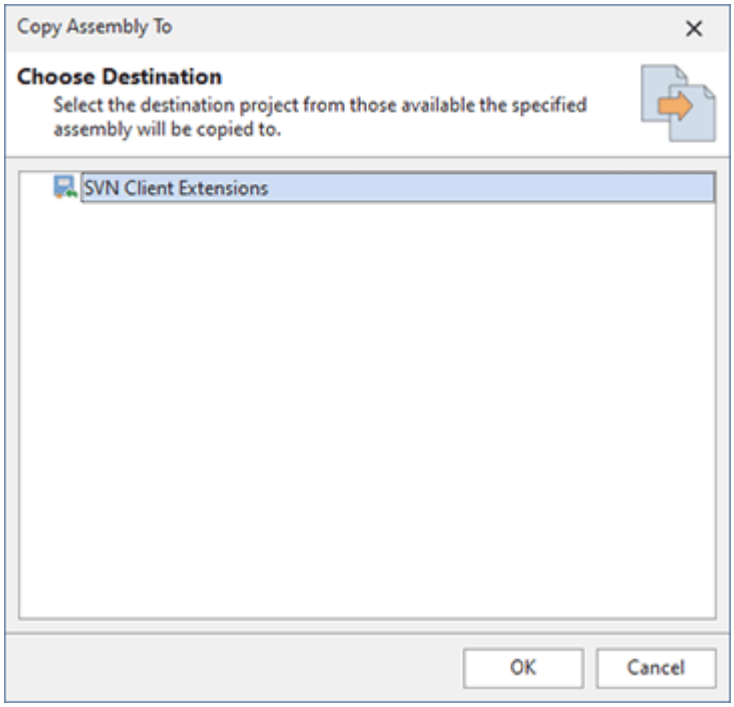
The **Move To** button from the **Management** group on the contextual **Assemblies** Ribbon page allows you to move the selected assemblies to another project.

Copyright © 2001-2023 EMCO. All rights reserved.

The side-by-side assemblies can be changed, deleted and transferred between projects. To change the assembly, select it and either double-click, or choose the **Edit** item in pop-up menu or on the contextual **Assemblies** Ribbon page. While editing an assembly, you can define the same scope of properties as during its creation. To delete the assembly from the project, select it and choose the **Delete** item in the pop-up menu or on the contextual **Assemblies** Ribbon page.

If the project contains some assemblies that you don't want to include into the generated package, you can use the context menu option **Exclude from Build**. Excluded resources are displayed in the project using the faded icons. Use the context menu option **Include in Build** for excluded resource to add them into the generated package. The **Exclude from Build** option allows you to make experiments with the set of resources included into the resulted package with no deleting resources from the project.

The well-known drag/drop and copy/paste techniques are fully supported for copying and moving assemblies between projects. You can also use the **Copy To** and **Move To** buttons from the **Management** group on the contextual **Assemblies** Ribbon page to transfer the selected assemblies to a different project. While using these buttons, you are suggested to select a project from those available in a dialog and confirm your selection Pic 2.




**Pic 2. Copying assemblies between projects**

Now you are introduced to the side-by-side assemblies concept and an ability to deploy such assemblies with MSI Package Builder, thus you should be able to define the assemblies to be installed by a deployment package without any misunderstanding.

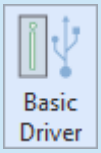
*Copyright © 2001-2023 EMCO. All rights reserved.*

## Drivers Deployment

MSI Package Builder is capable of deploying hardware and software drivers. It is possible to create or delete basic drivers and install or pre-install driver packages in scope of an MSI package installation.

App-V and MSIX/AppX packages do not support configuring the device drivers during deployment, so, when generated, those changes are included into MSI deployment packages only.

Both basic drivers and drivers from packages can be registered as device class filters by MSI Package Builder, if required. Changes to the operating system drivers are displayed in the **Drivers** view when the **Drivers** node of a project is selected in the **Projects** view. Changes performed during the monitoring process are added automatically to the drivers deployment actions. Let us take a closer look at configuring the drivers deployment actions.




**Basic Driver**

The **Basic Driver** button from the **New** group on the regular **Project** and contextual **Drivers** Ribbon pages should be used to add a new basic driver deployment action to the currently configured project.




**Driver Package**

The **Driver Package** button from the **New** group on the regular **Project** and contextual **Drivers** Ribbon pages should be used to add a new driver package to the currently configured project to be either installed or pre-installed by the resulting MSI package.



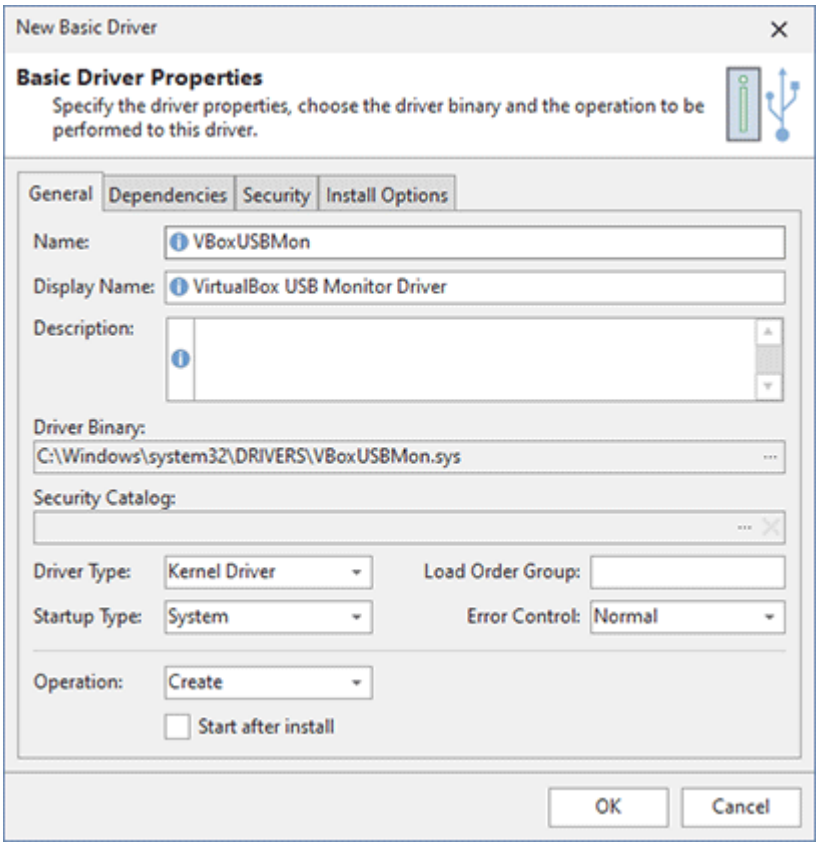

**DIFx Driver Package**

The **DIFx Driver Package** button from the **New** group on the regular **Project** and contextual **Drivers** Ribbon pages should be used to add a new **Driver Installation Framework** package to the currently configured project to be either installed or pre-installed by a resulting deployment package.

Copyright © 2001-2023 EMCO. All rights reserved.

The basic driver is a simple driver that is shipped as a single dynamic link library (usually as the *.sys file) and is managed like a windows service. A driver package consists of all the software components that you must supply for your device to be supported under Windows. The following components are necessary to install and support a device on a Windows operating system: the driver file (*.sys), the installation files (*.inf and *.cat) and other files (the device installation application, the device icon, the device property pages, etc).

To add a new basic driver deployment action, you should either choose the **New Basic Driver** item from the pop-up menu or press the **Basic Driver** button from the **New** group on the regular **Project** and contextual **Drivers** Ribbon pages. Within the **New Basic Driver** dialog, you are suggested to configure the basic driver deployment action in detail Pic 1.




**Pic 1. Adding a basic driver to be created and started during an MSI package installation**

Copyright © 2001-2023 EMCO. All rights reserved.

While adding a basic driver deployment action, you should first specify the driver name and the operation to be performed on the driver. You can choose between the **Create** and **Delete** operations. The scope of settings available for configuring depends on the selected operation. Let us describe each operation in detail.

The **Create** operation is used if the driver is a part of an installation. Thus, it is created together with the generated MSI package installation. Along with the driver name used to install the driver, you should also provide the display name used for the driver and the path to the file from the current project to be used as the driver binary on the **General** tab. You can also provide the driver security catalog file, if required. It is also recommended that the driver description be provided, which is displayed in the computer management console for each driver. Other options displayed on the **General** tab are the advanced driver loading settings, which are the driver type, the startup type, the load order group and the error control.

The **Driver Type** field is used to choose the type of the basic driver to be installed: either a kernel driver or a system driver.

As for the **Startup Type**, the following values can be used:

- **Boot** - the device driver required to start the computer;
- **System** - the non-boot-start driver that detects device(s) that are not PnP-enumerable;
- **Auto** - the Non-PnP driver that must be started by the service control manager;
- **On Demand** - the PnP driver;
- **Disabled** - the disabled driver.

The **Load Order Group** is used to define the group this driver should be added to for loading by the operating system.

You can choose from the following **Error Control** levels:

- **Normal** - the startup program logs error in the event log but continues the startup operation.
- **Critical** - the startup program logs 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** - the startup program ignores error and continues the startup operation.

*Copyright © 2001-2023 EMCO. All rights reserved.*

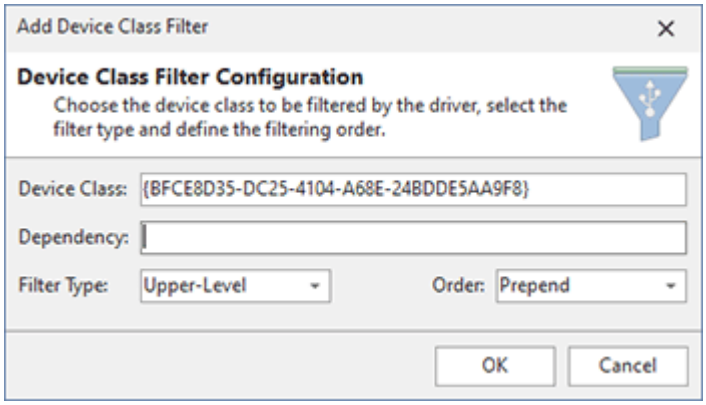
The **Dependencies** tab is also used while configuring the driver creation. It allows defining the drivers this driver depends on. The driver name is provided to the **Dependency** field, and you can add, edit and remove dependencies on-line using the corresponding buttons built into this field.

On the **Security** tab, you can specify the permissions to be explicitly assigned to this driver.

While specifying the driver name, display name, description and dependencies you can use the property definition placeholders standing for the standard MSI properties. See the Property Definition Placeholders section of this document for the list of available placeholders.

For the **Delete** operation, the **Dependencies** tab is not used and only the **Name** field from the **General** tab is applicable. The driver with the matching name will be deleted from the target PC while installing the generated MSI package.

If you would like to register the basic driver as a filter for one or more device classes, you should enable the **Register this driver as a device class filter** option on the **Install Options** tab. The **Install Options** tab contains a list of device class filters the configured driver is used as. You can add and remove the filter configuration entries using the corresponding buttons on the toolbar. Let us take a closer look at the device class filter configuration Pic 2.

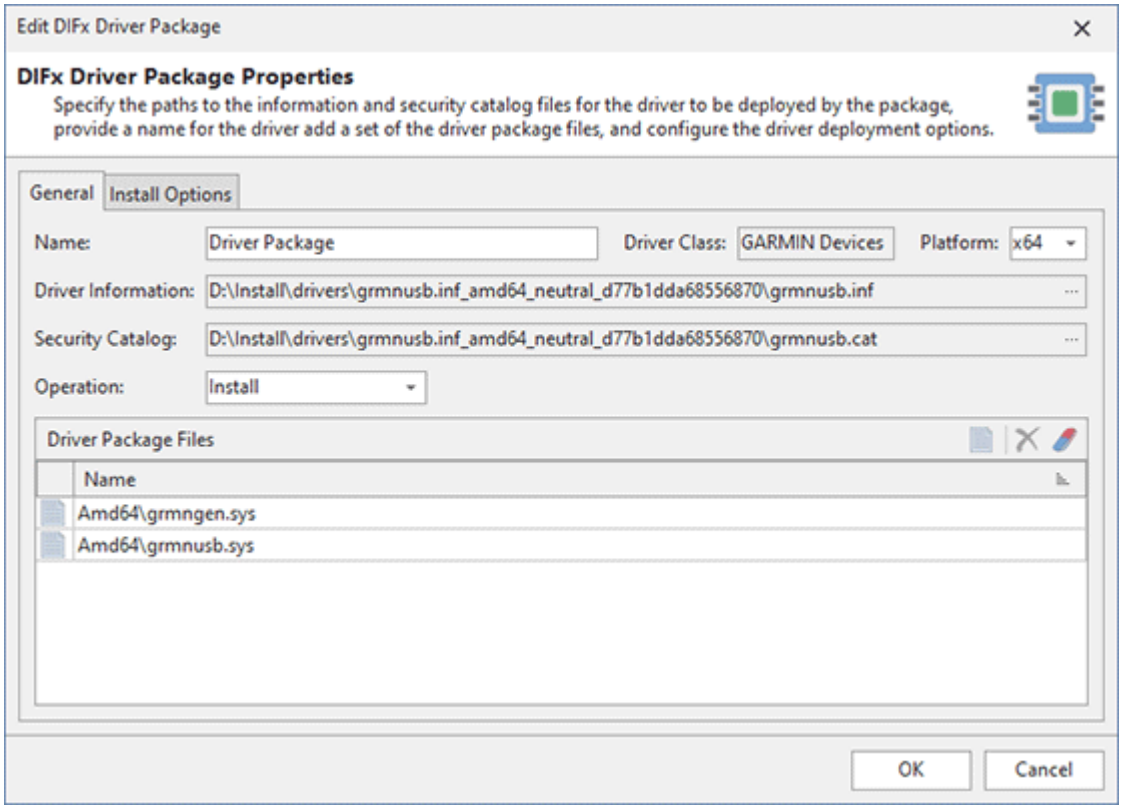



**Pic 2. Registering a driver as an upper-level device class filter**

Copyright © 2001-2023 EMCO. All rights reserved.

When registering a driver as a device class filter, you should provide the device class to register the driver to into the **Device Class** field. Within the **Filter Type** drop-down, you should choose the type of the filter to be registered. A **Lower-Level** filter driver monitors and/or modifies I/O requests to a particular device. Typically, such filters redefine the hardware behavior to match the expected specifications. For example, a lower-level class filter driver for mouse devices could provide acceleration performing a nonlinear conversion of the mouse movement data. An **Upper-Level** filter driver adds value to a particular device. For example, an upper-level device filter driver for a keyboard could enforce additional security checks. The **Dependency** and **Order** fields are responsible for the filter driver position in the scope of all filters for the specified device class.

To add a new driver package deployment action, you should either choose the **New Driver Package** item from the pop-up menu or press the **Driver Package** button from the **New** group on the regular **Project** and contextual **Drivers** Ribbon pages. If you are going to deploy the package that requires deployment via the DIFx (Driver Installation Framework), use the **New DIFx Driver Package** menu item and the **DIFx Driver Package** button respectively. The file system browser will be opened to let you choose the driver information file for the package. As soon as it is chosen, you will see the dialog where you are suggested to configure the driver package deployment action in detail. Let us use the **New Driver Package** one as the example Pic 3.




**Pic 3. Adding a driver package to be pre-installed during an MSI package installation**

*Copyright © 2001-2023 EMCO. All rights reserved.*

Common information on the driver package being configured is displayed on the **General** tab. As you can see, the **Driver Information** and **Security Catalog** fields, as well as the **Driver Package Files** table, are filled automatically with the files provided by the selected installation file. You can change any of these aspects in the future, if required.

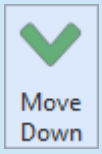
The next step is providing the driver name and configuring the deployment. The name is provided to the **Name** field. The next step is to specify the target platform in the **Platform** field. The platform can be specified only during the driver package creation. As for the deployment options, you can choose if the driver should be installed or pre-installed within the **Operation** drop-down. The **Install** operation should be chosen if the device with an appropriate ID is already present in the system or it is a virtual device created by the installer. The **Pre-Install** operation allows you to install a package to the driver storage. The operating system will use the installed driver as soon as an appropriate device is installed to the system.

The content of the **Install Options** tab differs depending on the type of the driver package. It may be possible to choose the devices or sections from the package to be installed, the hardware component ID and/or register a driver from the configured package as a filter for one or more device classes.




**Move Up**

The **Move Up** button from the **Order** group on the contextual **Drivers** ribbon page should be used to move the selected drivers up the deployment order.




**Move Down**

The **Move Down** button from the **Order** group on the contextual **Drivers** ribbon page should be used to move the selected drivers down the deployment order.

By default, the drivers deployment order is the same as the addition order, but you can reorder the drivers using the **Move Up** and **Move Down** items from the pop-up menu or the **Order** group on the contextual **Drivers** Ribbon page.



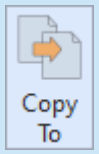

**Edit**

The **Edit** button from the **Management** group on the contextual **Drivers** Ribbon page should be used to change the selected driver deployment action.



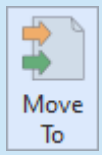

**Delete**

The **Delete** button from the **Management** group on the contextual **Drivers** Ribbon page allows you to delete the selected driver deployment actions from the selected project.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **Drivers** Ribbon page should be used to copy the selected driver deployment actions to another location.




**Move To**

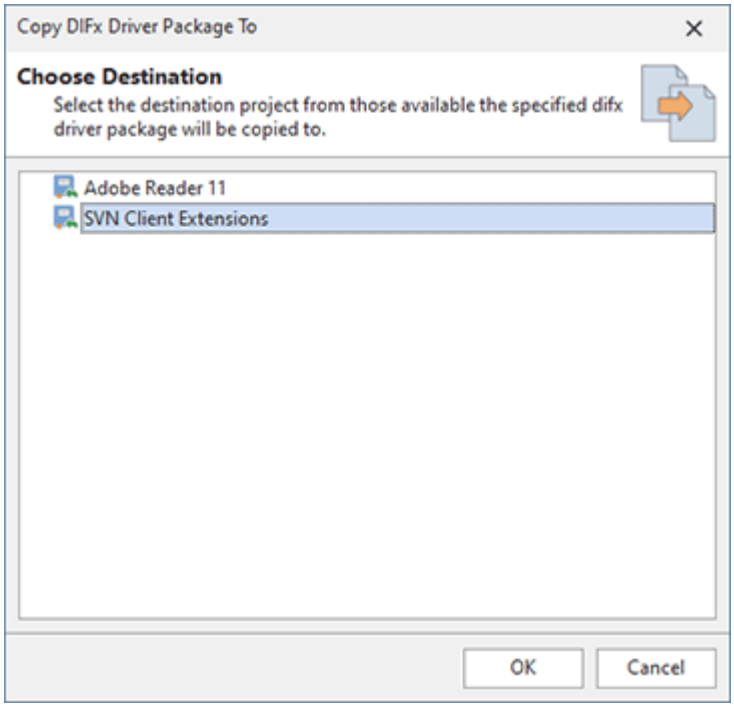
The **Move To** button from the **Management** group on the contextual **Drivers** Ribbon page allows you to move the selected driver deployment actions to another location.

Copyright © 2001-2023 EMCO. All rights reserved.

Driver deployment actions can be changed, deleted or transferred another location. To change the driver deployment action, select it in the **Drivers** view and double-click it. Alternatively, you can either choose the **Edit** item in the pop-up menu or click the **Edit** button on the contextual **Drivers** Ribbon page. While editing a driver deployment action, it is possible to configure the same options as those available during the action creation. To delete specific actions from a project, select them and choose the **Delete** item from the pop-up menu or click the **Delete** button on the contextual **Drivers** Ribbon page.

If the project contains some drivers that you don't want to include into the generated package, you can use the context menu option **Exclude from Build**. Excluded resources are displayed in the project using the faded icons. Use the context menu option **Include in Build** for excluded resource to add them into the generated package. The **Exclude from Build** option allows you to make experiments with the set of resources included into the resulted package with no deleting resources from the project.

The well-known drag/drop and copy/paste techniques are fully supported for copying and moving driver deployment actions between projects. You can also use the **Copy To** and **Move To** buttons from the **Management** group on the contextual **Drivers** Ribbon page to transfer the selected driver deployment actions to another location. While using these buttons, you are suggested to select the target location in the dialog and confirm your selection Pic 4.




**Pic 4. Copying driver deployment actions**

Now that you have been introduced to the features used to manage driver deployment actions with MSI Package Builder, you should be able to deploy drivers by the generated MSI package with full understanding.

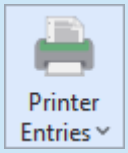
*Copyright © 2001-2023 EMCO. All rights reserved.*

## Printers Deployment

MSI Package Builder comes with an option of performing changes to the printing system configuration. It is possible to create, delete and modify printers; install and uninstall printer drivers; create and delete print processors and monitors; create, modify and delete printer ports, both local and TCP/IP. All those actions will be performed within the scope of the generated MSI package installation if defined in the project.

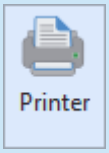
App-V and MSIX/AppX packages do not support configuring the printing system during deployment, so, when generated, those changes are included into MSI deployment packages only.

The set of changes to the printing system described in the project is displayed in the **Printers** view when the **Printers** node is selected in the **Projects** view. The changes performed during the monitoring process are automatically added to the printing system configuration actions. Let us take a closer look at the possible types of actions.



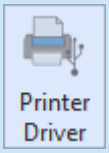

**Printer Entries**

The **Printer Entries** drop-down button from the **New** group on the **Project** Ribbon page can be used to create modifications to the printing system to be performed by the resulting MSI package



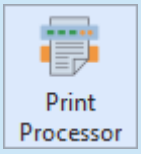

**Printer**

The **Printer** button from the **New** group on the **Printers** contextual Ribbon page should be used to add a new deployment action for creating, modifying or deleting a printer during the package deployment.



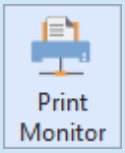

**Printer Driver**

The **Printer Driver** button from the **New** group on the **Printers** contextual Ribbon page should be used to add a new deployment action for installing or removing a printer driver during the package deployment.




**Print Processor**

The **Print Processor** button from the **New** group on the **Printers** contextual Ribbon page should be used to add a new deployment action for adding or removing a print processor during the package deployment.




**Print Monitor**

The **Print Monitor** button from the **New** group on the **Printers** contextual Ribbon page should be used to add a new deployment action for adding or removing a print monitor (either a language monitor or a port monitor) during the package deployment.

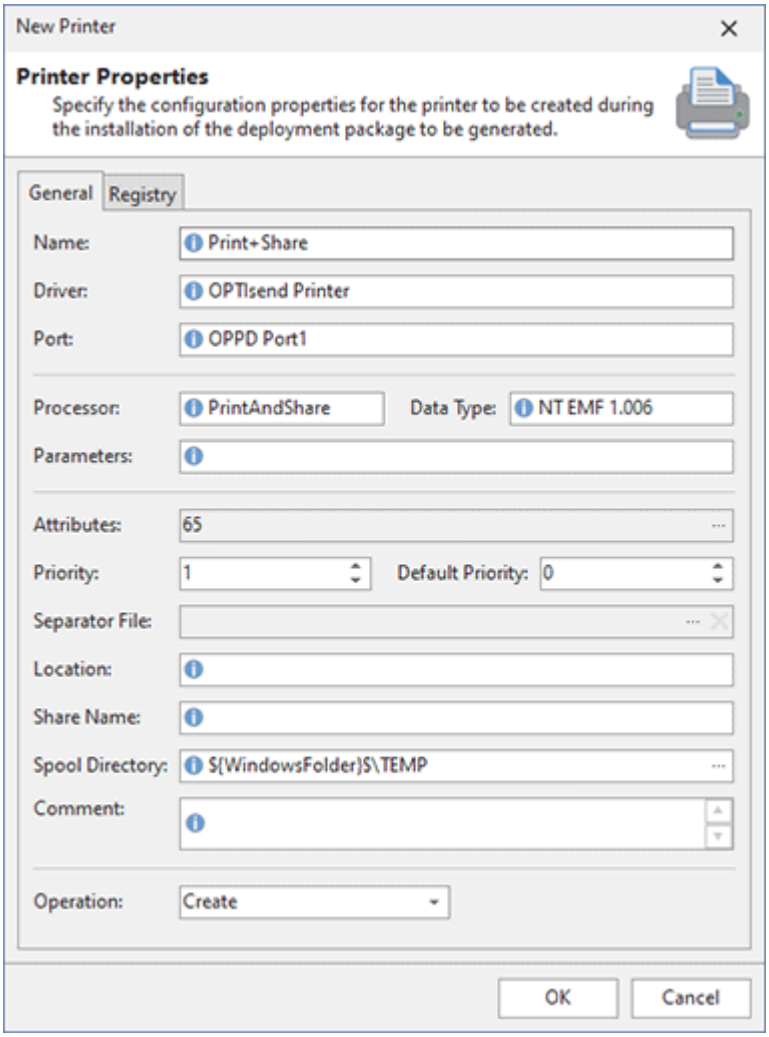



**Printer Port**

The **Printer Port** button from the **New** group on the **Printers** contextual Ribbon page should be used to add a new deployment action for creating, modifying or deleting a printer port during the package deployment.

Copyright © 2001-2023 EMCO. All rights reserved.

Let us start with adding a printer deployment action. As it has already been mentioned, it is possible to create, modify and delete printers during a package deployment. To add a printer deployment action, you should use the **Printer** item either from the **Printers** view pop-up menu or from the **New** group on the **Printers** contextual Ribbon page. Alternatively you can use the **Printer Entries** drop-down button located on the **Project** Ribbon page in the **New** group. The **New Printer** dialog will appear on the screen to let you configure the required deployment action Pic 1.




**Pic 1. Adding a printer to be created during installation of the resulting package**

*Copyright © 2001-2023 EMCO. All rights reserved.*

The common printer deployment action configuration is represented on the **General** tab of the dialog. If you would like the resulting MSI package to delete one of the existing printers, you should choose the **Delete** value in the **Operation** drop-down list and provide the name of the printer to be deleted to the **Name** field. For creation and modification of printers, there is a set of other fields to be specified beside the name. Along with the name, it is required to choose a driver to be used by the printer device and a port for the printer to reside on in the **Driver** and **Port** fields respectively. Two more required fields are **Processor** and **Data Type**. The **Processor** field should contain the processor to be used by the printer. The processor modifies the job as required for it to be printed properly. We will take a closer look at print processors later in this section of the documentation. The processor parameters can be provided to the **Parameters** field. The bitwise attributes mask can be specified by means of an easy-to-use editor using the ellipses button of the **Attributes** field. The current and default priorities of printers are specified in the **Priority** and **Default Priority** fields respectively. The page separator file is chosen from those already available in the project using the ellipses button of the **Separator** field. You can also specify the printer location, the share name, the custom spooler directory and a comment for the printer in the corresponding fields of this dialog.

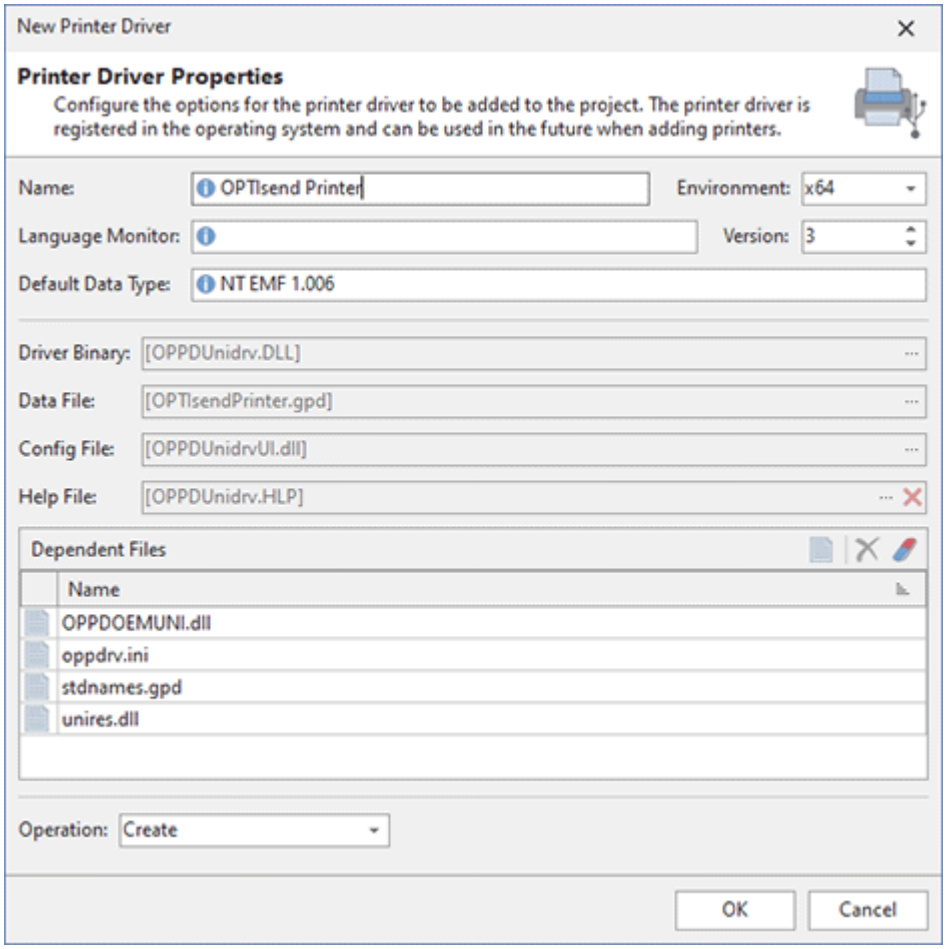
On the **Registry** tab of the configuration dialog it is possible to configure the changes applied to printer-specific sections of Windows registry while creating or modifying the printer during the MSI package deployment process.

If you want to make a specific printer as default after installing it, you can select it in the modifications list and choose the **Set as Default Printer** item from the **Printers** view pop-up menu.

The next printer system layer available for configuration is the printer drivers. Printer drivers contain information specific to the printer begin used. Printer drivers reside on user's computers and are used by the GDI to render print jobs. A printer driver is a software program that understands how to communicate with printers and plotters. Printer drivers translate information a user sends from his/her computer into commands that a printer understands. A printer driver sends printer-setting data, including the specifications needed to produce each character of the document, to the GDI. It also transmits helper services or utilities required to make the output print correctly. Each printer driver consists of the driver binary, data type and configuration files, an optional help file and dependent files. MSI Package Builder allows you to install and/or uninstall printer drivers during the resulting package deployment.

To add a deployment action to be performed for a printer driver, you should use the **Printer Driver** item either from the **Printers** view pop-up menu or from the **New** group on the **Printers** contextual Ribbon page. Alternatively you can use the **Printer Entries** drop-down button located on the **Project** Ribbon page in the **New** group. The **New Printer Driver** dialog will appear on the screen to let you configure the required deployment action Pic 2.

Copyright © 2001-2023 EMCO. All rights reserved.



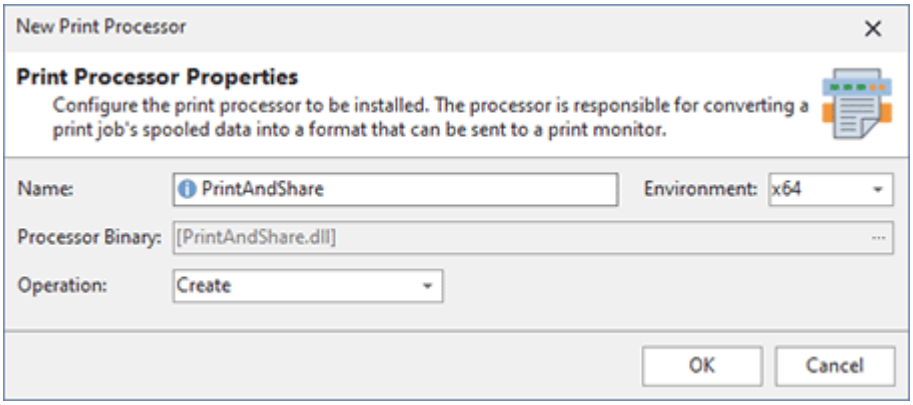

**Pic 2. Adding a printer driver to be installed with the resulting package**

*Copyright © 2001-2023 EMCO. All rights reserved.*

The **Name** and **Environment** fields are used to provide a driver name and specify the target environment. These fields must be filled for both **Create** and **Delete** operations. As for the **Create** operation, you should also define the print monitor for the driver, the driver version, the default data type and driver files as described above.

MSI Package Builder allows you to manage print processors when deploying the resulting package. Print processors are user-mode DLLs that are responsible for converting a print job's spooled data into a format that can be sent to a print monitor. They are also responsible for handling application requests to pause, resume and cancel print jobs. You can add and delete print processors during a package deployment.

To add a print processor to be managed during a package deployment, you should use the **Print Processor** item either from the **Printers** view pop-up menu or from the **New** group on the **Printers** contextual Ribbon page. Alternatively you can use the **Printer Entries** drop-down button located on the **Project** Ribbon page in the **New** group. The **New Print Processor** dialog will appear on the screen to let you configure the required deployment action Pic 3.



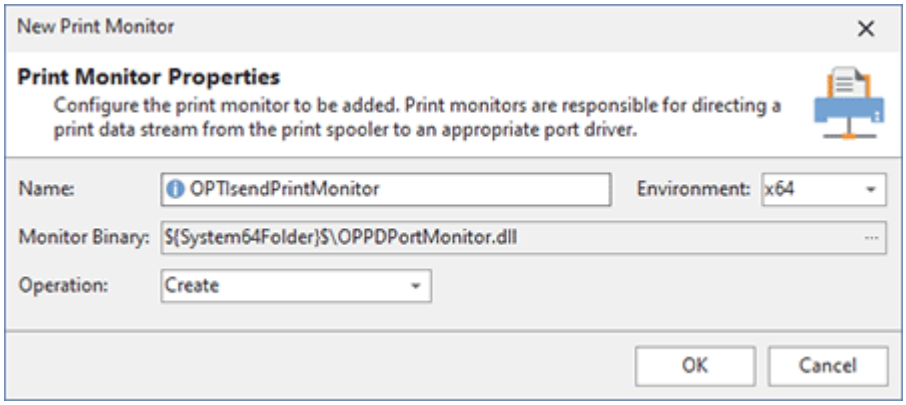

**Pic 3. Adding a print processor**

Copyright © 2001-2023 EMCO. All rights reserved.

Within the dialog, you can define a name for the print processor to be managed and the target environment. If you are going to create a processor, it is also required to provide a path to the processing library in the **Processor Binary** field.

With MSI Package Builder, it is also possible to manage print monitors. Print monitors are responsible for directing a print data stream from the print spooler to an appropriate port driver. Two types of print monitors are defined: language monitors and port monitors. Both can be created or deleted with a package generated with MSI Package Builder.

To add a print monitor entry, you should use the **Print Monitor** item either from the **Printers** view pop-up menu or from the **New** group on the **Printers** contextual Ribbon page. Alternatively you can use the **Printer Entries** drop-down button located on the **Project** Ribbon page in the **New** group. The **New Print Monitor** dialog will appear on the screen to let you configure the required deployment action Pic 4.



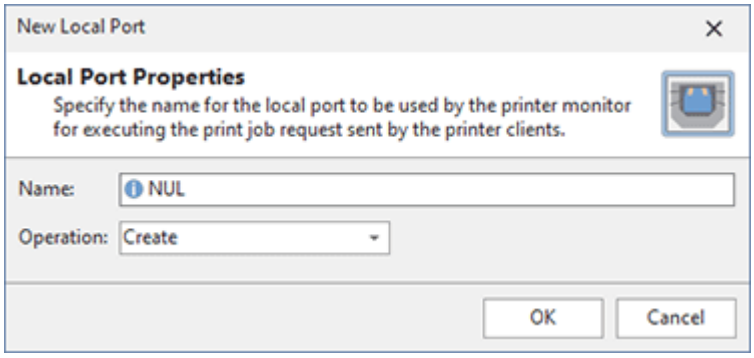

**Pic 4. Adding a print monitor**

Copyright © 2001-2023 EMCO. All rights reserved.

When adding a print monitor, you should define its name, the environment to be affected and the operation to perform over this monitor. For the monitors being installed, you should also choose the monitor binary from the file available in the configured project.

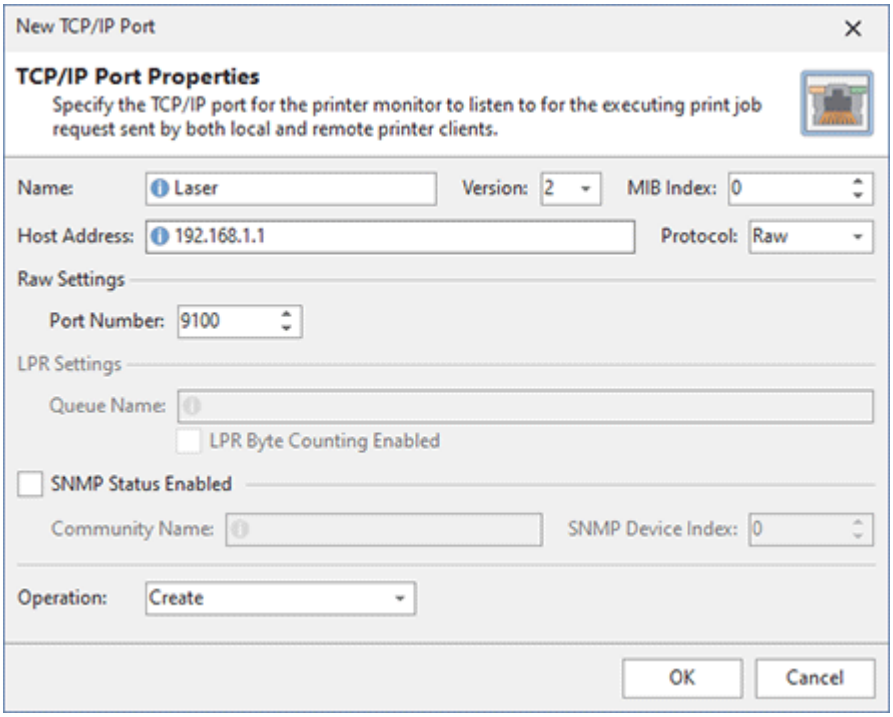
The last layer available for management with a package generated via MSI Package Builder is the printer ports. There are two types of ports, namely local and TCP/IP. Packages generated with MSI Package Builder can create and delete both local and TCP/IP printer ports as well as modify TCP/IP ports.

To add a printer port operation, you should use the **Printer Port** item either from the **Printers** view pop-up menu or from the **New** group on the **Printers** contextual Ribbon page. Alternatively you can use the **Printer Entries** drop-down button located on the **Project** Ribbon page in the **New** group. A dialog will be displayed for choosing the port type, and as soon as it is chosen, you will get to either the **New Local Port** dialog Pic 5 or the **New TCP/IP Port** dialog Pic 6.




**Pic 5. Adding a local printer port**

When adding a modification to local ports, you should simply provide the local port name to the **Name** field and specify the operation to be performed with this port (either **Create** or **Delete**) in the **Operation** drop-down.




**Pic 6. Adding a TCP/IP port**

Copyright © 2001-2023 EMCO. All rights reserved.

The TCP/IP ports configuration process is a bit more complex. You can choose if you would like to create, modify or delete the port being added using the **Operation** drop-down. For deleting a port, it is only required to provide its name. For both adding and modifying a port, you should also provide the port properties. Beside the name, the port is identified with its version, the MIB index and the host address. The values should be provided to the **Version**, **MIB Index** and **Host Address** fields respecrively. If the port is meant for raw bytes communication, then the **Raw** value should be selected in the **Protocol** drop-down list, and the port number should be specified in the **Port Number** field. In case the LPR protocol is used, the corresponding value should be selected in the **Protocol** drop-down, and the LPR queue name should be specified in the **Queue Name** field. In case the LPR byte counting is enabled for communication, the corresponding check box should be ticked. If the printer device residing on the port can report the SNMP status, the **SNMP Status Enabled** check should be ticked, and the required values should be provided to the **Community Name** and **SNMP Device Index** fields.



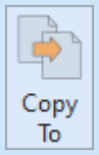

**Edit**

The **Edit** button from the **Management** group on the **Printers** contextual Ribbon page should be used to change the selected printing system configuration action.



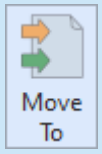

**Delete**

The **Delete** button from the **Management** group on the **Printers** contextual Ribbon page allows you to delete the selected printing system configuration actions from the selected project.




**Copy To**

The **Copy To** button from the **Management** group on the **Printers** contextual Ribbon page should be used to copy the selected printing system configuration actions to another location.




**Move To**

The **Move To** button from the **Management** group on the **Printers** contextual Ribbon page allows you to move the selected printing system configuration actions to another location.

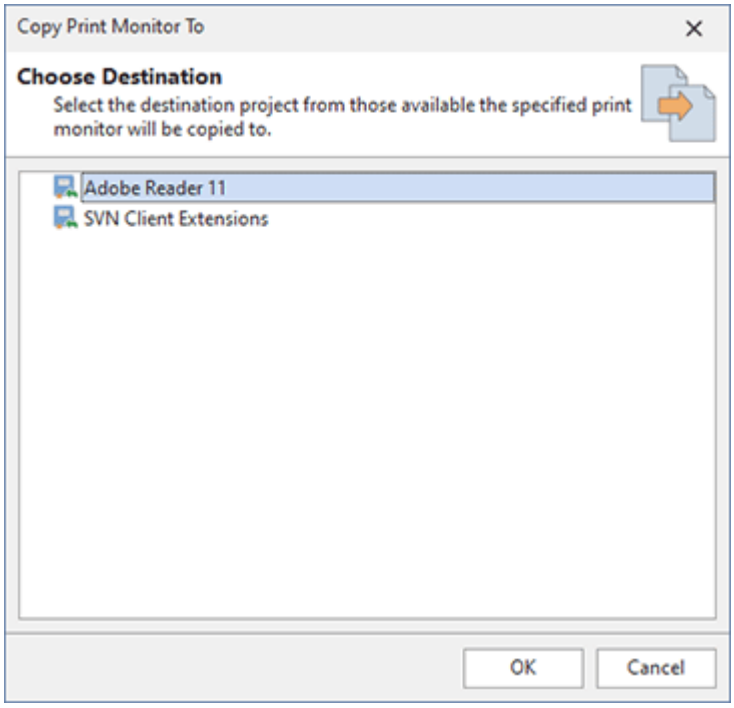
The printing system configuration actions can be changed, deleted and transferred to different location. To change the printing system configuration action, select it in the **Printers** view and double-click it. Alternatively you can either choose the **Edit** item in the pop-up menu or click the **Edit** button on the **Printers** contextual Ribbon page. While editing a printing system configuration action, it is possible to configure the same options as during its creation. To delete specific actions from the project, select them and choose the **Delete** item from the pop-up menu or click the **Delete** button on the **Printers** contextual Ribbon page.



If the project contains some printers that you don't want to include into the generated package, you can use the context menu option **Exclude from Build**. Excluded resources are displayed in the project using the faded icons. Use the context menu option **Include in Build** for excluded resource to add them into the generated package. The **Exclude from Build** option allows you to make experiments with the set of resources included into the resulted package with no deleting resources from the project.

Copyright © 2001-2023 EMCO. All rights reserved.

The well-known drag/drop and copy/paste techniques are fully supported for copying and moving printing system management actions between projects. Also you can use the **Copy To** and **Move To** buttons from the **Management** group on the **Printers** contextual Ribbon page to transfer the selected printing system configuration actions to a different location. While using these buttons, you are suggested to select a target location in a dialog and confirm your selection Pic 7.




**Pic 7. Copying printing system configuration actions**

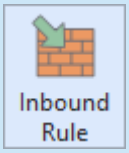
Now that you have been introduced to the features used for managing the printing system configuration actions with MSI Package Builder, you should be able to set up a printing system with the help of a generated MSI package without any difficulties.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Windows Firewall Modifications

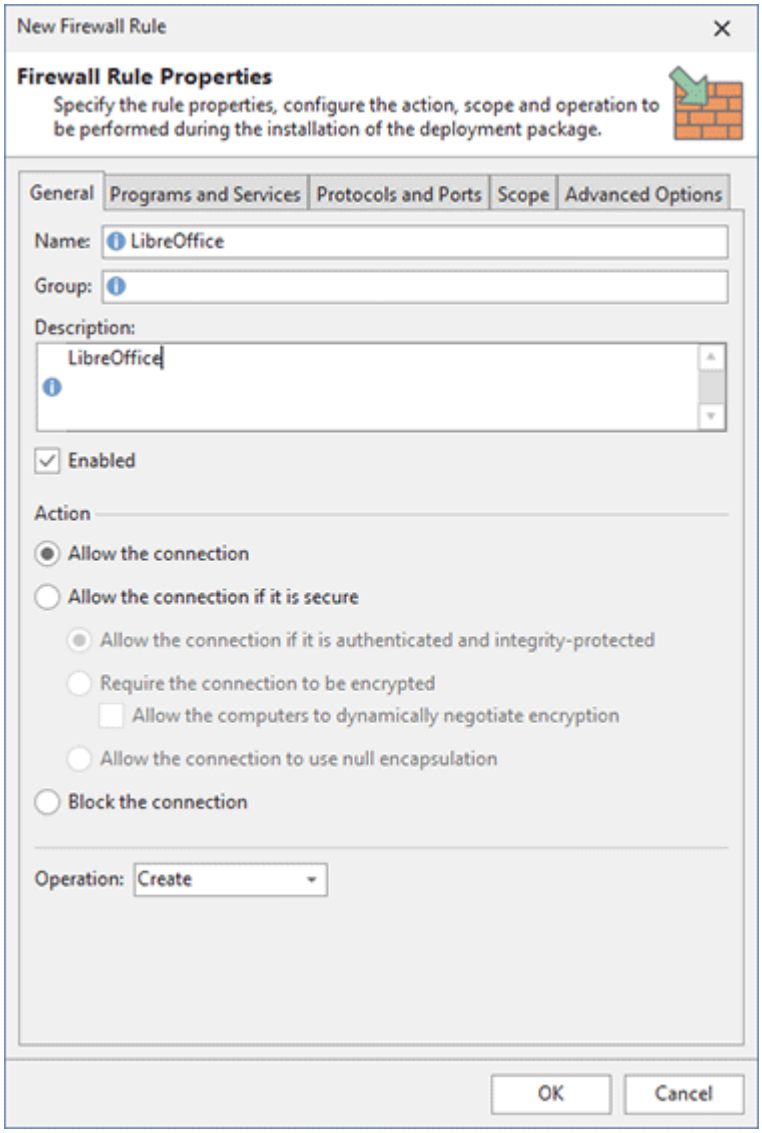
MSI Package Builder allows modifying Windows Firewall settings. It is possible to create, modify and delete inbound and outbound firewall rules and to change firewall profile settings during installation and uninstallation of the created package.

Windows Firewall settings described in the project are displayed in the **Windows Firewall** view when the **Windows Firewall** node is selected in the **Projects** view. Changes performed during the monitoring process are automatically added to the view, so you can edit them or add the required changes manually. Changes to the inbound and outbound rules are represented on the **Inbound & Outbound Rules** tab. Let us take a closer look at the possible types of actions.

| | |
|---|---|
|  | **Inbound Rule**<br>The **Inbound Rule** button from the **New** group on the regular **Project** and contextual **Windows Firewall** Ribbon pages should be used to add a firewall inbound-rule deployment action to the currently configured project. |
|  | **Outbound Rule**<br>The **Outbound Rule** button from the **New** group on the regular **Project** and contextual **Windows Firewall** Ribbon pages should be used to add a firewall outbound-rule deployment action to the currently configured project. |

Copyright © 2001-2023 EMCO. All rights reserved.

When creating an inbound or outbound rule, you need to configure the rule properties in the **New Firewall Rule** dialog Pic 1. First, you need to configure the operation type on the **General** tab of the dialog. The Create option allows creating a new firewall rule when a package is deployed. If you need to modify a rule, you should select the Modify option. To delete a rule, you can select the Delete or Delete by Name options.



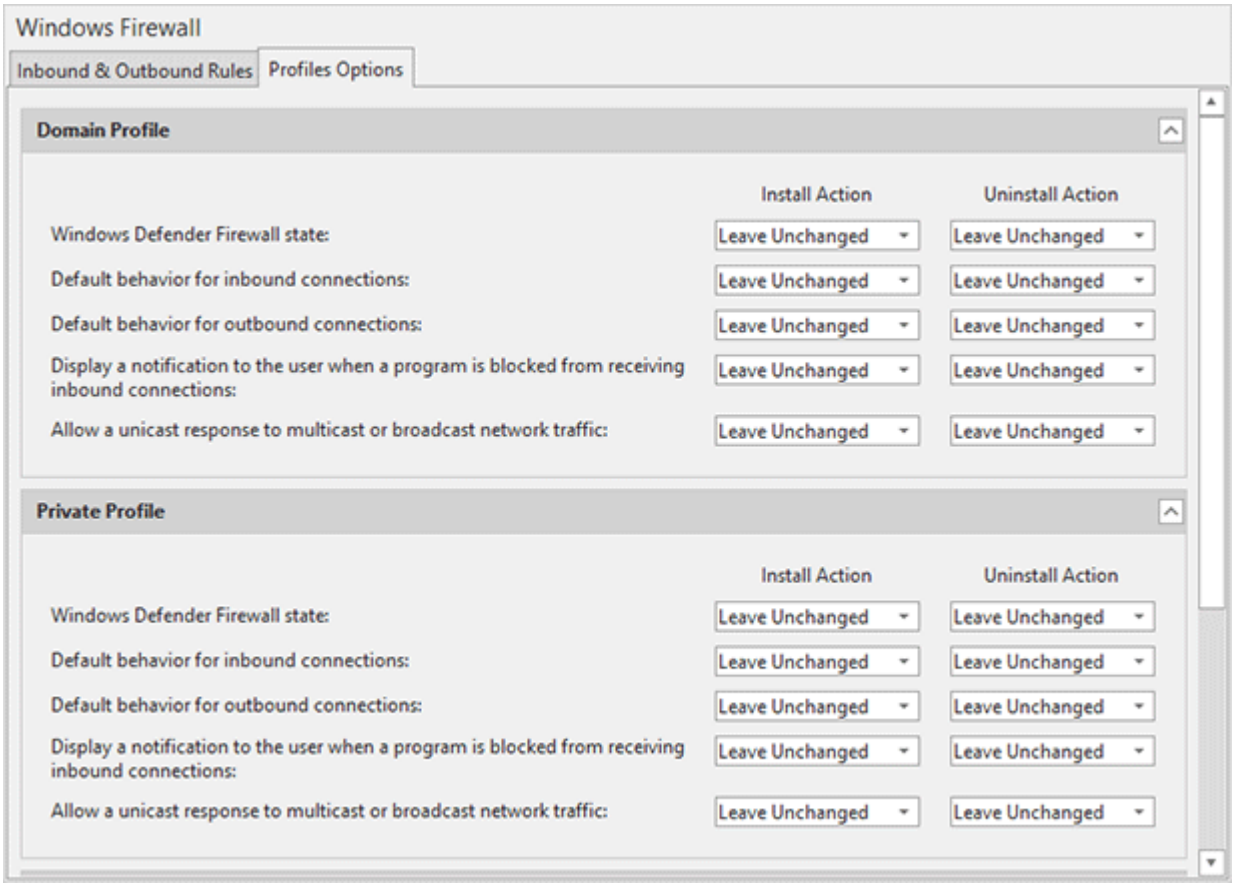

**Pic 1. Configuring firewall rule options**

*Copyright © 2001-2023 EMCO. All rights reserved.*

Using the options configured in the **Firewall Rule Properties** dialog, you can configure a rule to allow or block a connection. You can switch to different tabs of the dialog to configure the programs and services for the rule to be applied to and to specify ports, scopes and protocols. These settings are similar to the settings of the Windows Firewall configuration.

If the Delete option is specified in the firewall rule, you need to configure all the rule options. In this case, a rule is deleted during the package deployment only if all the options match the options of the rule on the machine. The rule deletion doesn't work if any option doesn't match. To delete a rule regardless of its options, you can use the Delete by Name option. In this case, you should specify the rule name, so all rules with this name will be deleted.

To configure profiles in Windows Firewall, you can switch to the **Profiles Options** tab. It is possible to configure options for the Domain, Private and Public profiles. There is a set of options for every profile, and you can configure the changes for those options to be applied upon the package installation and uninstallation Pic 2.



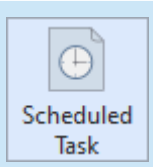

**Pic 2. Configuring firewall profile options**

Using the configuration options, you can turn on or off Windows Defender Firewall for a selected profile, set the default behavior for inbound and outbound connections or configure displayed notifications. You can configure different install and uninstall actions, if you need.

Copyright © 2001-2023 EMCO. All rights reserved.

## Scheduled Tasks Deployment

It is possible to create and delete scheduled tasks in Windows during project deployment. A set of scheduled tasks changes described in the project is displayed in the **Scheduled Tasks** view when the **Scheduled Tasks** node is selected in the **Projects** view. You can use the follow action to configure a scheduled task change in the project.



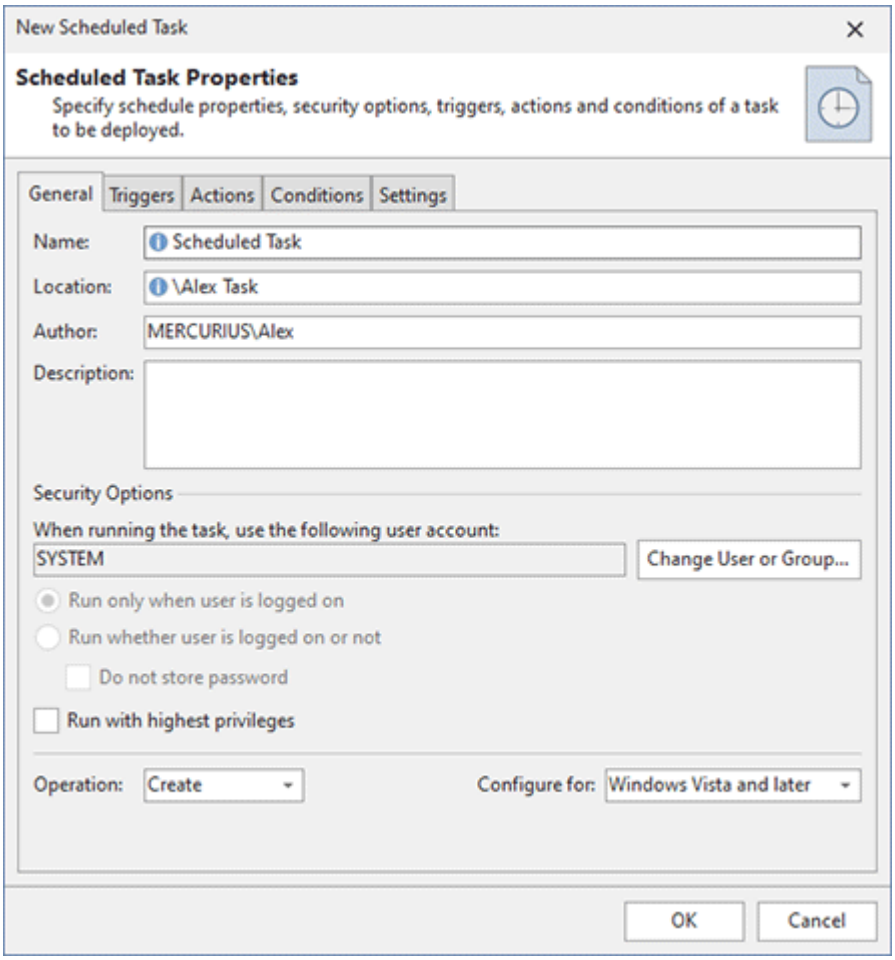

**Scheduled Task**

The **Scheduled Task** button from the **New** group on the regular **Project** and contextual **Scheduled Tasks** Ribbon pages should be used to add a new Windows scheduled task deployment change to the currently configured project.

When you configure a scheduled task you have to specify its general options, execution triggers, actions, executing conditions and run settings. All these configuration parameters and displayed in the correspond tabs of the **New Scheduled Task** dialog and are described below.

## Task General Options

When you configure a new scheduled task you start from providing settings on the **General** tab of the **New Scheduled Task** dialog Pic 1. On this tab you have to specify a name of the task, task storage location and the name of the task author. You can also specify a description for the task, if required.




**Pic 1. Scheduled task general options**

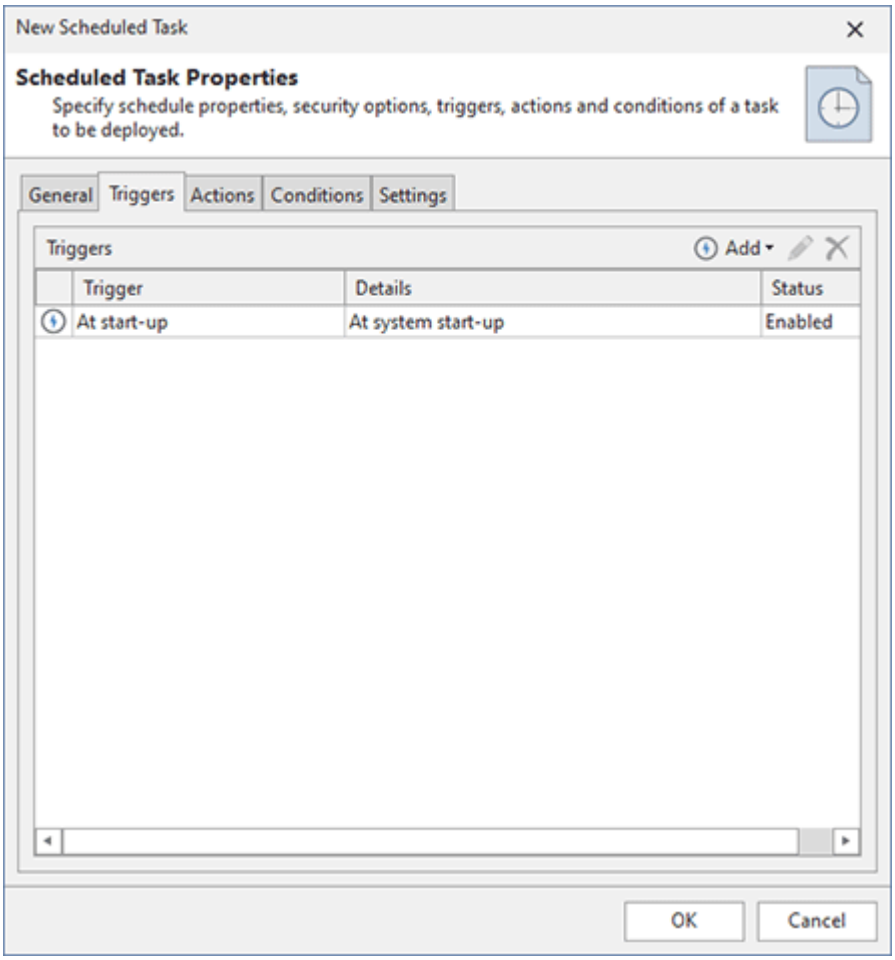
Copyright © 2001-2023 EMCO. All rights reserved.

On the same tab you can also specify security options, such as a Windows user account to run the task. Click the **Change User or Group...** button to retrieve and select an account, otherwise a task will be executed using the system account. In this section you can configure the task to run only if the selected user account is logged on.

The program allows you to create a new scheduled task or delete an existing task on the package deployment, so you have to select the corresponding **Operation** option. You can also select an operation system compatibility options.

## Task Triggers

On the **Triggers** tab you can configure triggers to be used to activate the task. You can add one or multiple triggers using the **Add** button, located on the toolbar or in the context menu of the **Triggers** table Pic 2.




**Pic 2. Task triggers configuration**

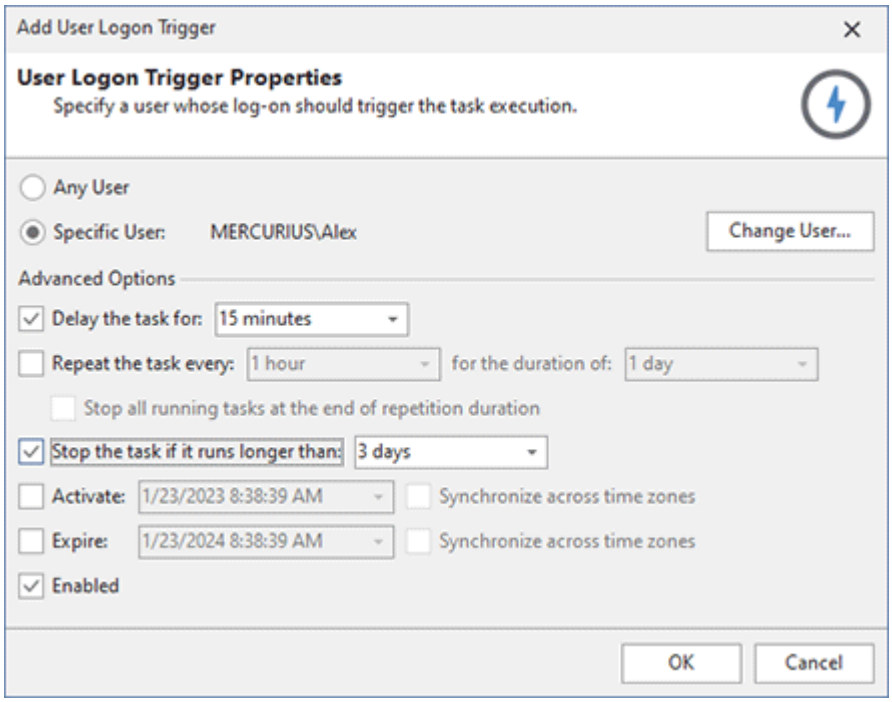
The program supports different types of triggers that are used to activate a task. Windows runs the task when the trigger condition is satisfied. Each trigger type contains its own configuration parameters depending on the nature of the trigger. Supported triggers are listed below.

| | |
|---|---|
| **System Startup Trigger** | Run the task on the system startup, when the machine is booted. |
| **Computer Idle Trigger** | Run the task when the machine is in the idle state. |
| **User Logon Trigger** | Run the task when a user logs on into system. |

Copyright © 2001-2023 EMCO. All rights reserved.

| | |
|---|---|
| **Scheduled Trigger** | Run the task on the regular basis – daily, weekly, etc. |
| **Event Trigger** | Run the task when a specific event occurs. |
| **Run Once Trigger** | Run the task at the specified date and time. |
| **Task Registration Trigger** | Run the task when a task is created or modified. |
| **User Session Trigger** | Run the task when a user connects or disconnects the system locally or remotely, or when the workstation is locked or unlocked. |
| **Windows Notification Trigger** | Run the task when a specific Windows Notification Facility (WNF) code occurs. |

Each trigger type can have custom configuration depending on the trigger type and also has common configuration suitable for all trigger types. Let's take a look at the trigger configuration. For example, when you configure a **User Logon Trigger** Pic 3, you need to specify a user account to fire the trigger or select the option to fire the trigger for any user. This configuration is specific for this type of the trigger.



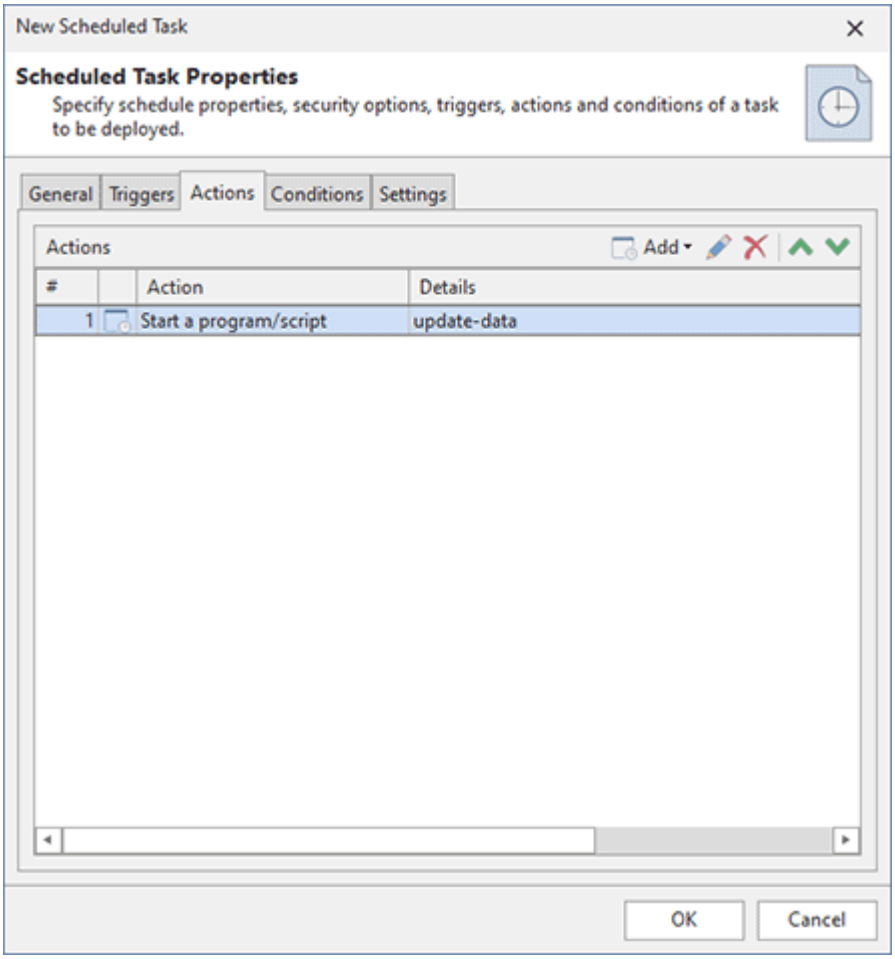

**Pic 3. User Logon Trigger configuration**

Copyright © 2001-2023 EMCO. All rights reserved.

Other trigger configuration options are the same for all types of triggers. You can delay the task for the configured time interval. Or you can repeat the task execution. Or stop the task if it runs longer than the specified period. It is also possible to activate and expire the task at configured date and time.

## Task Actions

A task can run one or multiple actions that are configured on the **Actions** tab. Pic 4 You can add a new action by pressing the **Add** button on the toolbar of the **Actions** table or select the **Add** option in the context menu.



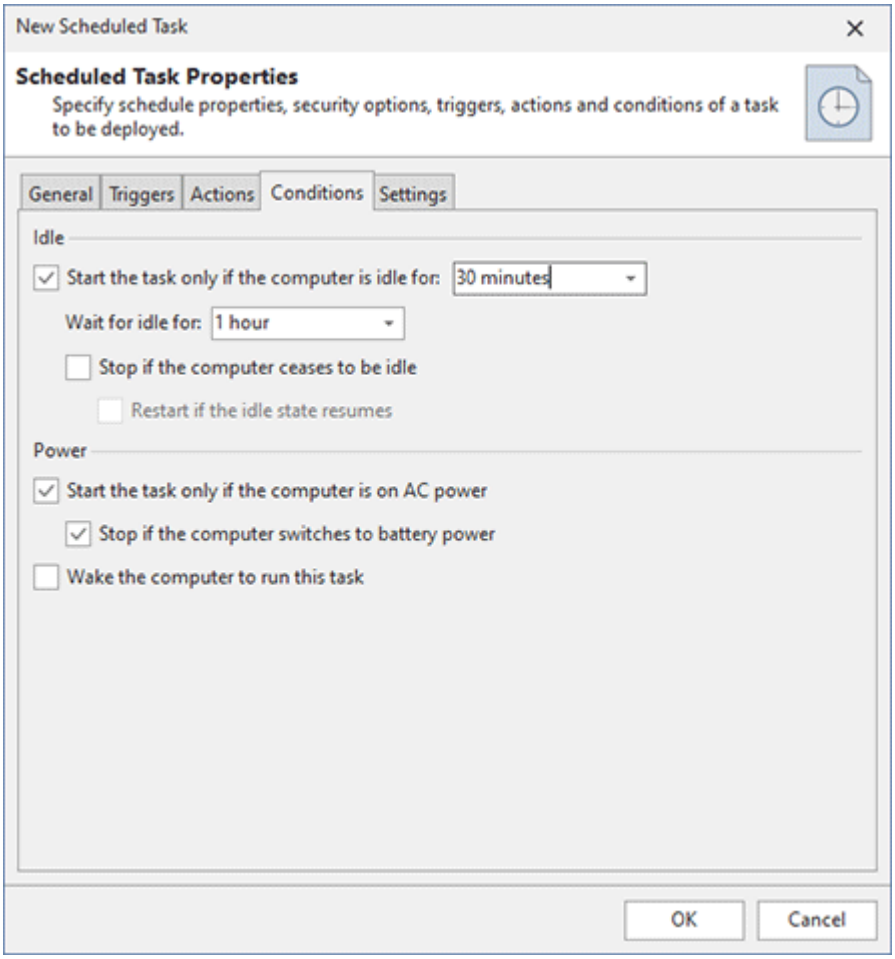

**Pic 4. Task actions configuration**

Copyright © 2001-2023 EMCO. All rights reserved.

It is possible to start a program or run a COM object as a task action. When you configure a program you have to specify a program or script to run, its execution arguments and the start folder. For a COM object you should specify ClassID and data.

## Task Conditions

On the Conditions tab you need to configure conditions that should be satisfied to run the task. Pic 5 You can configure a task to be started only when a computer is in the idle state. Specify how long the computer should be in idle before running this task and how long to wait for idle.



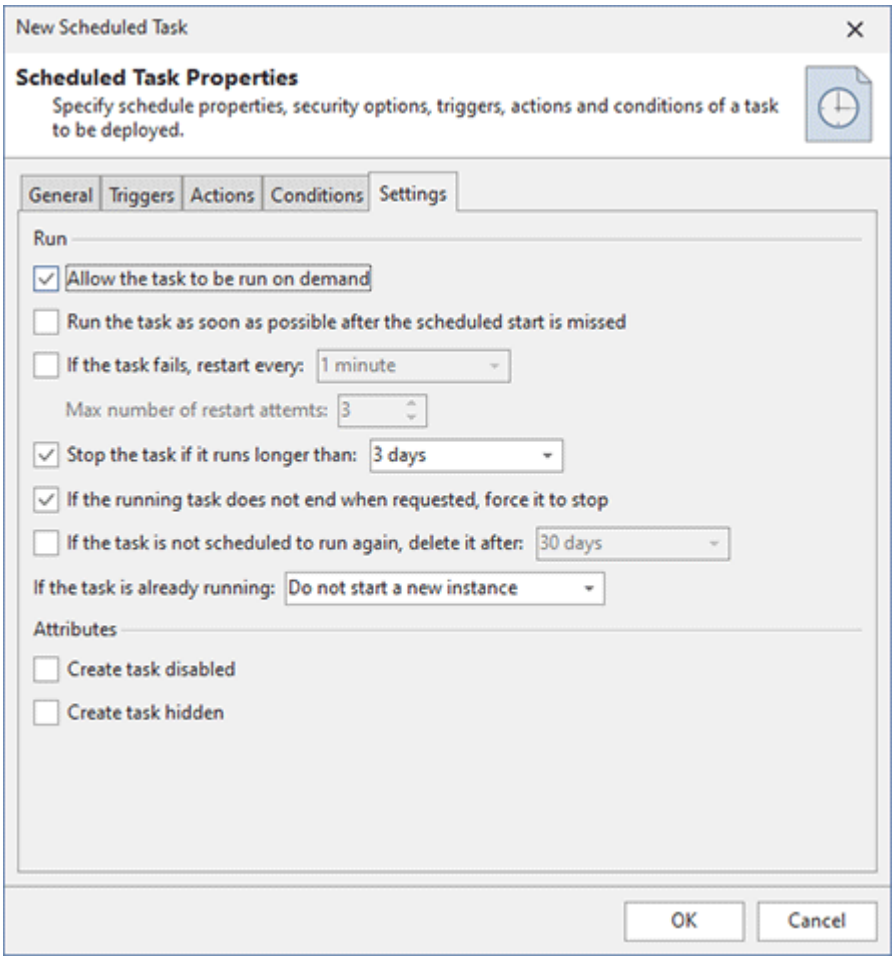

**Pic 5. Task conditions configuration**

Copyright © 2001-2023 EMCO. All rights reserved.

You can also configure to run the task when the machine runs on AC power or wake the machine to run the task.

## Task Settings

On the Settings tab you can configure task running and creating options. Pic 6 These options configure various aspects of task behavior – how it can be started, what to do on the task failure, how task should be stopped, etc.




**Pic 6. Task settings**

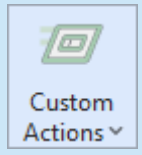
On the same tab you can also configure task creation options. It is possible to create a hidden and disabled task.

Copyright © 2001-2023 EMCO. All rights reserved.

## Using Custom Actions

The custom actions are the actions that can be performed together with the MSI package install and/or uninstall process. They can be used, for example, to prepare the system for installation of the package, to check prerequisites, to start the application on installation completion, etc. These actions are defined on a project level and are displayed in the **Custom Actions** view when the **Custom Actions** node of a project is selected in the **Projects** view.

There are two kinds of custom actions, namely pre & post actions and system actions. The pre & post actions are generic commands to be performed, and the system actions are predefined commands.




**Custom Actions**

The **Custom Actions** drop-down button from the **New** group on the **Project** Ribbon page allows creating a new action of a specific type and add it to the selected project.

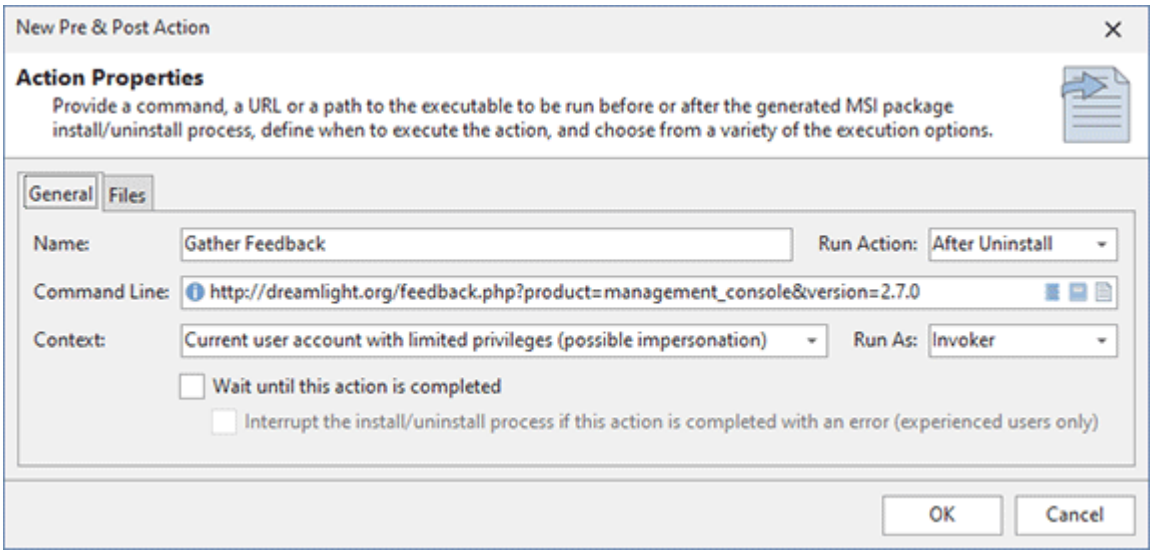



**Pre & Post Action**

The **Pre & Post Action** button from the **New** group on the **Custom Actions** Ribbon page should be used to create a new generic action and add it to the selected project.

## Managing Pre & Post Actions

To create a new action, choose the **New Pre & Post Action** item in the **Custom Actions** view pop-up menu from the **Pre & Post Actions** part. Alternatively, you can use the **Pre & Post Action** option within the **Custom Action** button from the **New** group on the **Project** Ribbon page and the **Pre & Post Action** button on the contextual **Custom Actions** Ribbon page. In any case the **New Pre & Post Action** dialog will appear on the screen to let you configure the action to be created Pic 1.




**Pic 1. Creating a new pre & post action**

Copyright © 2001-2023 EMCO. All rights reserved.

Each pre & post action can be represented with a command (such as an executable or script file) or a simple action to open a file or an URL. The action to be executed should be specified in the **Command Line** field, where you can also configure optional command-line parameters to be passed to the executable/script.

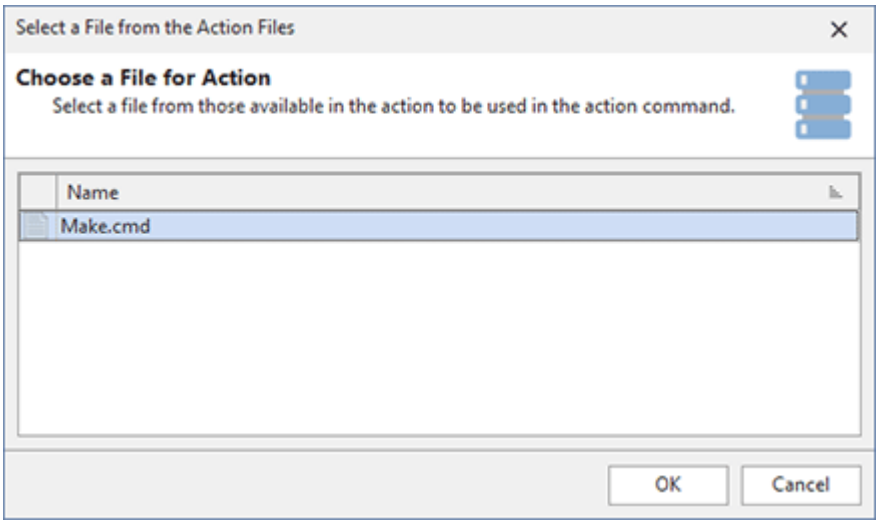
A custom action can be executed before install of the package, after install, before uninstall or after uninstall. It's important to understand when the action will be executed to configure a command properly. A custom action can run executable and script files that is a part of deployed package, but for actions executed before install or after uninstall these files are not available because the installation isn't deployed them (in the case of before install action) or already deleted them (in the case of after uninstall action). Custom actions executed before install or after uninstall can run scripts/executables always available in a system, for example system commands. Or you can configure action files, as explained below, and use them to run an action.

To specify a command using an executable/script file use buttons available in the **Command Line** edit box.

- Choose the file from those available in the action files
- Choose the file from those available in the project
- Choose the file from those available on the system

## Using a File from Action Files

Custom actions can operate with own set of files that are deployed independently from the deployed package. You can configure those files on the Files tab of the custom action configuration dialog Pic 2.




**Pic 2. Configuring custom action files**

Copyright © 2001-2023 EMCO. All rights reserved.

If you have been added custom actions files, you can use the **Select a File from the Actions Files** button of the **Command Line** edit box to configure a file to be executed as a custom action. When you select a file, it is represented as *${ActionFiles}$\<file>* in the commend line, where *<file>* is the file you selected and *${ActionFiles}$* is a property definition placeholder that specified a path to the storage of custom action files. You can add command-line parameters to the command, if required.

Using a **File from Actions Files** allows you to run executables/scripts independently from the type of the custom action (before/after install/uninstall) and environment. Custom actions operate with the own set of files always available in all the cases, so, for example, you can execute a custom script before package installation or after uninstallation, for example. Other options, represented below, have limitations and aren't as flexible as this option.

## Using a File from the Project

This option allows you to run an executable or a script deployed by the package. The executable/script file should be available in the **File System** section of the project. This option can be used, for example, to run a file that is a part of the captured installation, for example. This file is a part of deployed changes, so you can run it.

This option has some limitations. You cannot use it to run a command before installation, because the file to be executed isn't deployed yet. You cannot use it to run a command after uninstallation, because the file is already deleted. Use it when you need to run a file that is a part of the deployed installation after install or before uninstall.

When you select a file from those available in the project, the path to the file may be represented using the property definition placeholders. They replace absolute paths to the files location with placeholders to represent special system folders to make the package independent from the deployment environment.

## Using a File

This option allows you to select a location on the local file system and is applicable only to run standard Windows commands. The custom action will work only if the configured executable with the same file path is available on the system where the package will be deployed. If you need to run an executable that isn't available on each system where the package will be deployed, use any of the options above.

The **Command Line** field supports the property definition placeholders, so you can use the standard MSI properties while configuring the action. See the Property Definition Placeholders section of this document for the list of available placeholders.



The Windows Installer does not allow installing, configuring and uninstalling packages in parallel, though it is impossible to execute installation, repair or uninstallation of another MSI package within the custom actions execution scope.

As for the start type, you can choose between **Before Install**, **After Uninstall**, **Before Uninstall** and **After Uninstall**. Thus, you can, for example check some prerequisites and perform preparation steps before the package installation and perform a kind of clean-up after uninstalling the package. For each action, you can choose the execution context, if to run it as administrator or as invoker in case of current user account context, if the installer should wait for the action to complete and if it should analyze the completion result.



The **Administrator** value from the **Run As** drop-down should be selected only for executing the actions that required administrative privileges to function. It is insecure to run all actions as administrator.

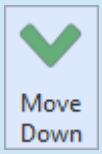
Copyright © 2001-2023 EMCO. All rights reserved.

If the successful action completion is required for the installation to complete successfully, than you can check both the **Wait until this action is completed** and **Interrupt the install/uninstall process if this action is completed with an error** options. However, be aware of the fact, that if the action is implemented incorrectly, the whole installation will fail.




**Move Up**

The **Move Up** button from the **Order** group on the contextual **Custom Actions** Ribbon page should be used to move the selected pre & post actions up the execution order.




**Move Down**

The **Move Down** button from the **Order** group on the contextual **Custom Actions** Ribbon page should be used to move the selected pre & post actions down the execution order.

By default, the pre & post actions execution order is the same as the addition order, but you can reorder the actions using the **Move Up** and **Move Down** items from the pop-up menu or the **Order** group on the contextual **Custom Actions** Ribbon page.

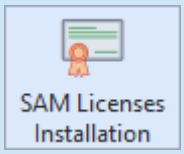
## Custom Action Templates

Some custom actions are used regularly in different projects. To simplify configuring those actions the program provides preconfigured templates. Use **Run PowerShell Script**, **Run VB Script**, **Run Batch File** or **Run Installed Application** actions available in the **Action Templates** group of the contextual **Custom Actions** tab on Ribbon to configure a corresponding action.

Action templates allow you to select a file to be executed. This file is automatically added to action files storage and the path to the file is specified in the command together with the standard command-line options. Other suitable configuration options are also set to the required values. You only need to set **Run Action** according with your needs.

## Managing System Actions

To create a new action for installing Software Assets Management (SAM) licenses, choose the **New SAM Licenses Installation** item in the **Custom Actions** view pop-up menu from the **System Actions** part. Alternatively, you can use the **SAM Licenses Installation** option within the **Custom Action** button from the **New** group on the **Project** Ribbon page and the **SAM Licenses Installation** button on the **Custom Actions** contextual Ribbon page.

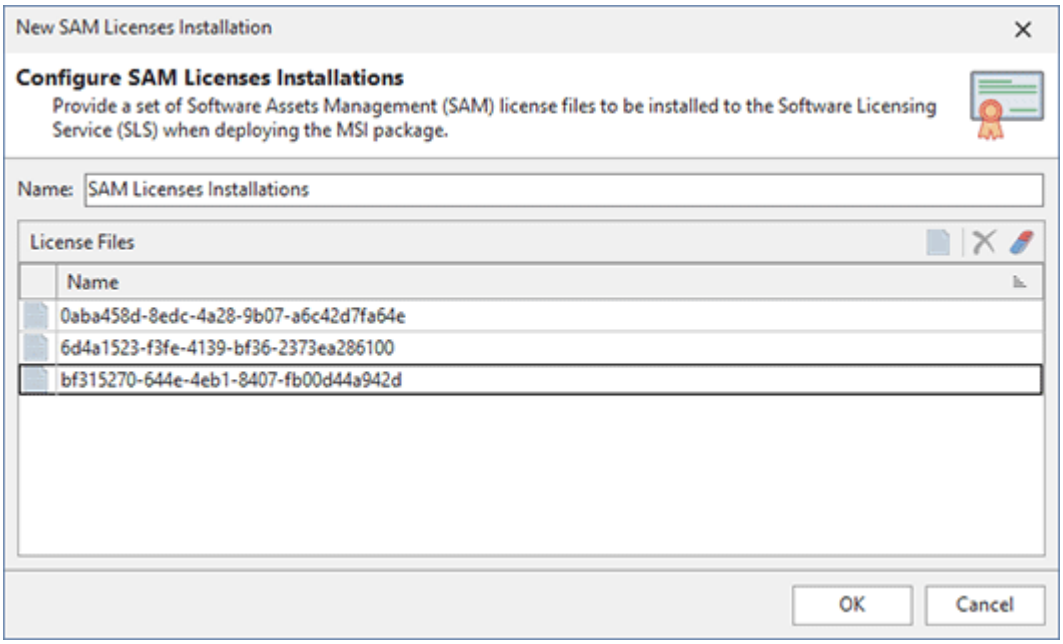



**SAM Licenses Installation**

The **SAM Licenses Installation** button from the **New** group on the **Custom Actions** contextual Ribbon page should be used to create a new action for installing Software Assets Management (SAM) licenses to the Software Licensing Service (SLS) and add it to the selected project.

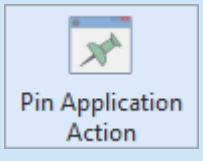
Copyright © 2001-2023 EMCO. All rights reserved.

Configure the action to be created in the New SAM Licenses Installation dialog Pic 3.




**Pic 3. Creating a new SAM Licenses Installation action**

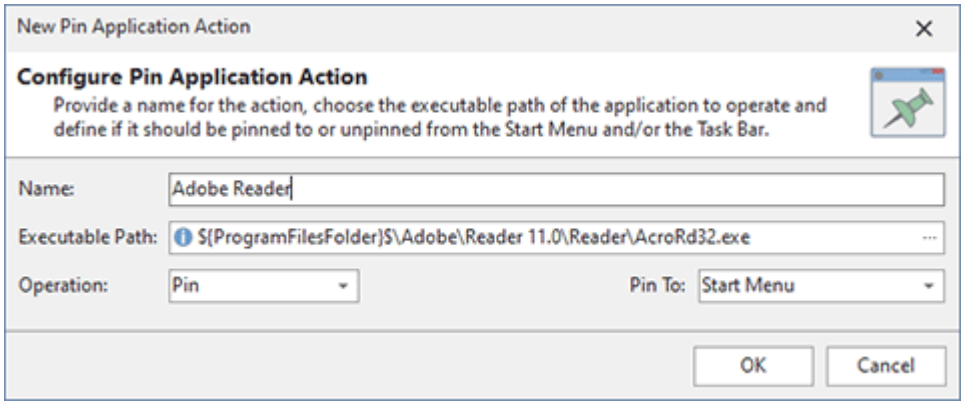
For each SAM licenses installation action, you can define a name and a set of licenses to be installed to the Software Licensing Service (SLS) during the installation of the generated MSI package.




**Pin Application Action**

The **Pin Application Action** button from the **New** group on the **Custom Actions** contextual Ribbon page allows you to create a new action to pin any application to or unpin it from the Start Menu and/or the Task Bar.

If you would like to pin any application to or unpin it from the Start Menu and/or the Task Bar, you can use the Pin Application action. To create such an action, choose the **New Pin Application Action** in the **Custom Actions** view pop-up menu from the **System Actions** part. Alternatively, you can use the **Pin Application Action** option within the **Custom Action** button from the **New** group on the **Project** Ribbon page and the **Pin Application Action** button on the **Custom Actions** contextual Ribbon page. The **New Pin Application Action** dialog will appear on the screen in any case Pic 4.




**Pic 4. Creating a new Pin Application action**

Copyright © 2001-2023 EMCO. All rights reserved.

For each action, you should define its name, the application to operate, whether the application should be pinned or unpinned, and if the operation should be performed for the Start Menu, the Task Bar or both.

## Editing, Deleting and Transferring Actions

The custom actions can be changed, deleted and transferred between projects. The corresponding actions are available in pop-up menu or on the contextual **Custom Actions** Ribbon page.



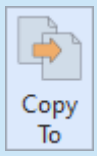

**Edit**

The **Edit** button from the **Management** group on the contextual **Custom Actions** Ribbon page should be used to change the selected action configuration.



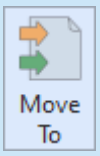

**Delete**

The **Delete** button from the **Management** group on the contextual **Custom Actions** Ribbon page allows you to delete the selected actions from the selected project.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **Custom Actions** Ribbon page should be used to copy the selected actions to another project.




**Move To**

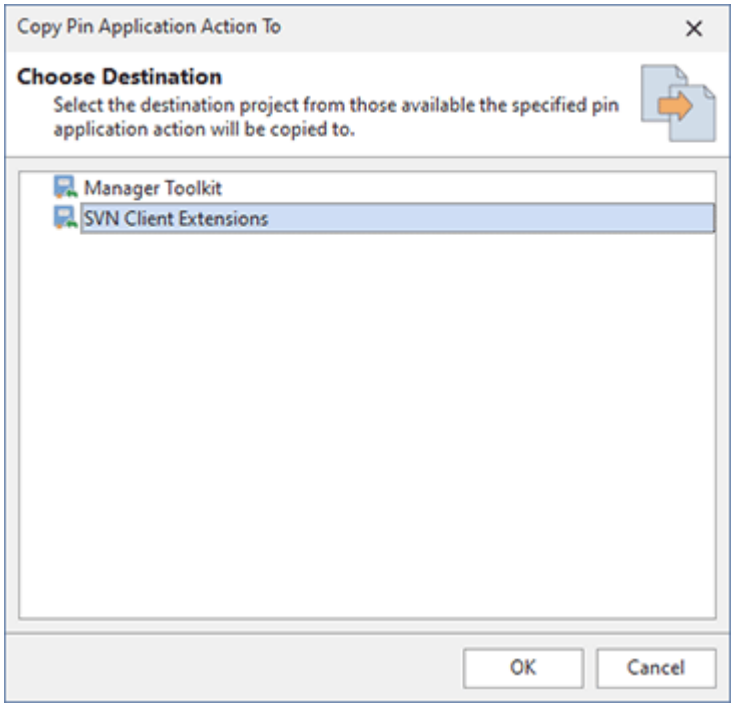
The **Move To** button from the **Management** group on the contextual **Custom Actions** Ribbon page allows you to move the selected actions to another project.

To change the action, select it and either double-click, or choose the **Edit** item in pop-up menu or on the contextual **Custom Actions** contextual Ribbon page. While editing an action, you can define the same scope of properties as during its creation. To delete the action from the project, select it and choose the **Delete** item in the pop-up menu or on the contextual **Custom Actions** contextual Ribbon page.

If the project contains some custom actions that you don't want to include into the generated package, you can use the context menu option **Exclude from Build**. Excluded resources are displayed in the project using the faded icons. Use the context menu option **Include in Build** for excluded resource to add them into the generated package. The **Exclude from Build** option allows you to make experiments with the set of resources included into the resulted package with no deleting resources from the project.

*Copyright © 2001-2023 EMCO. All rights reserved.*

The well-known drag/drop and copy/paste techniques are fully supported for copying and moving custom actions between projects. You can also use the **Copy To** and **Move To** buttons from the **Management** group on the contextual **Custom Actions** contextual Ribbon page to transfer the selected actions to a different project. While using these buttons, you are suggested to select a project from those available in a dialog and confirm your selection Pic 5.



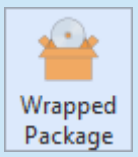

**Pic 5. Copying custom actions between projects**

Now you are introduced to the custom actions concept and the abilities the custom actions provide you with, thus you should be able to use this feature of MSI Package Builder to resolve appropriate tasks.

## Wrapping Existing Installations

One of the methods of installations repackaging supported by MSI Package Builder is wrapping existing installations into a generated MSI package. The generated MSI includes one or multiple installations that are deployed when you deploy the MSI. To create a wrap MSI package you need to specify wrapped installations and configure them for silent deployment, so they will be installed automatically when you install the generated MSI. You can also configure uninstall and repair options for wrapped installations if you want to uninstall and repair them when you uninstall and repair the MSI.

To use wrapping you need to create a project with wrapped packages and configure those packages. Each wrapped package in the project corresponds to an installation you need to wrap. The wrapped packages are defined on a project level and displayed in the **Wrapped Packages** view when the **Wrapped Packages** node of a project is selected in the **Projects** view.

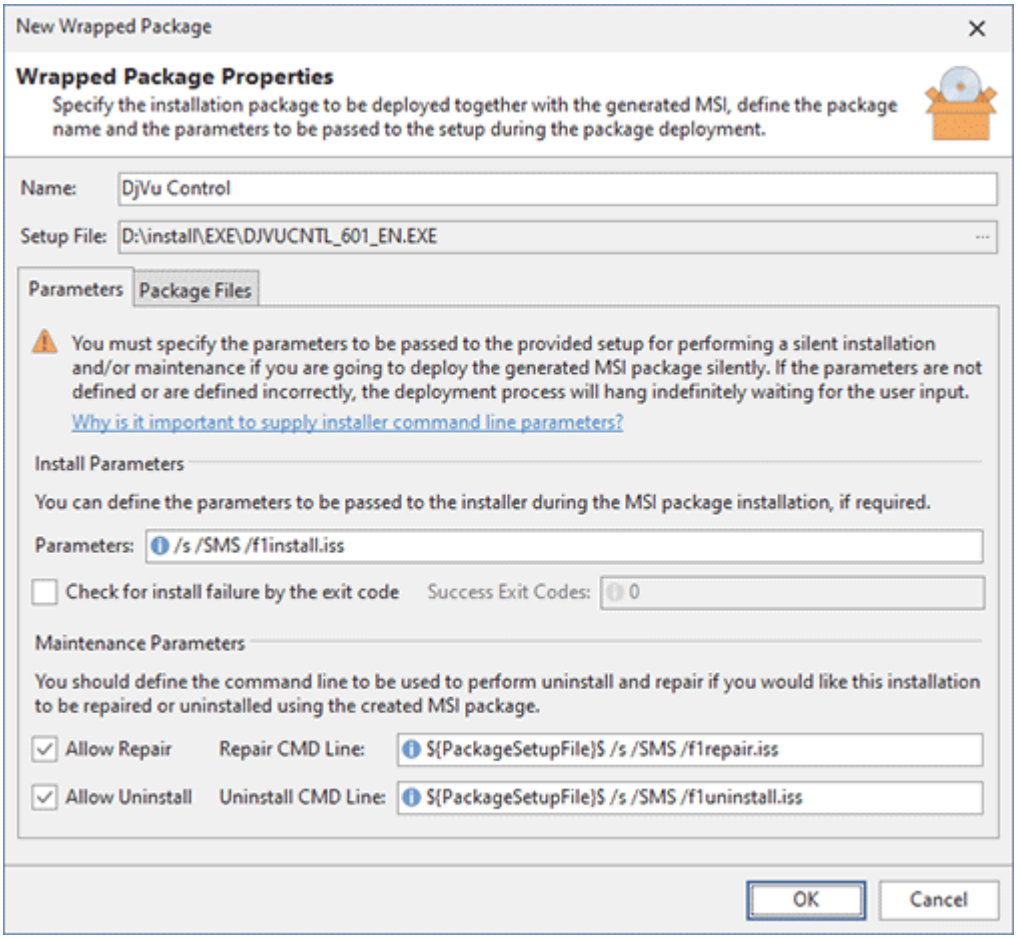



**Wrapped Package**

The **Wrapped Package** button from the **New** group on the **Project** Ribbon page and on the contextual **Wrapped Packages** Ribbon page should be used to create a new wrapped package and add it to the selected project.

Copyright © 2001-2023 EMCO. All rights reserved.

If the project was created in a scope of the **Repackage Installation** wizard, the packages defined during repackaging are automatically added to the project. You can also add any number of wrapped packages to any project manually. To add a new wrapped package, choose the **New Wrapped Package** item from the **Wrapped Package** view pop-up menu. Alternatively, you can choose the **Wrapped Package** button from the **New** group on the **Project** Ribbon page and on the contextual **Wrapped Packages** Ribbon page. The browse dialog will be displayed to let you choose the installation package to be wrapped. As soon as the installation is chosen, you are suggested to configure the package Pic 1.




**Pic 1. Configuring a wrapped package**

While configuring a package, you should provide a package name – this name is also used as the installation package name while deploying a generated MSI package. If additional files are required to install a wrapped package, they should be added to the **Additional Files** table, using the **Add File** and **Add Folder** buttons on the toolbar. You can add only files and folders that are located in the same folder as the setup file. In case the additional parameters are required to be passed to the installer during a generated MSI package installation, they should be provided to the **Parameters** field within the **Install Parameters** group. A wrapped installation has to be deployed silently. Usually an installation deploys silently when running with parameters activating silent deployment, so you have to specify these parameters in the **Parameters** field.

*Copyright © 2001-2023 EMCO. All rights reserved.*

It is possible to use the property definition placeholders when configuring both the installer parameters and maintenance parameters. See the Property Definition Placeholders section of this document for the list of available placeholders. Together with the common properties, you can also use the **${PackageSetupFile}$** and the **${PackageSetupDir}$** placeholders. The **${PackageSetupFile}$** placeholder stands for the path to the setup file during the package deployment and the **${PackageSetupDir}$** is replaced with the path to the setup file location.

In some cases you need to check the results of wrapped package deployment to revert MSI deployment if a wrapped installation fails to install. You can use the **Check for install failure by the exit code** option to check results of the wrapped packages deployment. You can enable this option and specify exit codes for a wrapped package. If during MSI installation this wrapped installation exits with different exit codes than specified, an installation failure is detected for the wrap MSI. In this case already deployed wrapped packages are uninstalled, if their uninstallation settings are specified, and the wrap MSI package entry isn't created.

To be able to repair and uninstall this wrapped package, you should enable the **Allow Repair** and **Allow Uninstall** options within the **Maintenance Parameters** group. In case if the wrapped package is an MSI, you can define the Windows installer parameters used for uninstall and repair, if required. Otherwise, if the wrapped package is an executable installer, you should define a command line to be executed to uninstall and/or repair the package.




**Move Up**

The **Move Up** button from the **Order** group on the contextual **Wrapped** Packages ribbon page should be used to move the selected packages up the deployment order.




**Move Down**

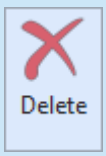
The **Move Down** button from the **Order** group on the contextual **Wrapped** Packages ribbon page should be used to move the selected packages down the deployment order.

By default, the wrapped packages deployment order is the same as the addition order, but you can reorder the packages using the **Move Up** and **Move Down** items from the pop-up menu or the **Order** group on the contextual **Wrapped Packages** Ribbon page. Please take into account, that the install, repair and uninstall order for the wrapped packages is the same.



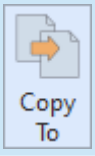

**Edit**

The **Edit** button from the **Management** group on the contextual **Wrapped** Packages Ribbon page should be used to change the selected wrapped package configuration.



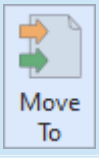

**Delete**

The **Delete** button from the **Management** group on the contextual **Wrapped** Packages Ribbon page allows you to delete the selected wrapped packages from the selected project.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **Wrapped** Packages Ribbon page should be used to copy the selected wrapped packages to another project.

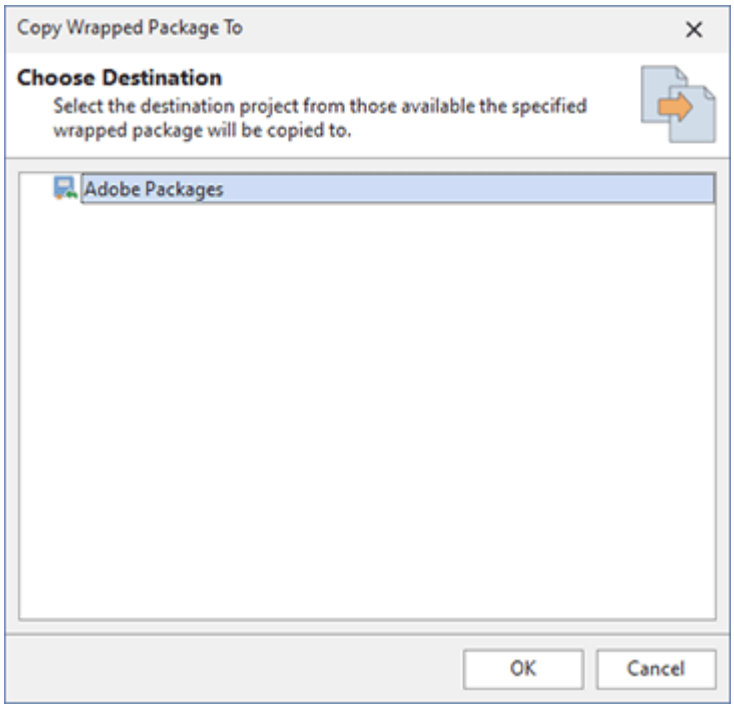



**Move To**

The **Move To** button from the **Management** group on the contextual **Wrapped** Packages Ribbon page allows you to move the selected wrapped packages to another project.

Copyright © 2001-2023 EMCO. All rights reserved.

The wrapped packages can be changed, deleted and transferred between projects. To change the wrapped package, select it and either double-click, or choose the **Edit** item in pop-up menu or on the contextual **Wrapped Packages** Ribbon page. While editing a wrapped package, you can define the same scope of properties as during its creation. To delete the wrapped from the project, select it and choose the **Delete** item in the pop-up menu or on the contextual **Wrapped Packages** Ribbon page. The well-known drag/drop and copy/paste techniques are fully supported for copying and moving wrapped packages between projects. You can also use the **Copy To** and **Move To** buttons from the **Management** group on the contextual **Wrapped Packages** Ribbon page to transfer the selected wrapped packages to a different project. While using these buttons, you are suggested to select a project from those available in a dialog and confirm your selection Pic 2.




**Pic 2. Copying wrapped packages between projects**

As you can see, the process of wrapping existing installations is fast and easy, but if you are going to deploy the generated MSI package in a silent mode, you should pay a special attention to the parameters passed to the wrapped package installer.

## Why is it important to supply installer command line parameters?

By default, most of the installation setup programs guide the end user through the setup wizard, thus the interaction with the user is required to complete the setup. So if you are going to deploy a generated MSI package in a silent mode, it is required the wrapped installation is performed without any interaction with the user. If the package is configured incorrectly, the deployment process may pause to wait for user input, and if there is no user during the remote deployment process, it will hang indefinitely. Installation setup programs mostly support silent execution, but the execution parameters may vary depending on the installation vendor.

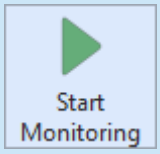
*Copyright © 2001-2023 EMCO. All rights reserved.*

Now let us take a look at the importance of an answer file. In some cases, the installer simply does not provide a default installation sequence without any user input, so a silent deployment without any additional configuration is impossible. Such installers commonly provide a technique for recoding an answer file to perform the deployment saving the user answers to that file. Another example is setups that use the same command for uninstall and repair and simply ask you what you would like to do on the first step of the installation wizard. If you launch an uninstall using this command with a default scenario, the installation setup may simply perform a repair, and that is not what you are expecting. Therefore, the required answer files, if any, must be added to the wrapped package and used while specifying the installer command line parameters Pic 1.

As a conclusion, we should emphasize that providing the parameters to be passed to the installer setup and the answer file, if required, is always a must. You should contact the installation package vendor or should do Internet search to get the proper parameters to be passed to each executable installer setup.

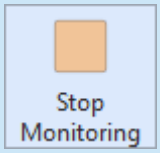
## Capturing Changes

MSI Package Builder comes with unique monitoring technology that provides you with an ability of tracking changes made to your operating system and create a project containing those changes. This feature can be used in a scope of the low-level repackaging process for creating a deployment package based on those changes. You can also use it to simply review the changes performed to the underlying operating system during the monitoring session.




**Start Monitoring**

The **Start Monitoring** button from the **Capture** group on the **Home** Ribbon page should be used to start a new session for tracking changes performed to the system.




**Stop Monitoring**

The **Stop Monitoring** button from the **Capture** group on the **Home** Ribbon page should be used to stop the currently running monitoring session.

To start a new monitoring session to capture the changes performed to the underlying operating system, you can press the **Start Monitoring** button from the **Capture** group on the **Home** Ribbon page or choose the **Start Monitoring** item from the **Projects** view pop-up menu. The wizard will appear on the screen, where you can provide a name for the project to be created on a basis of the changes monitored and choose additional processes to be filtered during the monitoring process. The monitoring modules will ignore the activity of the filtered processes, thus the changes performed by those processes will not be included into the resulting set. As soon as the monitoring is started, you can perform the changes to be monitored, thus run the applications you are going to track activity of and/or perform the changes manually.

To stop the capturing procedure, press the **Stop Monitoring** button from the **Capture** group on the **Home** Ribbon page or choose the **Stop Monitoring** item from the **Projects** view pop-up menu. When the monitoring process is stopped, the project with the specified name is created, containing all the changes performed to the underlying operating system while the monitoring was active. If you want to create a deployment package based on these changes, you should first prepare the created project. For detailed information on the projects preparation steps, refer to the Projects Preparation section of this document.

This feature is also used both in automated and advanced monitoring repackaging modes, but the preparation steps, as well as the MSI package creation steps, are automated. As you have now been introduced to the capturing abilities, you should be able to use those abilities in your everyday work.

Copyright © 2001-2023 EMCO. All rights reserved.

## Creating an MSI Package

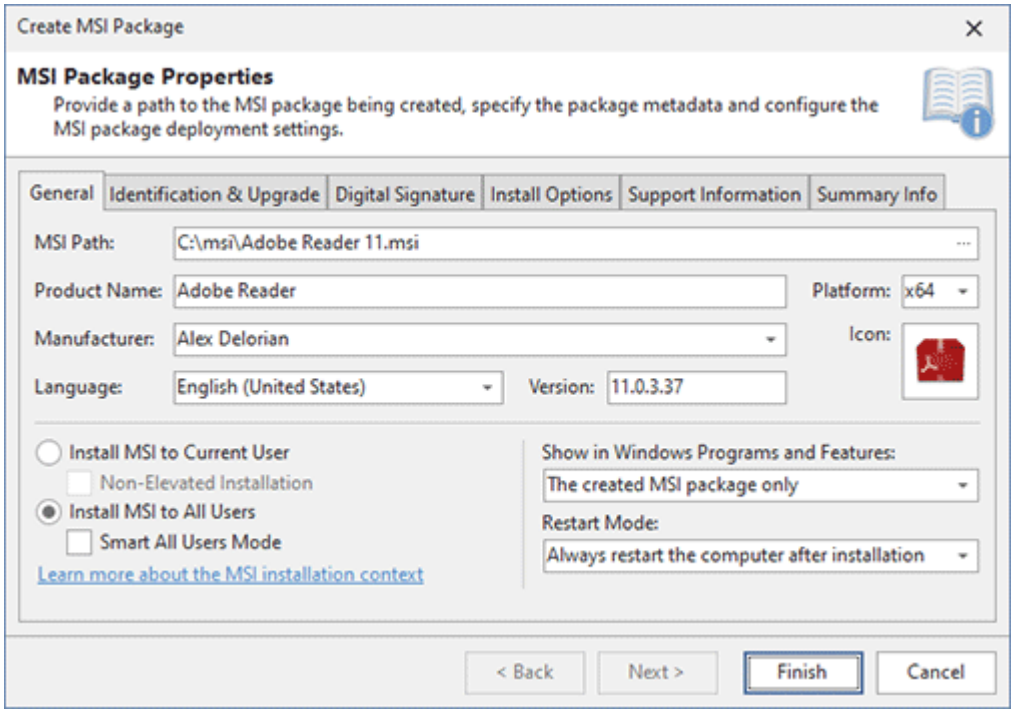
MSI Package Builder was designed to repackage existing installations into MSI packages, so one of the main aspects of the program is a flexible configuration of the package being created. The package can be created during the installation repackaging process, or manually from an existing MSI Package Builder project.




**Create MSI Package**

The **Create MSI Package** button from the **Builder** group on the **Home** Ribbon page should be used to generate an MSI package based on the selected project.

To create an MSI package manually, you can use the **Create MSI Package** item from the **Projects** view pop-up menu or from the **Builder** group on the **Home** Ribbon page. While creating an MSI package based on a project the options for package created are filled automatically using the values defined in the **Project Details** view, and during the repackaging process they are defined on-line and then saved to the created project. Let us take a look at the available options and describe each one, starting with the general product information Pic 1.



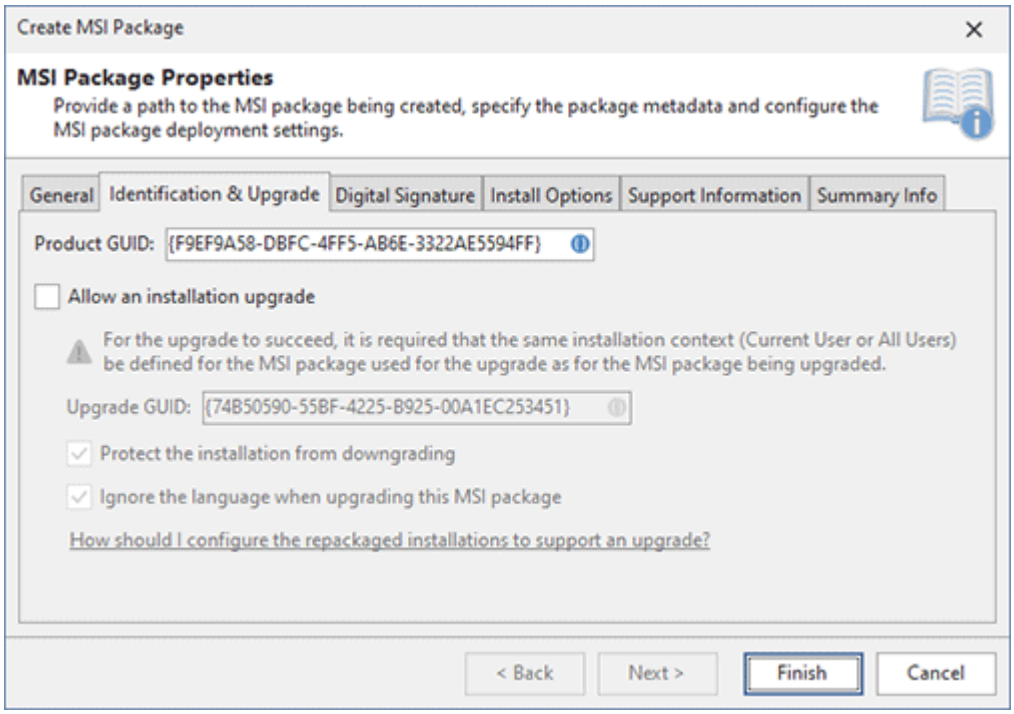

**Pic 1. Specifying general MSI package information**

Copyright © 2001-2023 EMCO. All rights reserved.

The general information consists of the **Product Name**, **Platform**, **Manufacturer**, **Icon**, **Language** and **Version**. The product name, manufacturer and icon are used to display the product in the **Programs and Features** section of the Windows control panel. The platform is used to define if the installer is deploying an x86 or an x64 application. The **Language** field is responsible for defining a language and encoding used by an installer package.

The version is the most important field of the general information, because it is responsible for the correct package installation and upgrade. That means that when you are preparing an updated package, using the same project, you must change the version to a bigger. This is also displayed in the **Programs and Features** section of the Windows control panel.

The other aspect of the general product information is an installation context. You can choose if the package when launched without additional parameters should be installed only for a current user account, or for all users. This is defined by switching between the **Install MSI to Current User** and **Install MSI to All Users** radio buttons. Please note, that the administrative privileges are required to install the MSI package for all users. You can also choose if the generated MSI package should be displayed in the **Programs and Features** section of the Windows control panel, and if the repackaged installations, if any, should also be displayed. If the reboot is required to complete the package installation, an appropriate options should be selected in the **Restart Mode** drop-down list.

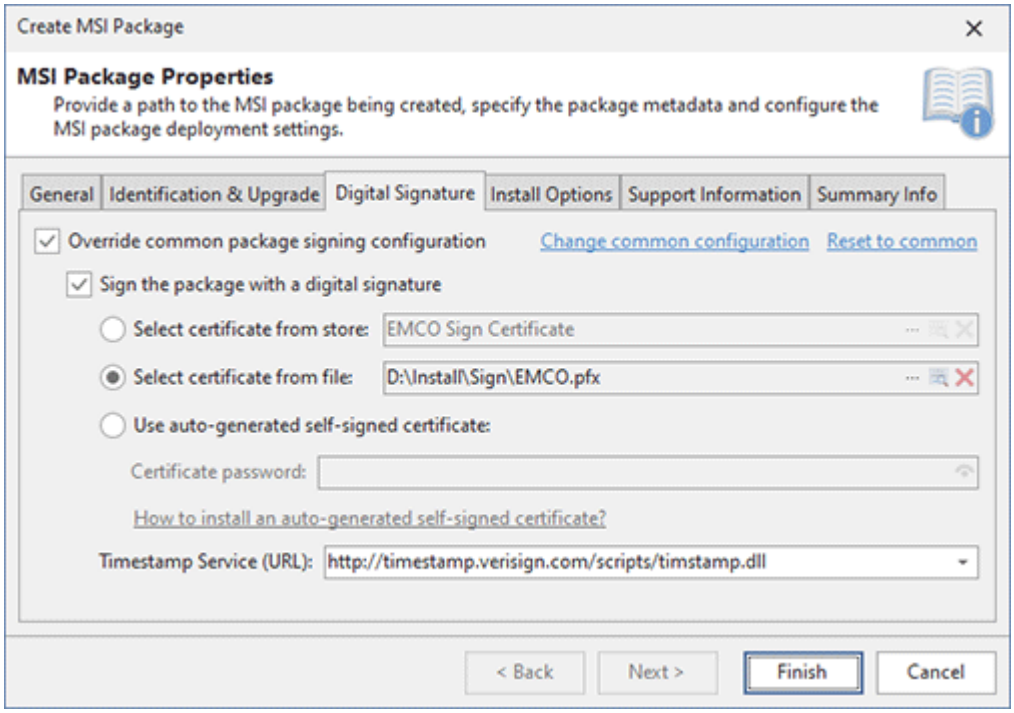



**Pic 2. Package Identification & Upgrade**

*Copyright © 2001-2023 EMCO. All rights reserved.*

The next step is configuring the package identification and upgrade abilities Pic 2. For MSI packages to be deployed and upgraded correctly, they should be properly identified. For each MSI package there is a **Product GUID**, that uniquely identifies the product, and **Upgrade GUID** that is used for matching upgrades. Windows Installer maintains its database and handles its consistency, depending on these identifiers, so you must be very careful with them. You may not provide the upgrade identifier, but in this case, the upgrade will not be supported. If you are going to support upgrades, you can define if the upgrade is language specific and if downgrade is allowed. In case you choose that the upgrade process is language specific, then it will be allowed to install a new version of an MSI package only if its language is the same as the language of the previous version installed. If you disable the protection from downgrade it will be possible to install a previous version of the product even when the newer version is installed, replacing the newer version with the previous one.

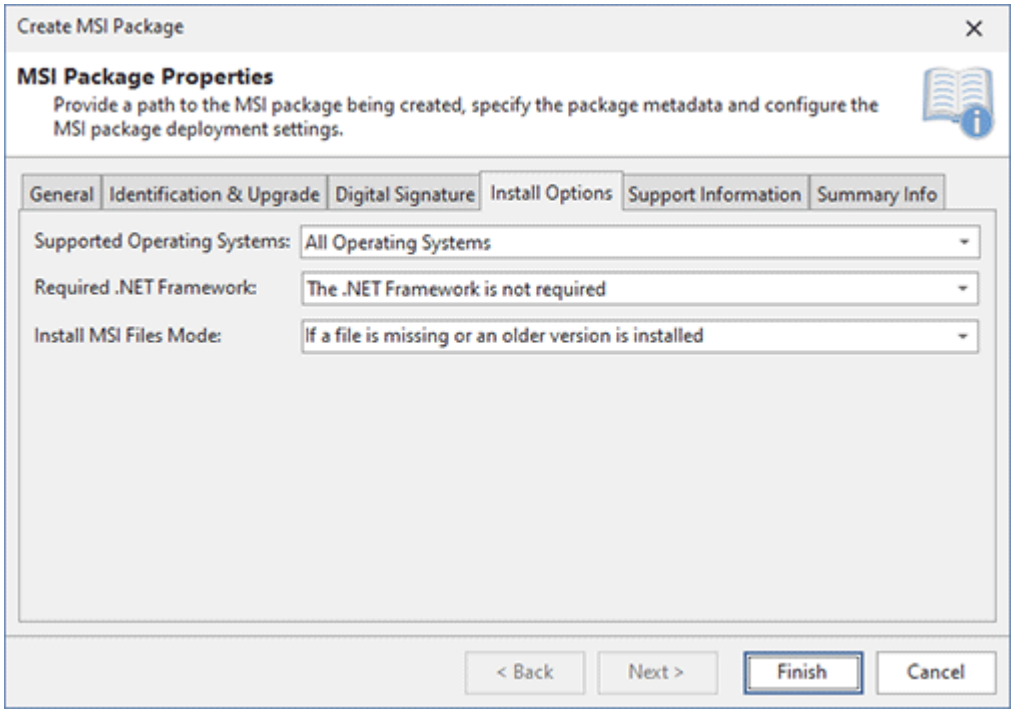
For the successful upgrade it is also required that the installed MSI package should have the same installation context (Current User/All Users) defined as the MSI used for update. It is also required that at least one of the first three version figures of the MSI package used for update differs from the version of the installed MSI package.




**Pic 3. Overriding the common package signing options**

Copyright © 2001-2023 EMCO. All rights reserved.

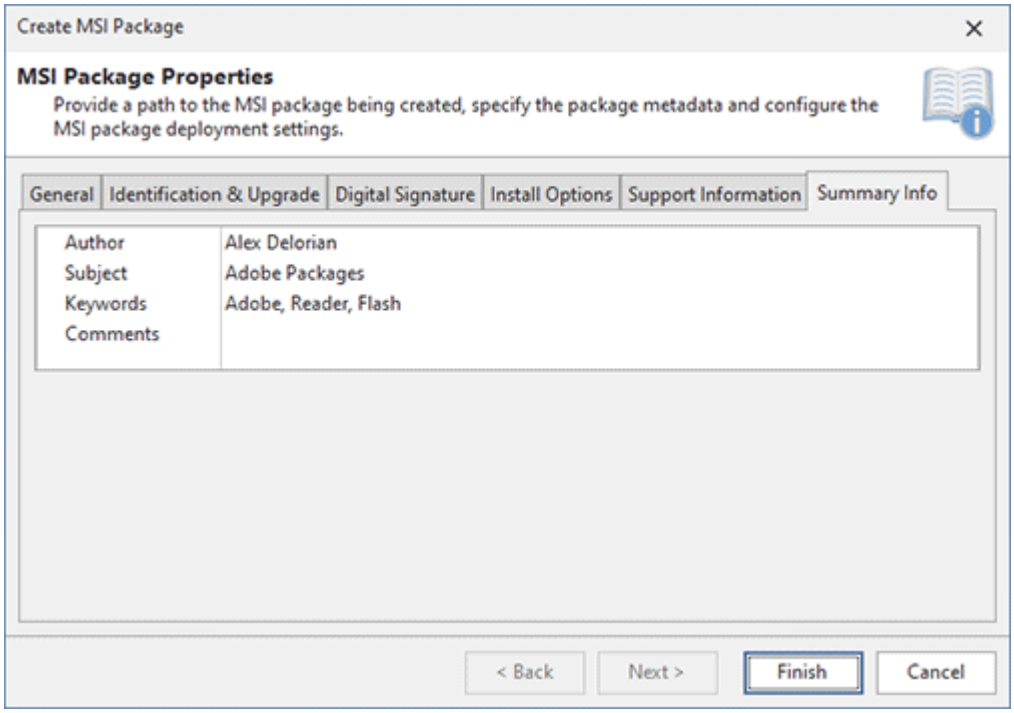
Before generating the MSI package, you can choose if the common package signing options should be used, or they should be overridden for this specific project Pic 3. As when defining the common settings, you can choose if the digital signature should be added to the package, select the signing certificate from the certificate storage and choose the time server used to generated a digital signature time stamp.




**Pic 4. Configuring Install Options**

*Copyright © 2001-2023 EMCO. All rights reserved.*

The next aspect of an MSI package configuration is install options Pic 4. You can specify the list of supported operating systems and choose the reinstall mode used if the files deployed by an MSI package already exist in the target location. You can also define the Microsoft .NET Framework versions required by deployed software packages – the Windows Installer will check for those versions and interrupt the deployment process if the required version is missing on a target PC.




**Pic 5. Defining Summary Information**

The last thing to define, that is also optional, is a set of support and summary properties. Those are available in the package properties and are displayed in the **Programs and Features** section of the Windows control panel. As for the support information, you can use the properties defined in the program preferences, or override them for this package.

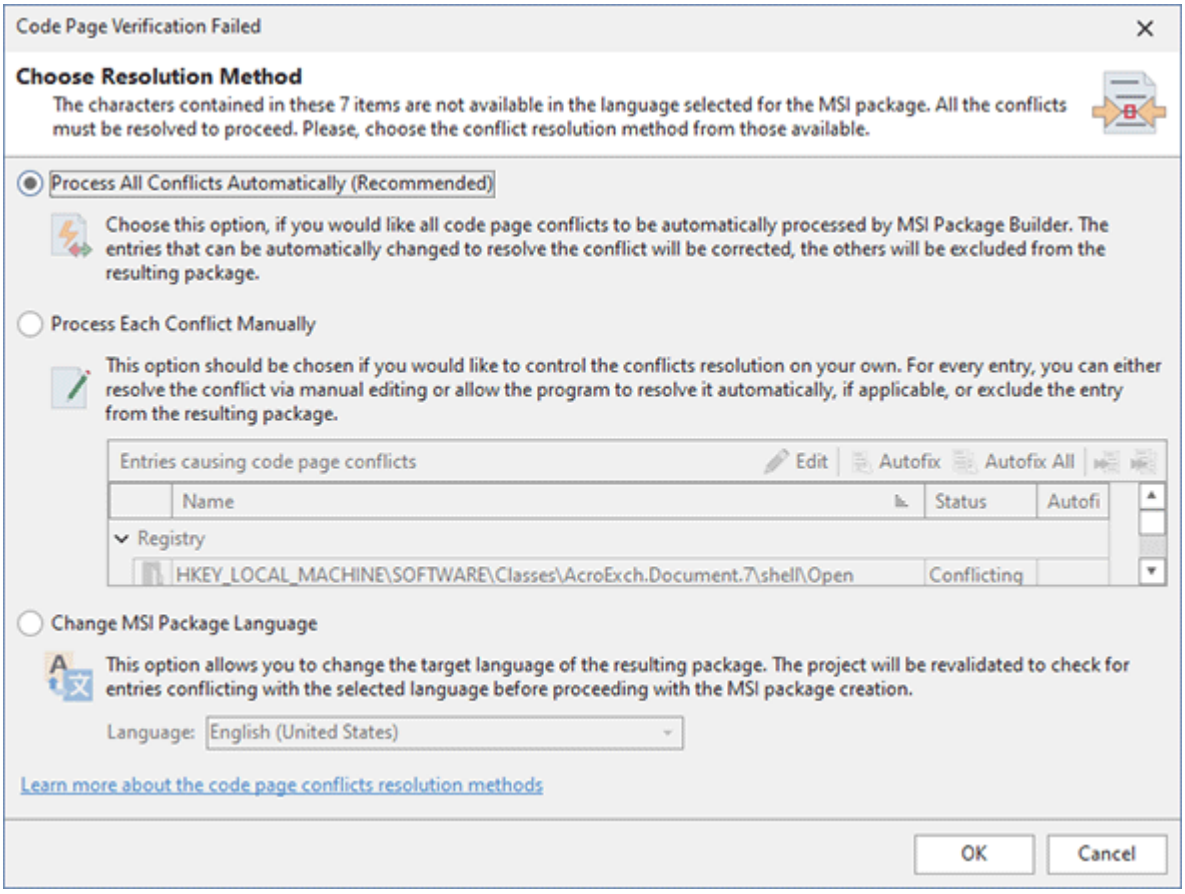
When all the required settings are configured, either within the **Project Details** view or within the package creation wizard, all you need is to provide a path to the location to save the resulting MSI package to and press **Finish**. As soon as the package is created, it is ready for deployment.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## MSI Package Language

The fact that any MSI package can target only one language is a well-known limitation of the Windows Installer technology. Within MSI Package Builder, the installer language is selected during the project configuration and can be changed within the **Project Details** view. As any MSI package is language specific, for a successful deployment, all the characters that may be present in any project entry should be available in the code page of the chosen language when the package is being generated.

To help you avoid language mismatches that are really hard to detect on your own, MSI Package Builder performs the code page verification before generating each MSI package and warns you if the test has not been passed successfully allowing you to choose the resolution mode for the detected conflicts Pic 1.




**Pic 1. Code page verification failure**

Copyright © 2001-2023 EMCO. All rights reserved.

There are actually two types of conflicts: those that can be resolved automatically and those that cannot. As for the conflicts that cannot be resolved automatically, you can either resolve them manually or exclude the entries causing such conflicts from the resulting MSI package. As you can see, the displayed warning allows you to choose among three options.

The first options, which is **Process All Conflicts Automatically**, means that it is up to MSI Package Builder to choose what to do with each entry causing a code page conflict. It is the easiest and the most transparent choice for the user. If his option is used, the program performs necessary changes to the entries containing conflicts that can be resolved automatically. The entries containing conflicts that cannot be resolved automatically are excluded from the resulting package. This options is suitable in most cases, but you should always check the resulting package for correctness in case you are letting MSI Package Builder to perform automatic conflicts resolution.

The next option, which is **Process Each Conflict Manually**, is used to let you decide what to do with each conflicting entry on your own. This option provides the maximum flexibility but requires a great responsibility. Using this method, you can choose for each entry whether you would like to edit it to resolve the conflict, or to apply automatic conflicts resolution to the entry (if applicable), or to exclude the entry from the resulting package. You can also choose to automatically resolve or skip all conflicts using the buttons on the toolbar above the entries list. For each entry in the list, you can see its status, which is either **Conflicting**, or **Resolved**, or **Skipped**.

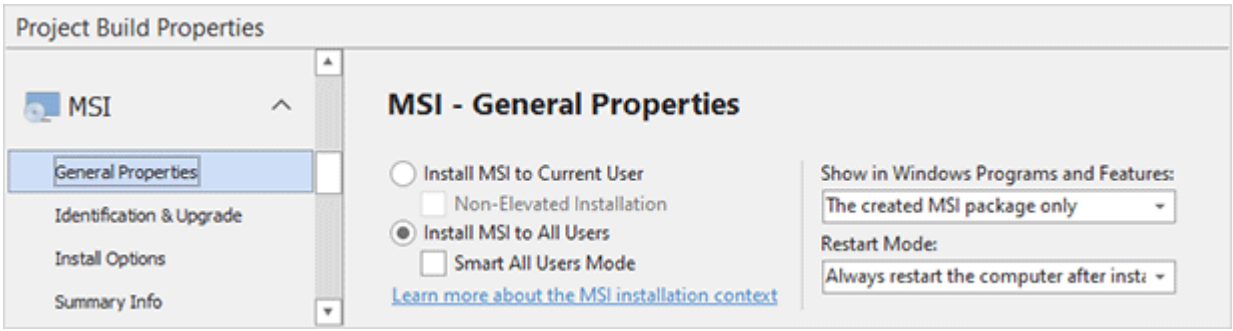
The last method of resolving code page conflicts is changing the target language of the resulting MSI package. The **Change MSI Package Language** option is used in such a case. As soon as you select the required language from the drop-down list, you are suggested to revalidate the project. If the validation is passed, the resulting MSI package is generated.

We have done our best to help you avoid problems caused by the language limitation of the Windows Installer technology and hope you will find the code page verification process with different conflict resolution methods useful.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## MSI Installation Context

Windows Installer can install a package on a computer within two installation contexts: per-machine and per-user. A per-machine package installation is required if all users of the computer need to access and use the application. Based on the context, Windows Installer automatically redirects the values of the folder properties and registrations to the locations for a per-user or per-machine installation. The installation context is specified when configuring the MSI Package Builder project Pic 1.




**Pic 1. Choosing the installation context**

When a package is installed for a current user, it is not visible in the **Programs and Features** section of the control panel for other users. In most cases, the deployed content is also accessible only for the user that deployed the package, though it is not always true. To install a package per-user, you should choose the **Install MSI to Current User** radio button during the project configuration.

The generated MSI packages can be installed by Windows users with administrative rights. If an MSI package is installed manually, the Windows User Account Control (UAC) prompt is displayed requiring permissions elevation to complete the MSI installation. Non-administrator users have to provide administrative credentials to install MSI packages. You can avoid seeing the UAC prompt and elevating permissions for installing an MSI package if this package was generated as **Install MSI to Current User** with the **Non-Elevated Installation** option enabled.

Non-elevated installations have some restrictions. If the **Non-Elevated Installation** option is enabled for an installation project that violates such restrictions, this option is skipped and a regular MSI that requires permissions elevation is generated. In this case, the restriction violations are reported in the **Log** view.

Copyright © 2001-2023 EMCO. All rights reserved.

Note that it isn't allowed to generate a non-elevated installation if the installation project contains any of the resources listed below:

| Resource Type | Resource Value |
|---|---|
| File System | CommonFiles64Folder,<br>CommonFilesFolder,<br>ProgramFiles64Folder,<br>ProgramFilesFolder,<br>System16Folder,<br>System64Folder,<br>SystemFolder,<br>WindowsFolder,<br>CommonAdminToolsFolder,<br>CommonDesktopFolder,<br>CommonProgramMenuFolder,<br>CommonStartMenuFolder,<br>CommonStartupFolder,<br>FontsFolder,<br>UserProfilesFolder |
| Registry | HKEY_LOCAL_MACHINE,<br>HKEY_USERS |
| Environment Variables | System Environment Variables |
| Services | All |
| Assemblies | All |
| Printers | All |
| Drivers | All |
| Windows Firewall | All |
| Custom Actions | SAM Licenses |

To create a package targeting a per-machine installation, you should choose the **Install MSI to All Users** radio button. As for the per-machine installation, there is some specifics in handing the user-specific folders and registry keys defined in the package. By default, when a per-machine installation is deployed and the package contains user-specific files and registry entries, such files and entries are created only for the user running the deployment. For other users, only the common files and registry entries are deployed. MSI Package Builder allows you to configure the package in such a way that user-specific entries will also be installed for other users on their first log-on after the package deployment. This feature is enabled by means of the **Smart All Users Mode** box.



For the smart per-machine installation to work correctly, the MSI package should be accessible to users logged in future via the same path as used during the initial deployment. In case the package is not available, the user-specific files and registry entries won't be created for other users.

To obtain proper deployment results, it is recommended that the default installation context be chosen carefully taking into account the specifics of each and every package.

Copyright © 2001-2023 EMCO. All rights reserved.

# Signing Packages

Digitally signing files helps protect against changes to a file by validating that a hash of the current file matches the hash stored in the digital signature. Digital signatures also help verify that a package came from a particular publisher by encrypting the hash with the publisher's private key. Verifying the signature using the publisher's public key or a trusted certificate authority that signed their public key validates the publisher.

You can sign Windows Installer packages to help guarantee that users know if your packages have been modified and that they came from you, the publisher. Windows Installer validates that a package hasn't been changed if it contains a digital signature when attempting to install it. Properly signed MSI packages can be installed via GPO even within the environments with strict security policies. As for an MSIX/AppX package, it is required to be digitally signed to be deployed.

## About Digital Signatures

A digital signature is based on a signing certificate. A certificate is a set of data that completely identifies an entity, and is issued by a certification authority only after that authority has verified the entity's identity. The data set includes the entity's public cryptographic key. When the publisher of a package signs the package with its private key, the installer can use the publisher's public key to verify the publisher's identity.

In order to perform a package signing operation, both private key and signer identification information must be supplied. The digital certificate used in the signature usually supplies the signer identification information, however. Thus, the private key must be supplied through some other means. Additionally, the signature must include the certificate chain for the cryptographic service provider (CSP), up to a root certificate trusted by the user, in order for the signed file to be authenticated. So in all, there are several items that need to be provided in order to generate a digital signature.

Microsoft has developed a certificate store technology to reduce the above complexity. Using this technology, when a user enrolls to obtain a certificate, they specify the private key information, the CSP information, and the certificate store name for the certificate. The certificate will then be stored in the certificate store and be associated with the other items. When the user wants to sign a package, they only need to identify the certificate in the certificate store. The code-signing tool will retrieve the certificate, the private key, and the certificate chain for the CSP, all based on the specified certificate.

When signing a package, a trusted time server is used to generate a time stamp for a digital signature. This is performed, to guarantee that the package is signed with the certificate that is neither expired not revoked.

## Requirements for a Digital Certificate

For a digital certificate to be used by MSI Package Builder for signing the generated packages, the following set of requirements must be met:

1. The certificate must include the **Code Signing (1.3.6.1.5.5.7.3.3)** within its Intended Purpose.
2. The certificate's **Valid From** date must be less and the **Valid To** date must be greater than the package signing date.

*Copyright © 2001-2023 EMCO. All rights reserved.*

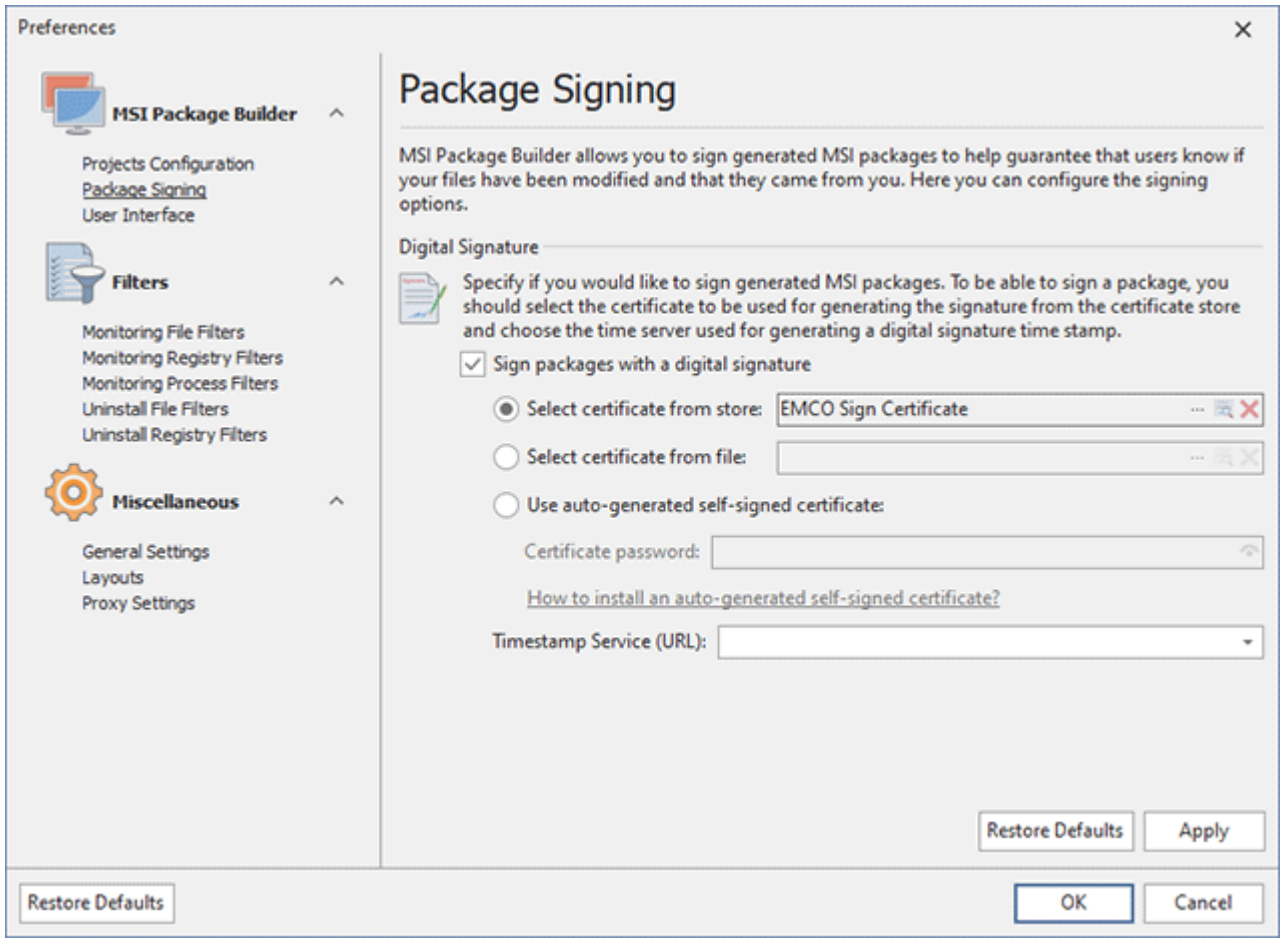
If you would like to use the certificate from certificates storage:

1. The digital certificate must be placed in the **Current User** certificates storage.
2. Both the private key and the signer identification information must be supplied.
3. The private key must be available together with the signing certificate in the certificate storage. In case you have a private key in a separate file, please use the tool provided by Microsoft for preparing the a private & public key pair for importing into the certificates storage as described here: *Pvk2Pfx*, *Combine PVK + SPC to PFX*.

In case the digital certificate does not meet the above-stated requirements, MSI Package Builder will not suggest that you use it for signing packages.

## Configuring Packages Signing

MSI Package Builder allows you both to define the common packages signing configuration to be used for adding digital signatures to generated MSI packages and to override those settings for specific projects. The common digital signing options are specified on the **Packages Signing** preference page Pic 1, and the overriding feature can be used either when creating an MSI package, creating an MSIX/AppX package or in the **Project Details** view.




**Pic 1. Configuring the package signing options**

Copyright © 2001-2023 EMCO. All rights reserved.

In any case, if you enable the packages signing, you are suggested to select the signing certificate to be used for creating a digital signature and choose the time server for generating a digital signature time stamp.

The required certificate can be selected from those installed to the above-mentioned certificate storage. To select the certificate from store, check the **Select certificate from store** radio button and press the  button within the following field. The dialog will be displayed to let you choose the certificate from those available. When choosing the certificate, you can press the **View Certificate** button on the toolbar to view detailed information on the selected certificate. This information dialog can also be reached via the  button from the certificate choosing field when the certificate is already specified. To reset the certificate, you can use the  button.

If you would like to use the certificate stored within the private information exchange (pfx) file, check the **Select certificate from file** radio button press the  button within the following field. The file browser dialog will appear on the screen to let you provide the path to the file. Normally, private keys in pfx files are password encrypted, so you're going to be prompted for the password to use the certificate from the specified file for signing packages. The same as for the certificates chosen from store, you can open the certificate information dialog via the  button from the certificate choosing field and reset the certificate via the  button.

If you do not have a required certificate, as a workaround it is possible to generate a self-signed certificate for the package being generated. In such a case the pfx file containing the automatically generated certificate will be placed next to the generated package and must be installed on the client machines before deploying the package. Please refer to the How to install the auto-generated self-signed certificate? section for details. To use the described approach, check the **Use auto-generated self-signed certificate** radio button and provide the password to protect private key to the **Certificate password** field.

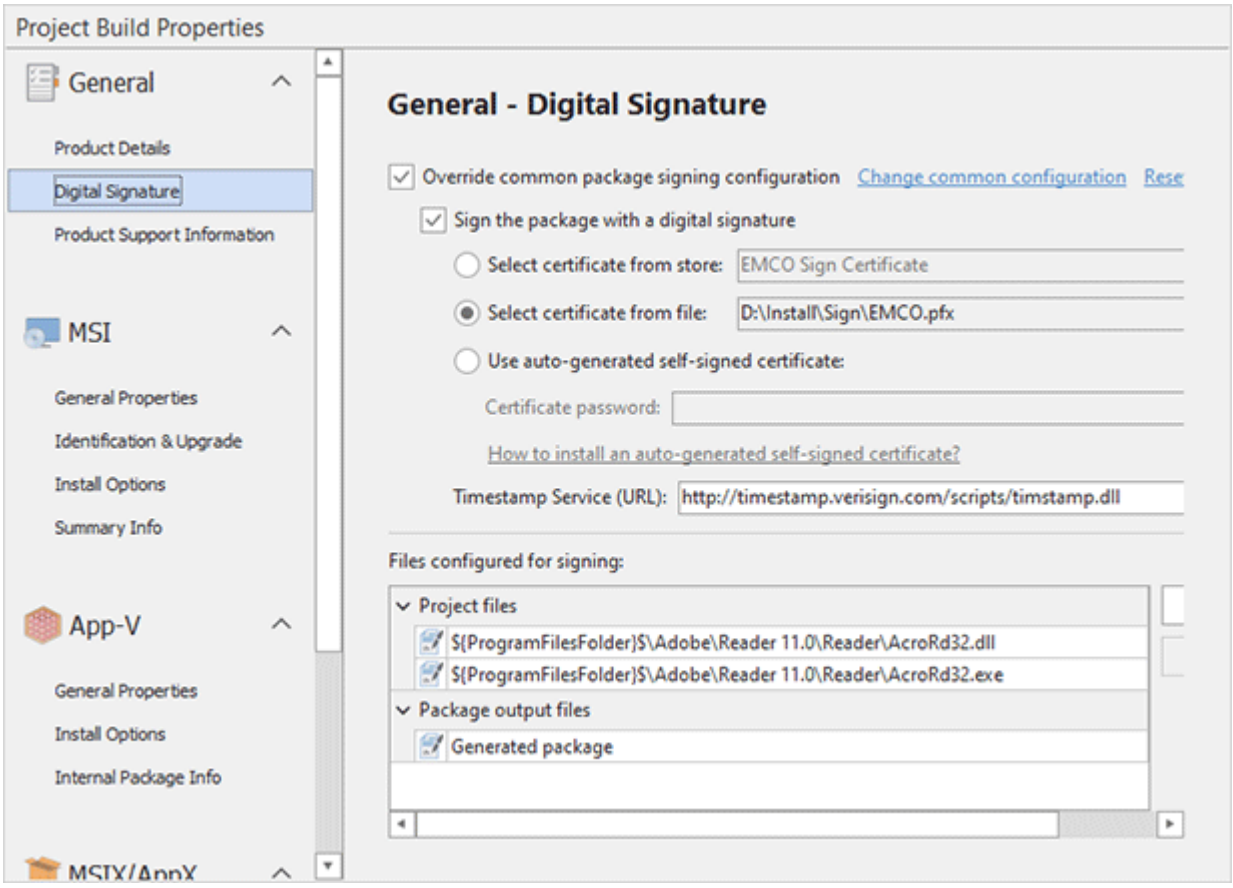
As for the time server, you can either choose the one from those predefined in the **Time Server** field or provide the address of another trusted server that can generate a time stamp for digital signatures. A time stamp should always be added when signing a package. Although it is strongly recommended that a digital signature time stamp be added immediately when signing packages with MSI Package Builder, you can leave the **Time Server** field empty, thus skipping the time stamping. In case a time stamp is not added, it is possible to time stamp a signed package in future with the help of the sign tool.

If the package signing is enabled, MSI Package Builder will add a digital signature using the specified signing certificate and chosen time server when a package is generated. If there are any problems occurred during the signing process, they are added to the **Log** view.

## Project Files Signing

When the package signing is enabled, the program signs a generated package, but you can configure the program also to sign the files deployed by the package. You can configure files to be signed in **Project Build Properties**. Select a project in the **Projects** view and open the **General** tab of the build properties. In the **Digital Signature** section you can find the list of files configured for signing. By default, the program signs the generated package only, but you can add the project files for signing Pic 2.

Copyright © 2001-2023 EMCO. All rights reserved.




**Pic 2. Signing project files**

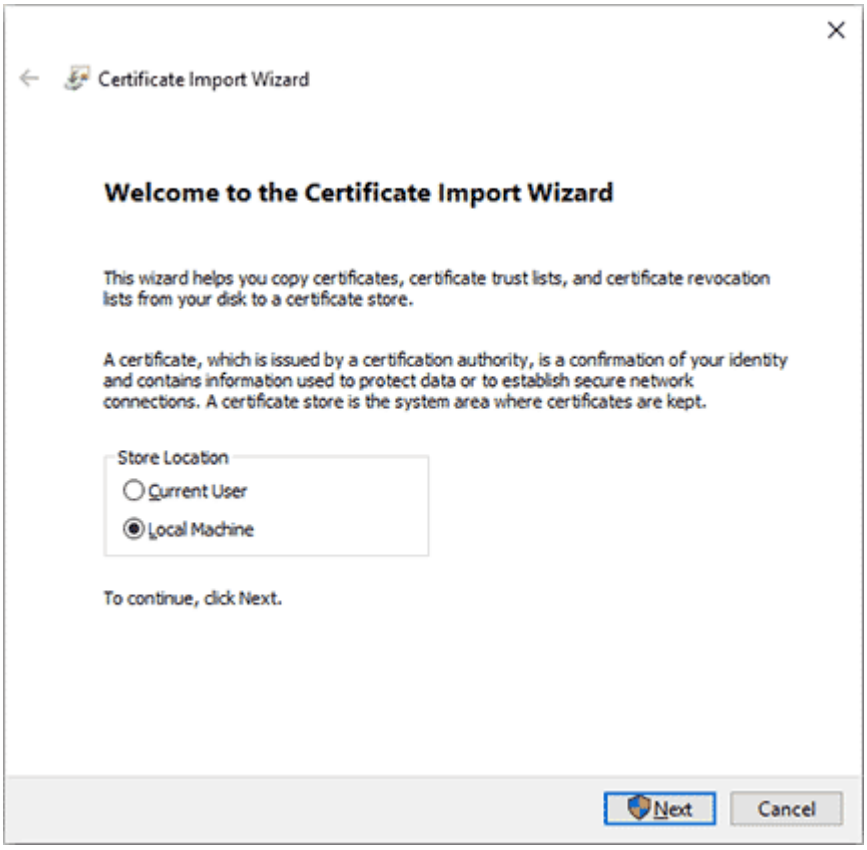
Press the **Add File** button and select a file available in the project, to add it to the list of files to be signed during the package generation. You can select EXE, DLL or SYS files because these file types support digital signing. To delete a file from the list, select it and press the **Remove** button.

## How to install the auto-generated self-signed certificate

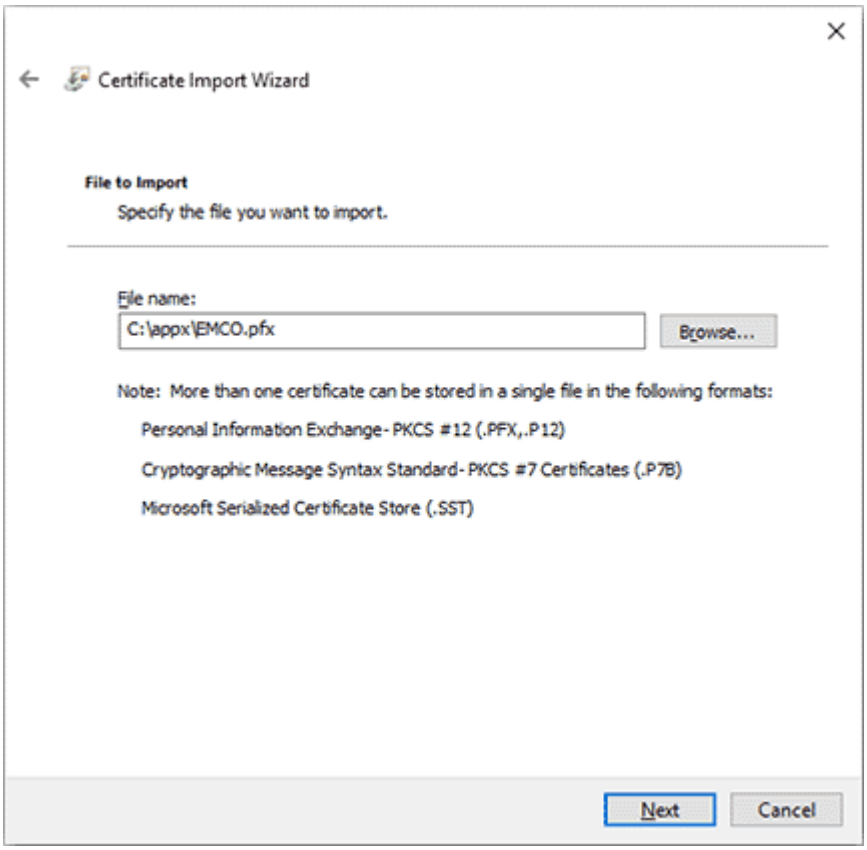
When using auto-generated self-singed certificate, it should be installed on client Machines before deploying generated packages. The following steps should be followed to install the certificate:

1. Double click the auto-generated pfx file. The **Certificate Import Wizard** will appear on the screen.
2. Select the **Local Machine** store and press **Next**.

*Copyright © 2001-2023 EMCO. All rights reserved.*

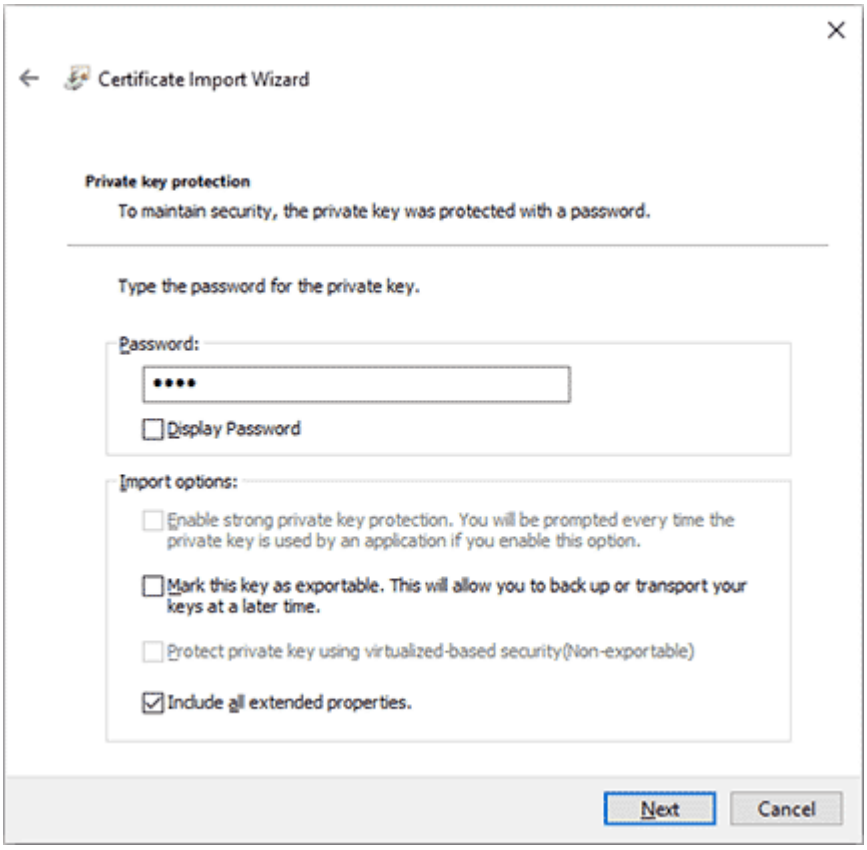



3. Double check that you are importing the certificate from the appropriate pfx file and press **Next**.

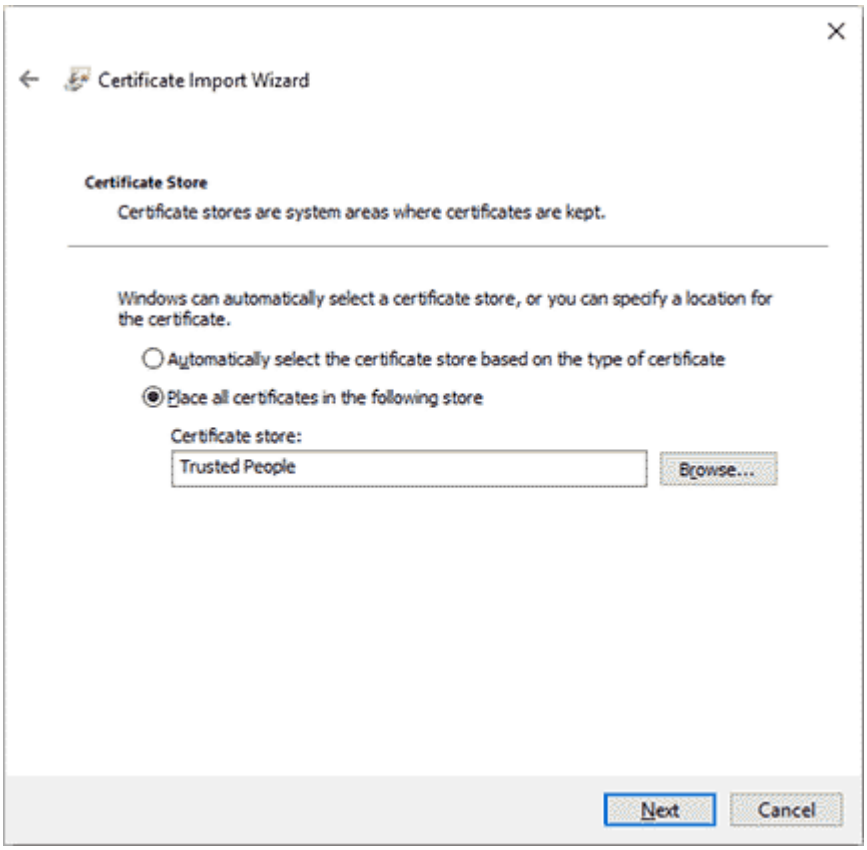



4. Provide the password you have specified in the **Certificate password** field to the **Password** field and press **Next**.

Copyright © 2001-2023 EMCO. All rights reserved.



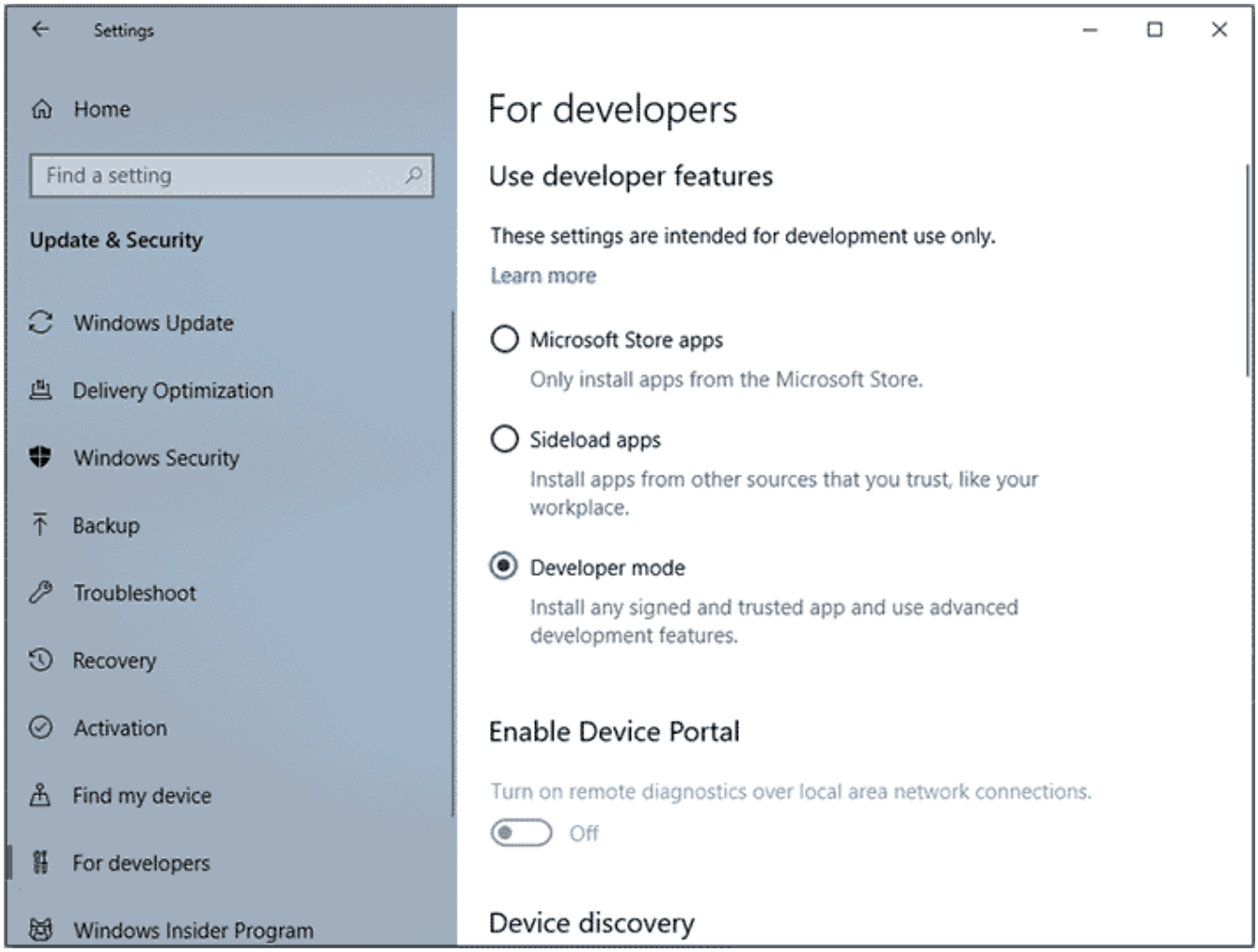

5. Select the **Place all certificates to the following store** radio button, press the **Browse** button and choose the **Trusted People** store. Then press **Next**.




6. After checking the import settings on the final wizard page, press **Finish**.

Copyright © 2001-2023 EMCO. All rights reserved.

As soon as the certificate is imported to the **Trusted People** store, the packages signed with the certificate are considered trusted thus fully secure, so you can successfully install MSI packages signed with such certificate via GPO. To allow deploying MSIX/AppX packages signed with the auto-generated certificate, it is also required to enable the **Developer mode** in the Windows settings Pic 3.




**Pic 3. Enabling the Developer mode**

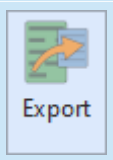
As you can see, using a self-singed certificate as a workaround has a set of limitations, but allows you to get a signed packaged that can be trusted within your organization or by specific PCs only.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Exporting a Project

All project data, including a set of files, registry changes, project settings and other data, is stored in the project. All projects are stored on a local hard drive in the project storage location configured on the **Projects Configuration** page of the program **Preferences**. To transfer projects from one computer to another or to back up projects, the program allows you to export the project contents as a single project archive file.

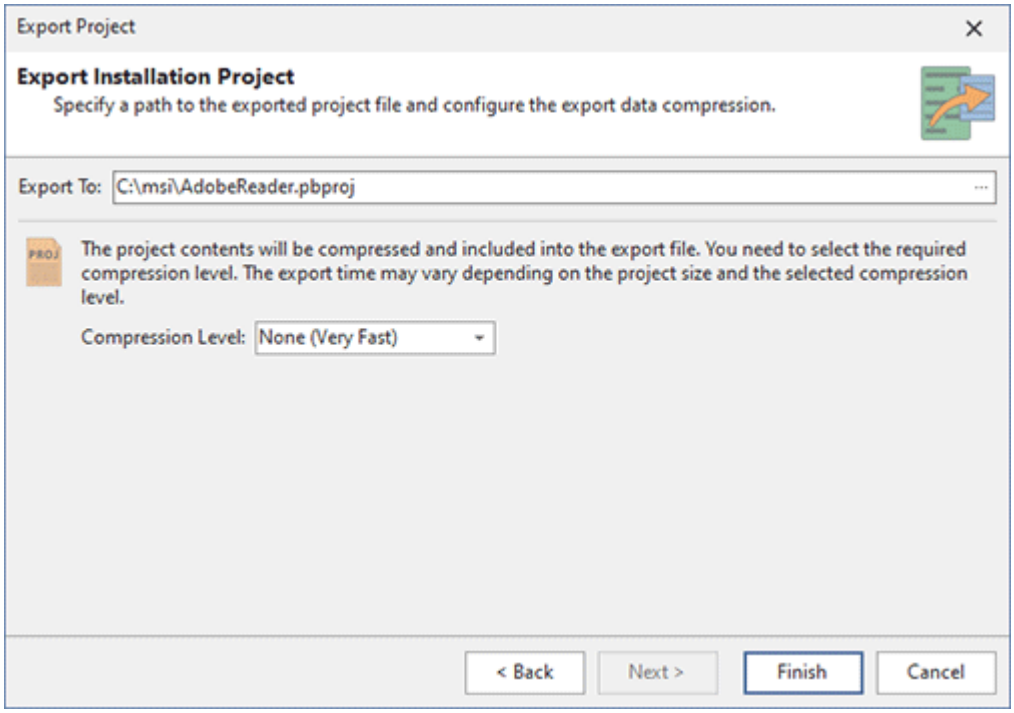
The export option is available as the **Export Project...** action on the project context menu in the **Projects** view. In addition, to export a currently opened project, you can click the **Export** button from the **Project Management** group on the **Home** ribbon page.




**Export**

The **Export** button from the **Project Management** group on the **Home** Ribbon page should be used to export all project contents to a compressed file.

When you initiate project export, the **Export Project** wizard is opened, where you can configure the export operation Pic 1.




**Pic 1. Exporting a project**

On the displayed dialog, you should specify a path to the export file and select the file compression level. The export file size depends on the selected compression level. Hence, the smaller output file you need, the higher compression level you should select, but exporting project data with a higher compression may require more time.

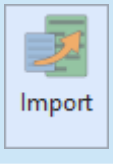
Copyright © 2001-2023 EMCO. All rights reserved.

## Importing a Project

With MSI Package Builder, you can not only create new packages but also modify existing ones using the project importing feature. You can import both project archive files and existing packages. The program allows you to select a package you want to modify, import it to a new project, perform the changes needed and build a new package. This feature is designed to handle the situation, when you have generated a deployment package, deleted the project, but need to make some changes to the package or upgrade it.

Though it is guaranteed that the import of the deployment package built with any previous version of MSI Package Builder is fully supported, some MSI, App-V and MSIX/AppX features are not available in the MSI Package Builder import technology. So if you are importing a generic deployment package, please check the import results and a newly generated deployment package carefully.

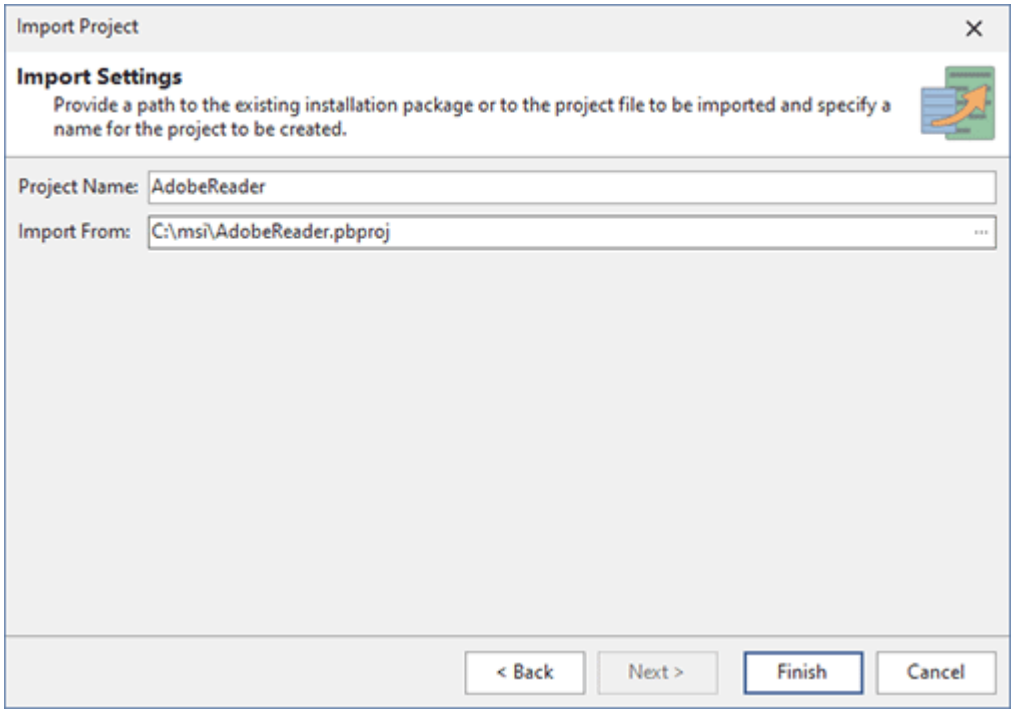
The feature of creating a new deployment package based on an existing deployment package is available in the **Project Setup** wizard. Alternatively, you can use the **Import** button from the **Project Management** group on the **Home** Ribbon page or the **Import Project** link in the **Product Actions** group on the **Welcome Screen**.




**Import**

The **Import** button from the **Project Management** group on the **Home** Ribbon page should be used to create a new MSI Package Builder project based on an existing deployment package or a project archive file.

In any case, you will reach the page where you are proposed to provide the path to the existing deployment package to import and a name for the project to be created as a result Pic 1. The project allows you to import *.msi, *.appv, *.appx, *.msix and *.pbproj files.




**Pic 1. Importing a project**

When the import is complete, you can see the project that contains the changes performed by the specified deployment package created in the **Projects** view.

Copyright © 2001-2023 EMCO. All rights reserved.

# Chapter 6: MSIX Packaging

MSIX is a new packaging format introduced by Microsoft in 2018 as a successor of the AppX technology. It was designed to deploy Windows applications on desktops, servers, mobile and other devices running Windows 10 or later OS. Applications deployed as MSIX packages run in a container, so that applications and their child processes are isolated. File system and registry operations are redirected and some security restrictions are applied.

You can use MSI Package Builder to create new MSIX packages from scratch or repackage existing Windows applications to the MSIX format. Application packaging into MSIX works on the same way as packaging to MSI, so you can use installations monitoring to create a new MSIX package, as described in the previous chapters. Note that MSIX applications work in isolated containers, so if the original application wasn't designed to work in isolated environment, you may need to configure MSIX fixups to let it work properly, as described in the following chapters.

## What's Inside

Creating an MSIX/AppX Package
MSIX Applications Configuration
MSIX Fixups

Copyright © 2001-2023 EMCO. All rights reserved.

## Creating an MSIX/AppX Package

MSI Package Builder is shipped with a full support of currently available features of MSIX, the packaging technology recently introduced by Microsoft. It extends the AppX one, used for deploying applications from Windows Store. The MSIX and AppX format has lots in common and both target Windows 10 platform, but MSIX is used starting from Windows 10 October 2018 update, and earlier the AppX format is used. MSI Package Builder generates either the MSIX or AppX package, depending on the targeting.

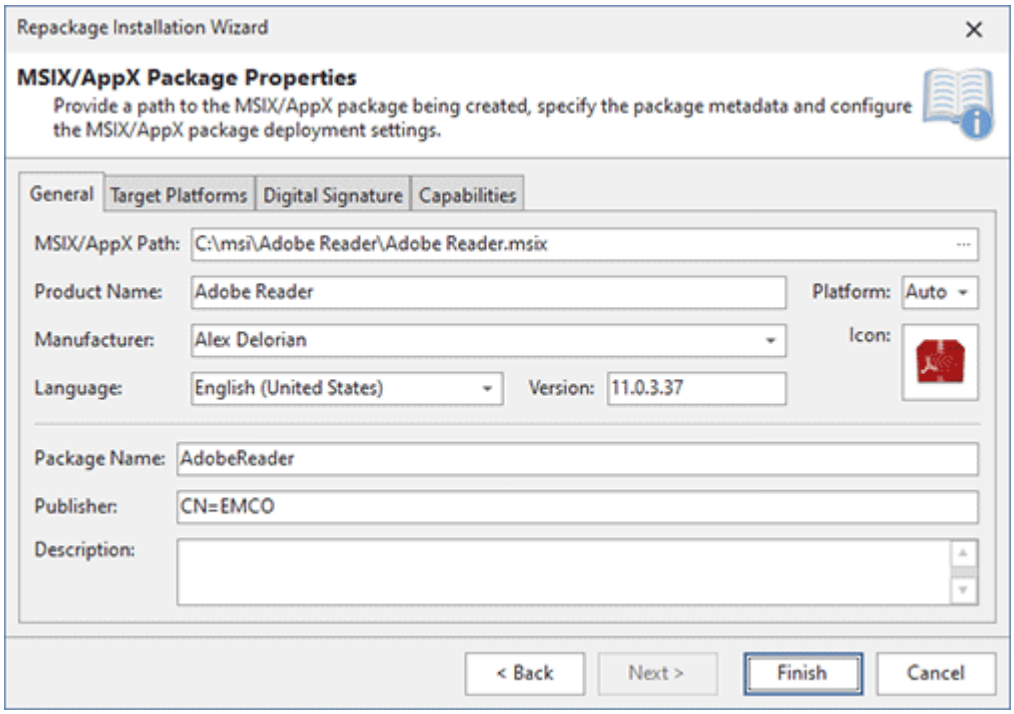
MSIX/AppX packages can be generated only on PCs running Windows 10 or later. If you run the program on earlier version on Windows, you cannot generate MSIX/AppX. In this case you need to install the program on a PC running Windows 10 or later, copy the installation project and generate MSIX/AppX there.




**Create MSIX/AppX Package**

The **Create MSIX/AppX Package** button from the **Builder** group on the **Home** Ribbon page should be used to generated an MSIX/AppX package based on the selected project.

To create an MSIX/AppX package manually, you can use the **Create MSIX/AppX Package** item from the **Projects** view pop-up menu or from the **Builder** group on the **Home** Ribbon page. When an MSIX/AppX package is created based on a project, the options for the package created are filled automatically using the values defined in the **Project Details** view, while during the repackaging process they are defined on-line, then saved to the created project. Let us take a look at the available options and describe each one starting with the general product information Pic 1.




**Pic 1. Specifying general MSIX/AppX package information**

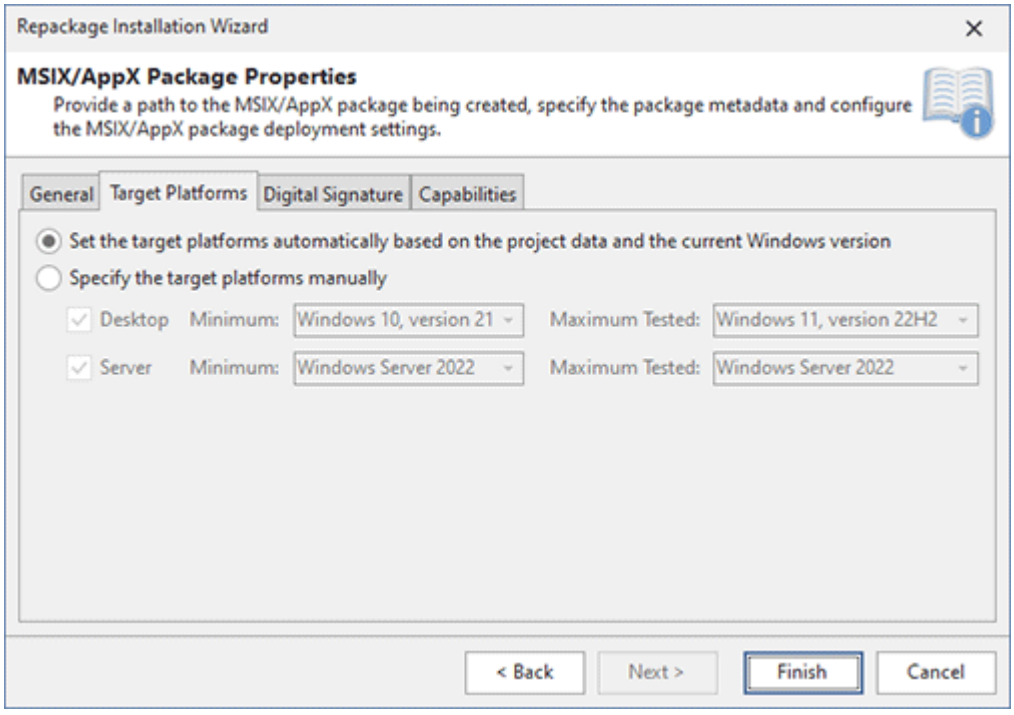
Copyright © 2001-2023 EMCO. All rights reserved.

The general information consists of the **Product Name**, **Platform**, **Manufacturer**, **Icon**, **Language** and **Version**. The product name, manufacturer and icon are used to display the product on the system. The platform is used to define if the installer is deploying an x86 or an x64 application. The **Language** field is responsible for defining the language and the encoding used by the installer package.

The version is the most important field of the general information because it is responsible for a correct package installation and upgrade. That means that every time you prepare an updated package using the same project you should change its version to a higher one. And it is also displayed on the system.

The **Package Name** and **Publisher** fields are responsible for the package identification. For packages signed with a certificate from store or from file, the **Publisher** field is automatically filled from the certificate. The **Description** field is used to provide a description for the package.

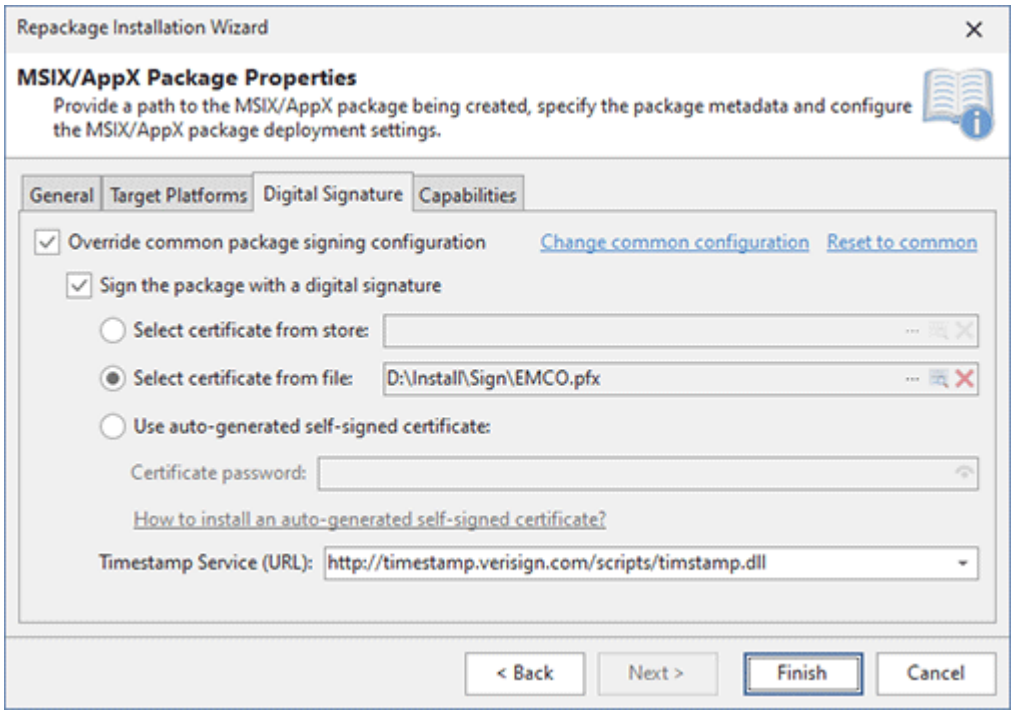
The next aspect of an MSIX/AppX package configuration is choosing the target platforms Pic 2. It's recommended to set target platforms automatically based on the project data and current Windows version. You can also specify the list of supported desktop and server operating system versions manually. When you create an MSIX/AppX package the output package format depends on the selected target platform. If you select a Windows platform that supports MSIX, the program will generate an MSIX package. For earlier versions of Windows that don't support MSIX, an AppX package will be generated.




**Pic 2. Defining the Target Plarforms**

Copyright © 2001-2023 EMCO. All rights reserved.

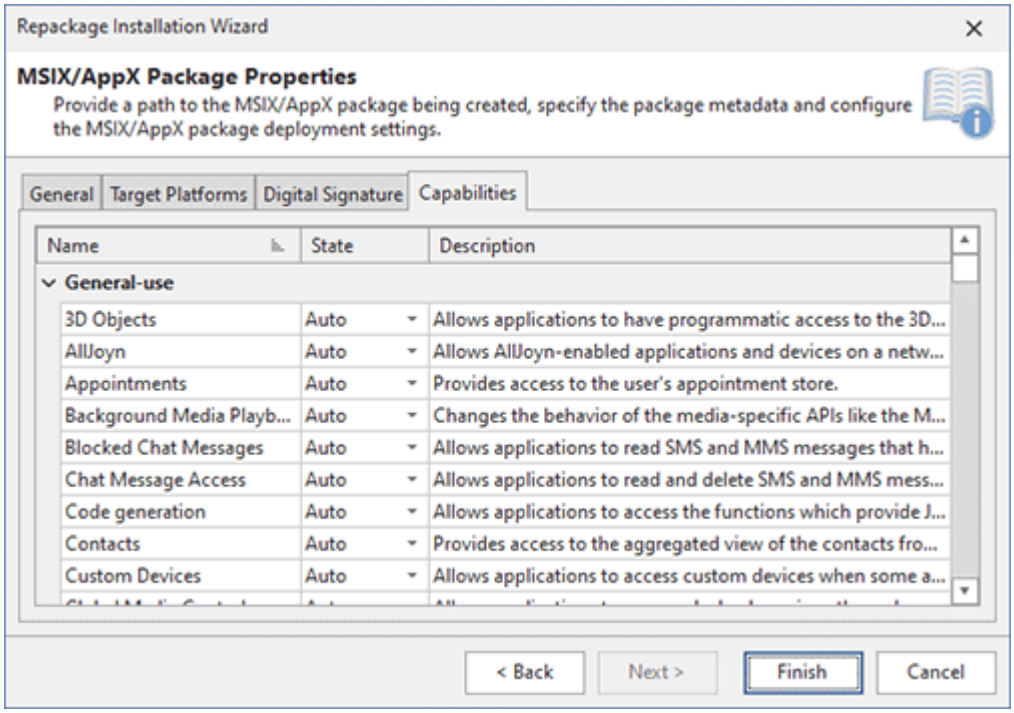
Another important thing to configure is a digital signature as MSIX/AppX packages cannot be installed until they are digitally signed. You can use the configuration defined in the program preferences or override it for this package Pic 3.




**Pic 3. Configuring Digital Signature**

Copyright © 2001-2023 EMCO. All rights reserved.

To create an MSIX package you need to declare its capabilities, i.e. access to APIs or resources, such as pictures, music, camera or microphone. Some capabilities provide applications with access to sensitive resources. These resources are considered sensitive because they can access the user's personal data. You can enable or disable required capabilities manually, if required. By default capabilities are set to auto. If a package contains resources that require specific capabilities, those capabilities are enabled by default Pic 4.




**Pic 4. MSIX capabilities configuration**

After all the required settings have been configured, either within the **Project Details** view or within the package creation wizard, all you need is to provide a path to the location to save the resulting MSIX/AppX package to and press **Finish**. As soon as the package is created, it is ready for deployment.

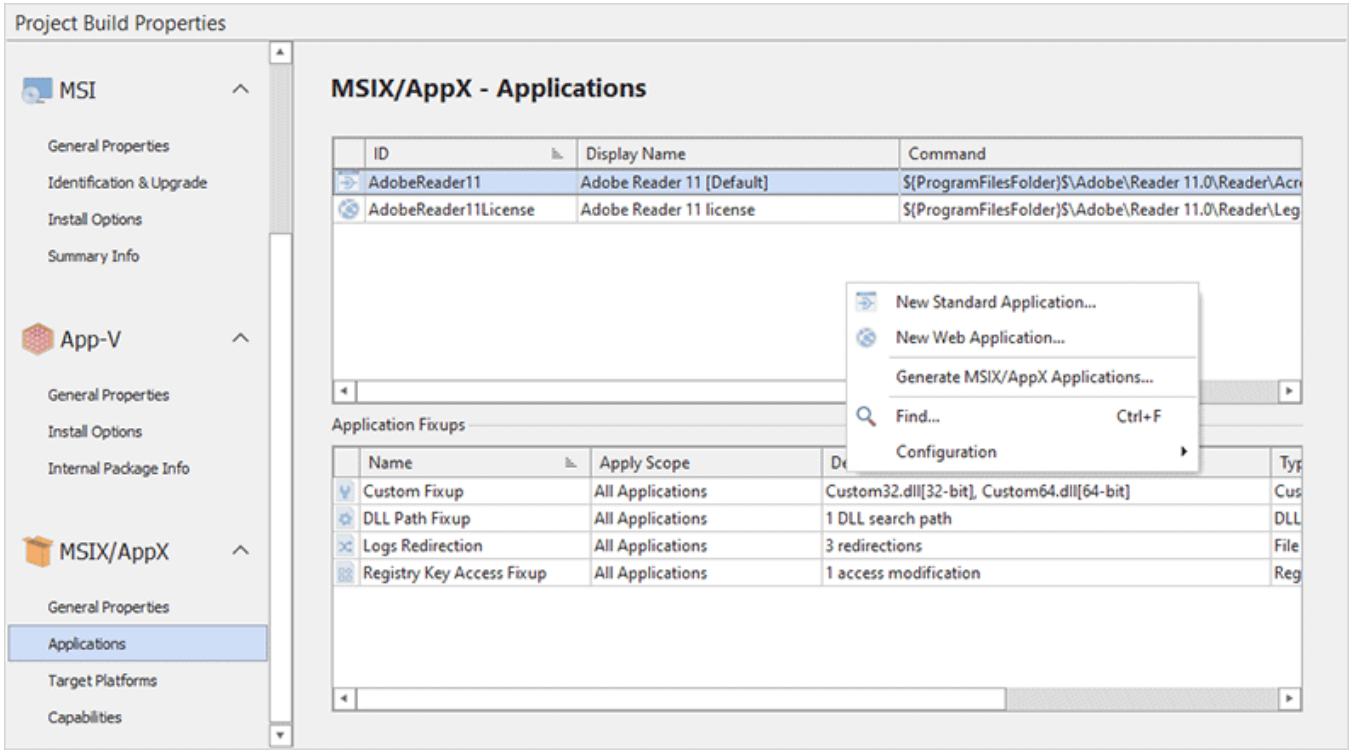
*Copyright © 2001-2023 EMCO. All rights reserved.*

## MSIX Applications Configuration

To build an MSIX package you need to create at least one MSIX application configuration, where the application represents an MSIX entry point. It specifies an executable file and its startup configuration that is executed when you run the application. This concept is similar to Windows shortcuts, which refers the program executable file and contains configuration of the program execution settings. MSIX application entry is displayed in Windows Start menu after MSIX deployment similarly to shortcuts created after installing a regular Windows application.

If you repackage an existing Windows application into MSIX using monitoring, MSI Package Builder automatically detects all shortcuts and prompts you to select at least one of them as an MSIX application. All settings for the selected shortcut is converted to MSIX application properties automatically.

MSIX applications can be configured on the **MSIX/AppX** tab of **Project Build Properties** for a selected project. **Applications** displays all configured MSIX applications and allows you to add and delete applications using a context menu Pic 1.



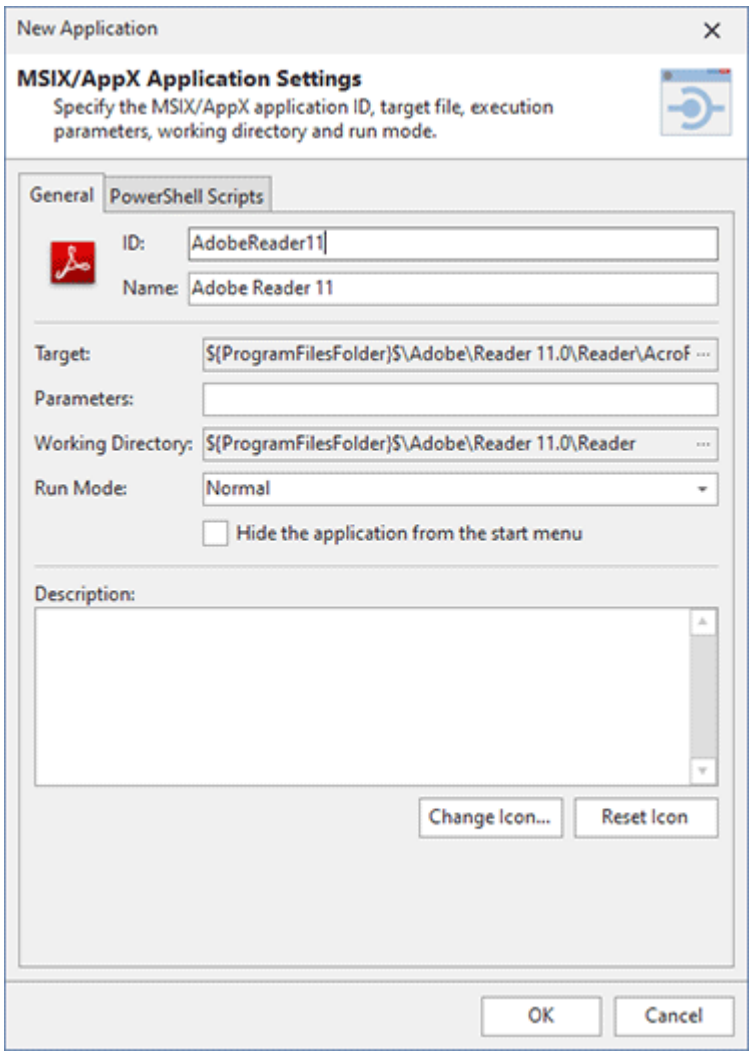

**Pic 1. MSIX applications**

Copyright © 2001-2023 EMCO. All rights reserved.

The **Generate MSIX/AppX Applications** action of the context menu allows you to use Windows shortcuts entries of the project to create corresponding MSIX applications automatically. Alternatively, you can create new standard and web application entries and provide the configuration manually.

## Standard Application Configuration

When you create a new standard application entry, you need to specify an application **ID** and **Name** to be displayed in the Windows Start menu. The **Target** field sets the path to an executable file. You can set an entry point of the application by selecting a file from those available in the project. The **Parameters** field specifies command-line parameters to be passed to the executable file when you run the application. You can leave them empty or set the required values. The **Working Directory** field sets a directory from those available in the project, to be used as a working directory of the application. The **Run Mode** sets whether the application window should be maximized, minimized or set to normal at start Pic 2.

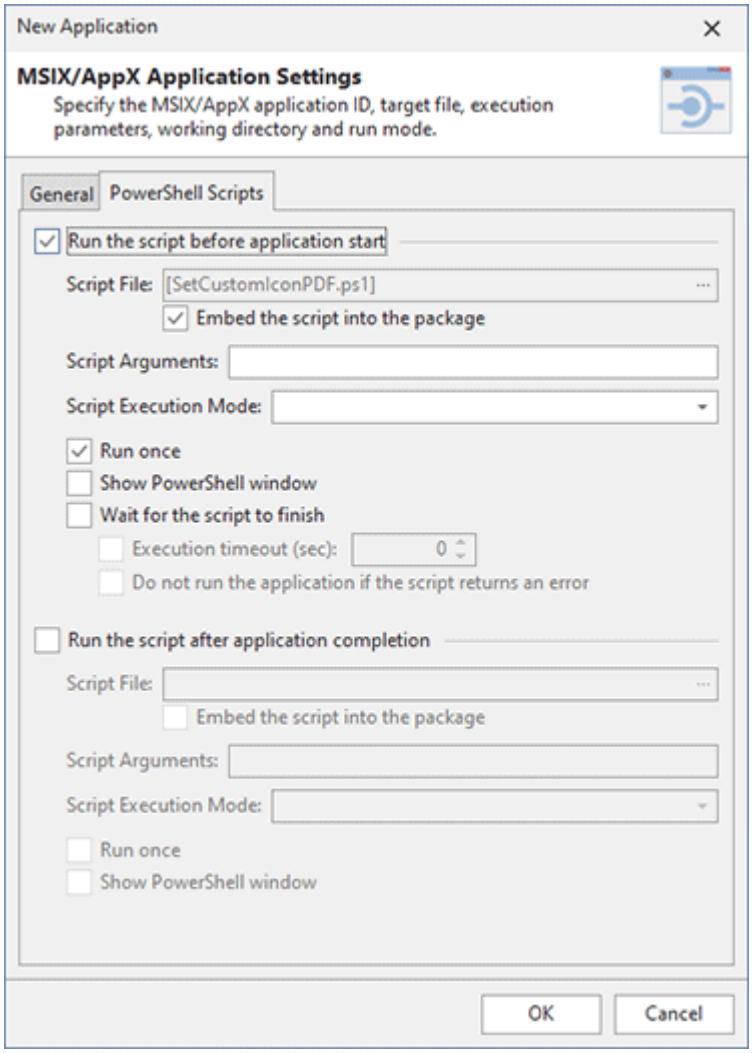



**Pic 2. Configuration options of the standard application**

Copyright © 2001-2023 EMCO. All rights reserved.

You can set a description for the created application and choose the application icon. Click the **Change Icon...** button and choose one of the standard icons available, or specify a path to a file to import an icon from it.

If required, you can configure an MSIX application to run a PowerShell script before the application start or after completion of the application. You can provide the corresponding settings on the **PowerShell Scripts** tab Pic 3. Set a path to a script using the **Script File** field. If the script isn't available on the machine where MSIX application will be executed, you can embed the specified script file to deploy the script together with MSIX. The **Script Arguments** field is used to set optional command-line parameters to be passed to the script. **Script Execution Mode** sets a power shell execution option. It is a safety feature that controls the conditions under which PowerShell loads configuration files and runs scripts. Use the execution mode suitable to run the selected script.




**Pic 3. PowerShell scripts configuration**

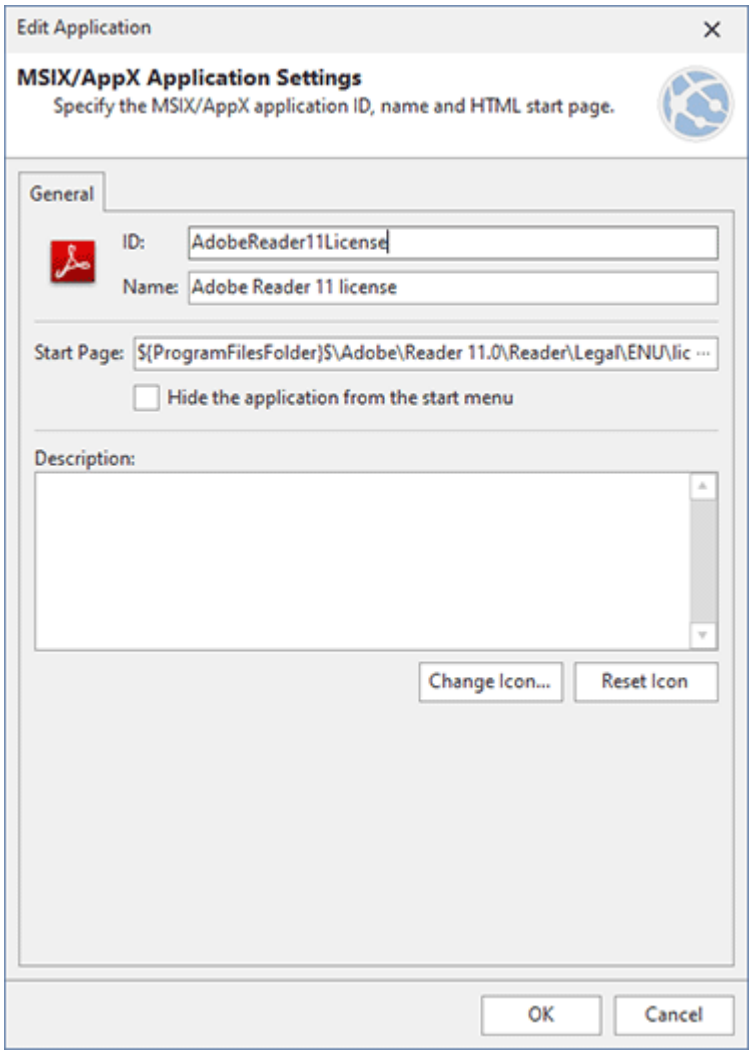
*Copyright © 2001-2023 EMCO. All rights reserved.*

The **Run Once** option allows you to run the script only once, at the first time when MSIX application is executed. If it is enabled, the script isn't executed on the second MSIX execution and all subsequent runs. If the **Show PowerShell** window option is enabled, the PowerShell console will be displayed when the script is running, so you can see the console output.

If you configure a script to run before MSIX application, you can configure the application to wait for the script to finish. The application can wait the script to finish as long as required, or you can set a timeout, so if this timeout is expired, the script is considered to exit with an error. It is possible to set if application execution should be skipped if the script returns an error.

## Web Application Configuration

If MSIX deploys a web application, you can configure the web application entry point by creating a new web application configuration Pic 4.




**Pic 4. Web application configuration**

To create a new web application set the application **ID** and the application **Name** to be displayed in Windows Start menu. In the **Start Page** field select an HTML file from those available in the project to be opened in the browser on application start. You can also specify a description and set the application icon if required.

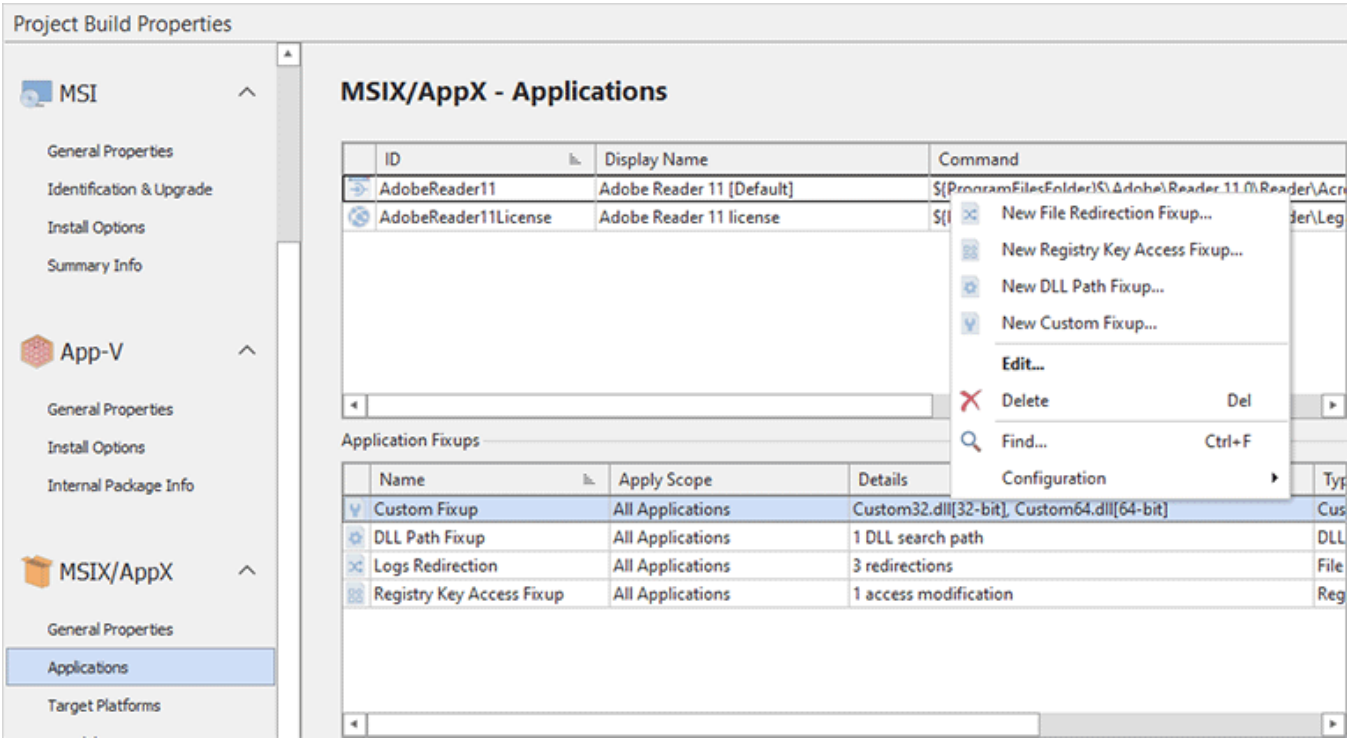
*Copyright © 2001-2023 EMCO. All rights reserved.*

## MSIX Fixups

MSIX apps work in a container that restricts access to some of the resources, so MSIX apps should be designed accordingly and don't use the restricted resources. When you package existing applications into MSIX, usually you work with traditional and legacy applications designed with no restrictions. If such an application is packaged to MSIX, it may violate container restrictions. As the result it reports some runtime errors caused by the fact that some resources cannot be accessed in MSIX container environment.

Those legacy applications can be successfully packaged to MSIX, but you need to configure so-called fixups that reconfigure system calls for a running application to comply with the restrictions of the MSIX container. MSIX fixups are based on Package Support Framework (PSF) provided by Microsoft to allow migrating legacy applications to MSIX. If a legacy application crashes at launch after packaging to MSIX, you need to configure fixups for it in the project and generate a new MSIX package to resolve the problem.

Application fixups can be configured on the **MSIX/AppX** tab of **Project Build Properties** for an opened project. You can add and delete fixups using context menu actions in **Application Fixups** Pic 1.




**Pic 1. MSIX application fixups**

MSI Package Builder allows you to configure different types of fixups, such as file redirection, DLL path and so on. You have to use a corresponding fixup, depending on the type of the runtime error reported by the running application. In the following chapters you can learn more about the available fixups.

### What's Inside

Copyright © 2001-2023 EMCO. All rights reserved.

File Redirection Fixup

Registry Key Access Fixup

DLL Path Fixup

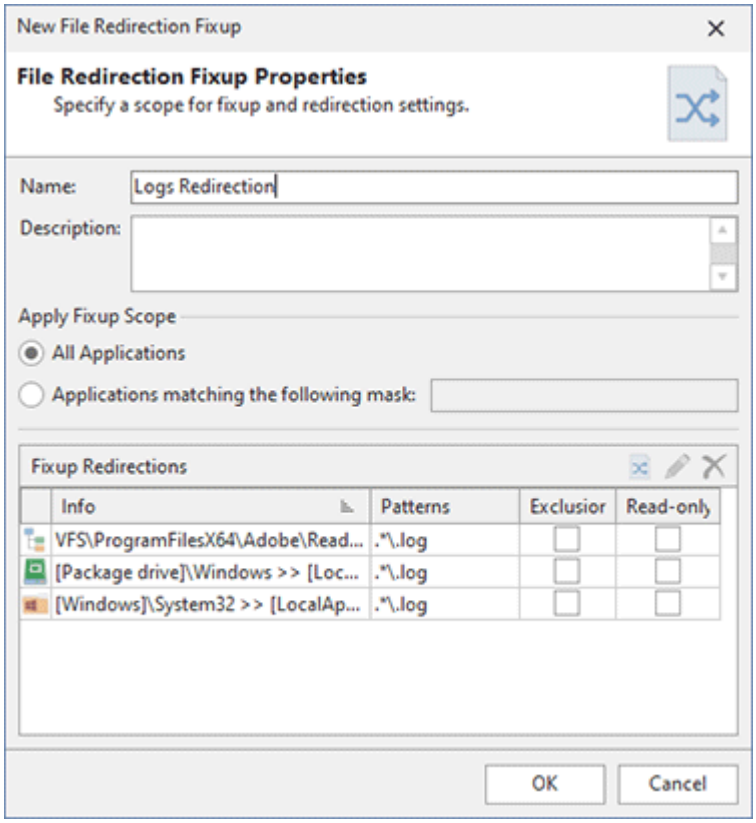
Custom Fixup

Copyright © 2001-2023 EMCO. All rights reserved.

## File Redirection Fixup

MSIX applications work with the virtualized file system (VFS) that provides access to files deployed by the package and to the system files. Access to the system resources is virtualized and redirected to specific locations, so that the application considers that it works with a system resource using a well-known path, while the call is redirected to the corresponding directory in the MSIX package deployment location. MSIX redirects only a predefined list of well-known directories, such as a Windows folder, Program Files folder, etc. If an MSIX application uses a file or folder that isn't redirected, an error is reported when you run the application. To resolve the error you need to create a file redirection fixup that helps MSIX to redirect a referred file. In this case the fixup provides configuration that describes how to redirect file system calls.

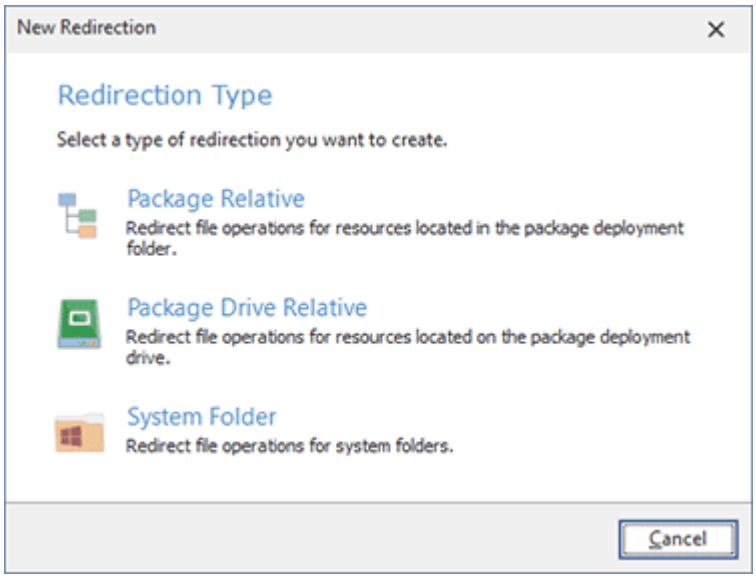
To create a fixup choose **New File Redirection Fixup...** in the context menu of **Application Fixups**. In the appeared dialog you can set the fixup settings. Set a name of the fixup in the **Name** filed and enter **Description** if required Pic 1. MSIX can contain multiple applications, so you can choose if the created fixup should be applied to **All Applications**, or apply it to specific applications only. Specify an application mask if you want to restrict applications to apply the fixup to. The mask is a regular expression (in the regexp format) to match the application names.




**Pic 1. Configuring a file redirection fixup**

Copyright © 2001-2023 EMCO. All rights reserved.

In **Fixup Redirections** you can specify one or multiple fixups. Click the toolbar buttons or context menu to add or delete fixups. When you create a new fixup, you can select the redirection type on the appeared dialog Pic 2.




**Pic 2. Select a fixup redirection type**

You can redirect resources, located in the package deployment folder, the package deployment drive and in a system folder. Configuration of all of those redirections is described below.

All redirection types have the common configuration settings, including **Pattern** and a couple of options. You can configure one or multiple patterns for each redirection. Patterns are regular expressions (specified in the regexp format) that are matched with the actual file or folder names of the resources to decide if the redirection should be applied for the resource.

How redirection works? When MSIX application makes a Windows API call to work with a file system resource, its path is matched with the base path (specified as **Redirect From**) together with **Patterns**. If the path is matched to the regular expressions, the API call is redirected to the path specified as **Redirect To**.

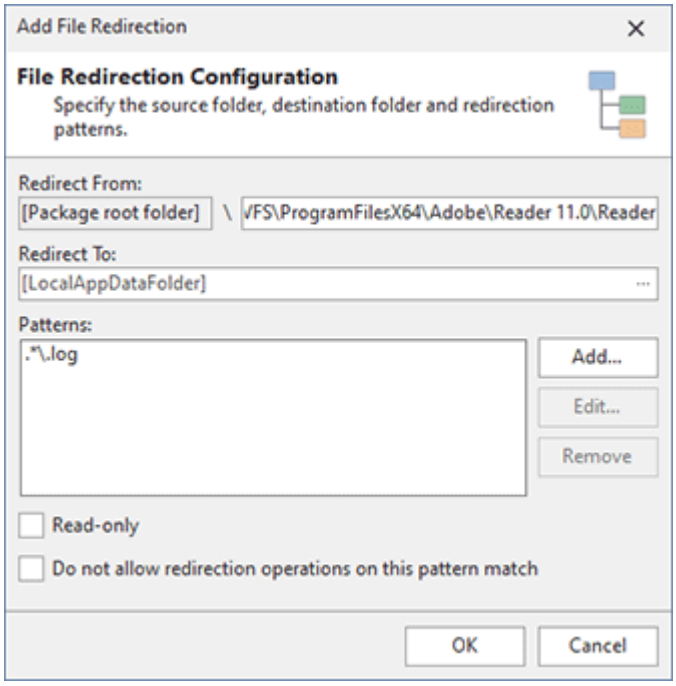
Copyright © 2001-2023 EMCO. All rights reserved.

The **Read-only** option specifies whether the redirected file system resources can be modified. If it is set to true, any changes are forbidden.

The **Do not allow redirection of this pattern match** option allows you to forbid redirection with the matched pattern. This option can be used to prevent redirections.

## Package Relative Redirection

This option allows redirecting files and folders, located in the package installation root, to the specified location. Specify a path in the package installation root folder set in the **Redirect From** field, to be matched together with the specified **Patterns** to make a redirection Pic 3.



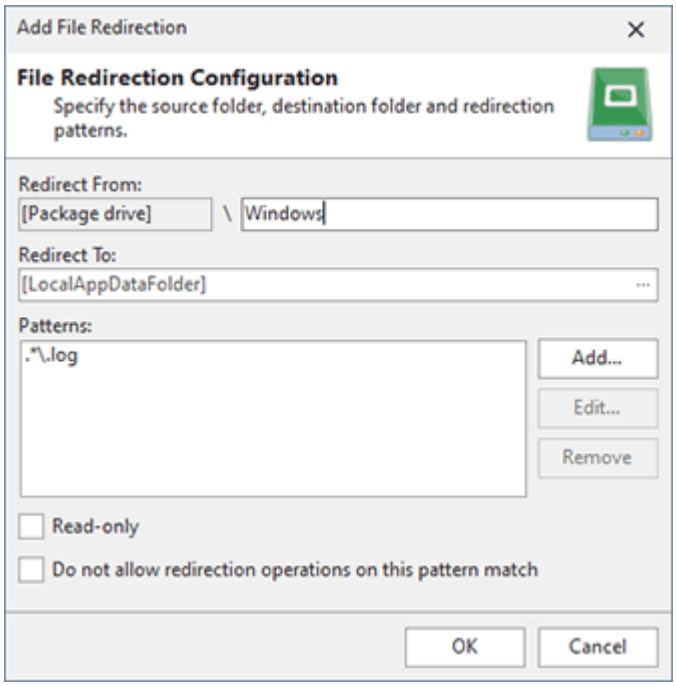

**Pic 3. Package relative redirection configuration**

Copyright © 2001-2023 EMCO. All rights reserved.

If the pattern matches, the call is redirected to the path specified in the **Redirect To** field. By default it is redirected to LocalAppDataFolder, but you can select any custom path.

## Package Drive Relative

You can use this option to redirect files and folders, located on the same drive as the deployed application. The **Redirect From** field allows you to specify a path on the drive to be matched together with **Patterns** with the actual paths to redirect MSIX file system calls Pic 4.



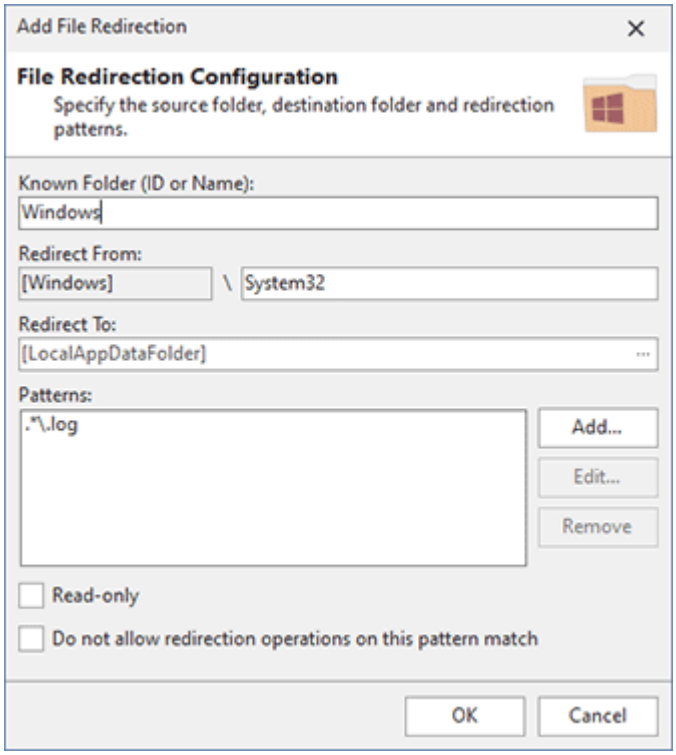

**Pic 4. Package drive relative redirection configuration**

Copyright © 2001-2023 EMCO. All rights reserved.

If the pattern matches, the call is redirected. The redirection path is specified in the **Redirect To** field. You can specify a custom path, or the call will be redirected to LocalAppDataFolder if the field is set to empty.

## System Folder

Use this option to redirect calls for system folders that aren't redirected automatically. Specify the folder ID or name and specify a subfolder, if required, in the **Redirect From** field Pic 5. To redirect a system folder you can specify its GUID as an ID field. You can find a list of GUIDs for system folders in *this atricle*.




**Pic 5. System folder redirection configuration**

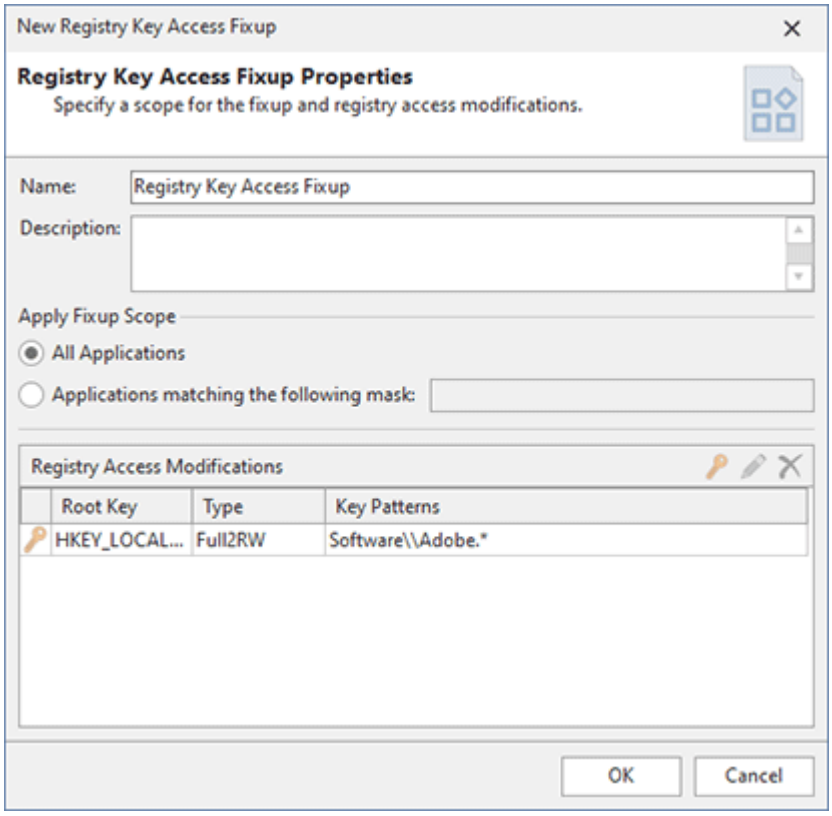
If the actual path matches the **Redirect From** together with **Patterns**, the corresponding MSIX file system call will be redirected to the folder specified in the **Redirect To** field.

Copyright © 2001-2023 EMCO. All rights reserved.

## Registry Key Access Fixup

MSIX applications work in a container with some restrictions, so that full access or write access to some registry keys can be forbidden. Some regular or legacy applications may request full or write access to the registry keys, so when such applications are packaged to MSIX, they report errors when trying to access registry. You can resolve this problem using the registry key access fixup. This fixup modifies certain registry calls that do not work due to the MSIX container restrictions, by modifying the call parameters to a form that would be allowed. Note that it doesn't guarantee that the application could work, but it helps in many cases.

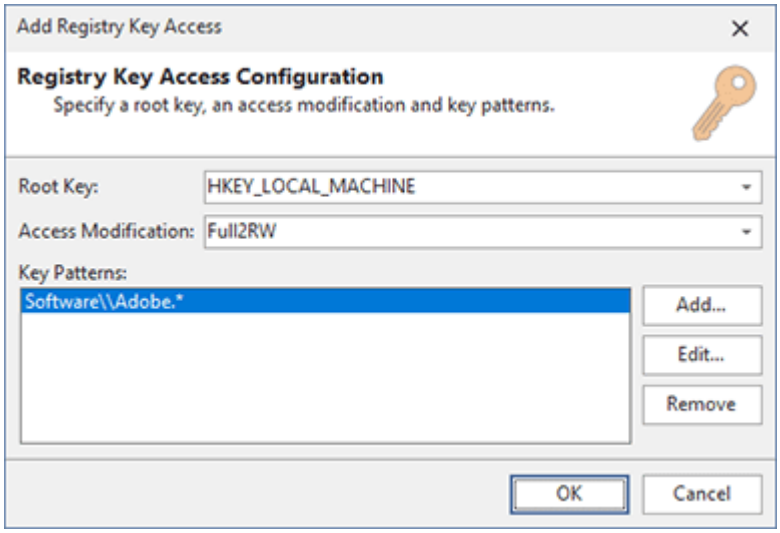
The registry key access fixup changes Windows API calls for accessing registry keys to remove permissions restricted by MSIX container. To create a fixup you need to specify a fixup **Name** and can set **Description** Pic 1. The fixup can be applied to all or specific MSIX applications only, so you can set a mask to configure applications to apply the fixup to.




**Pic 1. Creating a registry key access fixup**

Copyright © 2001-2023 EMCO. All rights reserved.

A fixup can include one or multiple configurations provided in **Registry Access Modifications**. To add a new configuration you need to click the **Add Registry Key Access** button on the toolbar or select the corresponding item in the context menu. In the appeared dialog you can set a configuration of the registry access modification Pic 2.




**Pic 2. Registry key access configuration**

To set a configuration you need to select **Root Key** to apply the configuration. Specify one or multiple **Key Patterns** that are represented as regular expressions (use the regexp format) and select **Access Modification Option**. If the MSIX application makes a Windows API call in the specified root key and the accessed registry key matches one of the configured key patterns, the configured access modification is applied.

The following access modifications are supported:

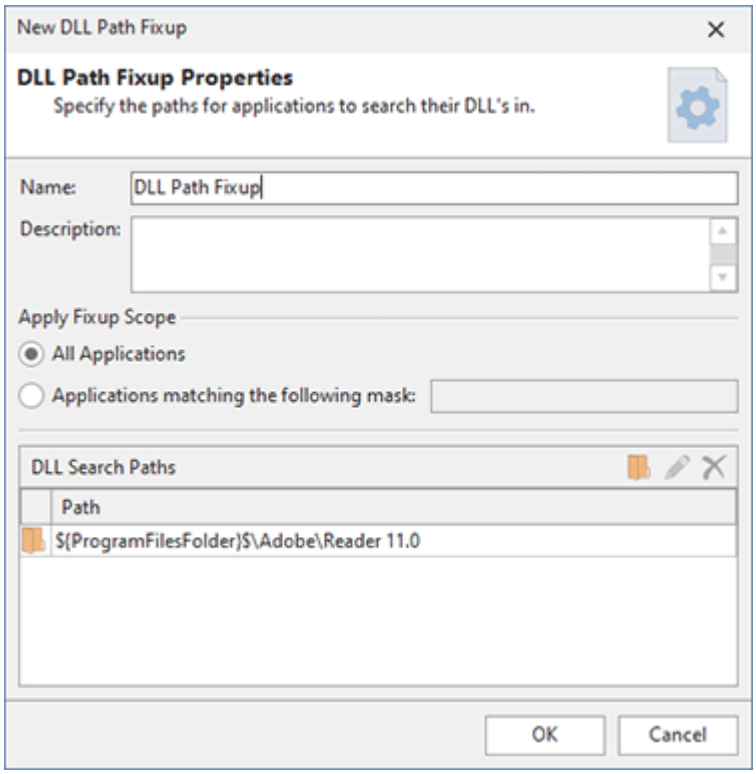
| Access Modification | Description |
|---|---|
| Full2RW | If full access is requested, modify to allow read-write access |
| Full2R | If full access is requested, modify to allow read access |
| Full2MaxAllowed | If full access is requested, modify to set maximum allowed |
| RW2R | If read-write access is requested, modify to allow read access |
| RW2MaxAllowed | If read-write access is requested, modify to set maximum allowed |

Copyright © 2001-2023 EMCO. All rights reserved.

## DLL Path Fixup

This fixup type allows configuring folders where DLL files should be looked for. If an application packaged to MSIX needs to load DLL files from locations restricted in MSIX container, you can use this fixup to resolve DLL loading errors.

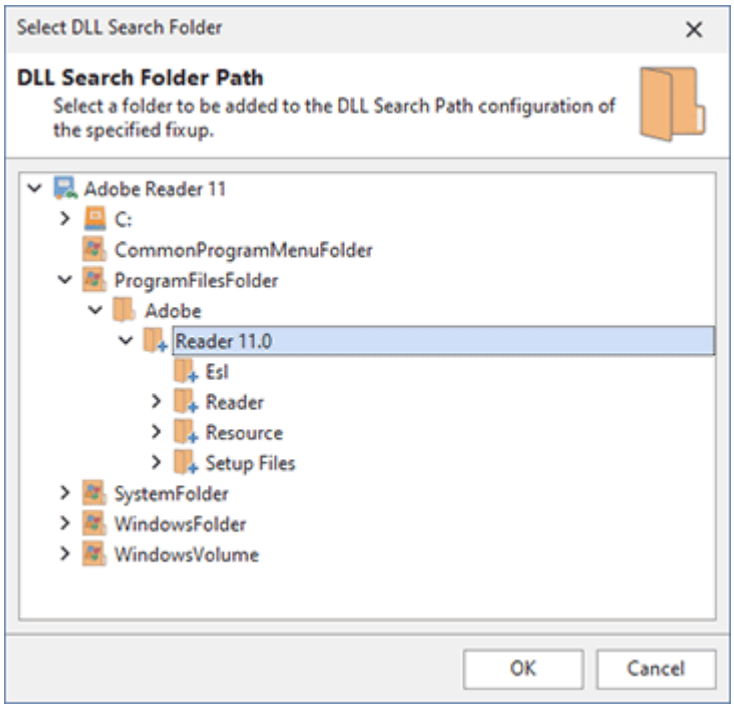
To create a DLL path fixup you need to specify the fixup **Name** and set **Description** if required. If the fixup should be applied to all applications, set the corresponding option, otherwise specify a mask to select applications for which the fixup should be applied Pic 1.




**Pic 1. DLL path fixup configuration**

Copyright © 2001-2023 EMCO. All rights reserved.

To configure this fixup you need to select one or multiple paths where DLLs should be located. You can choose the folders available in the project Pic 2.




**Pic 2. Select DLL search folders**

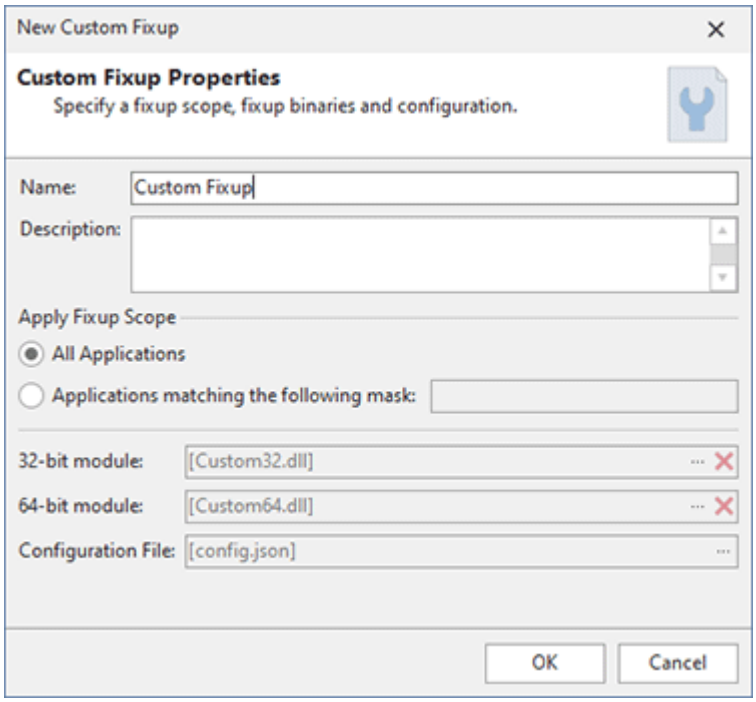
When the fixup is applied, the MSIX application will load DLLs from the specified folders.

Copyright © 2001-2023 EMCO. All rights reserved.

## Custom Fixup

In some cases existing fixups don't allow you to meet the requirements and resolve errors caused by the restricted access to resources in the MSIX container. In this case you can create a custom fixup that meets technical requirements of Package Support Framework (PSF). A custom fixup allows you to apply the created fixup for an MSIX package.

To create a custom fixup you need to specify its **Name** and set **Description** if required. You need to specify a fixup scope, so it should be applied to all applications or those applications that match the specified mask. The mask should be provided as a regular expression (in the regexp format) Pic 1.




**Pic 1. Configuring a custom fixup**

A fixup should be specified as DLL files for 32-bit and 64-bit environments. These modules must be created and compiled to DLL files that meet the technical requirements of PSF. You also have to provide a JSON configuration file. Refer the PSF documentation for the details about the configuration format.

*Copyright © 2001-2023 EMCO. All rights reserved.*

# Chapter 7: App-V Packaging

Microsoft Application Virtualization (App-V) for Windows allows deploying applications in real-time to any client from a virtual application server. Virtual applications are installed on a virtualization server and are delivered (streamed) to clients on demand. App-V thus allows centralized installation and management of deployed applications. It supports policy based access control; administrators can define and restrict access to the applications by certain users, or on certain computers, by defining policies governing the usage.

With a streaming-based implementation, the App-V client needs to be installed on the client machines. Users launch virtual applications from familiar access points and interact with them as if they were installed locally.

Using MSI Package Builder you can create App-V packages from scratch or repackage existing Windows applications into App-V packages. You can use installations monitoring to package existing installations into App-V packages. The repackaging process is identical to repackaging into MSI, described in the previous chapters.

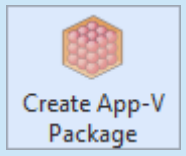
## What's Inside

Creating an App-V Package

Copyright © 2001-2023 EMCO. All rights reserved.

## Creating an App-V Package

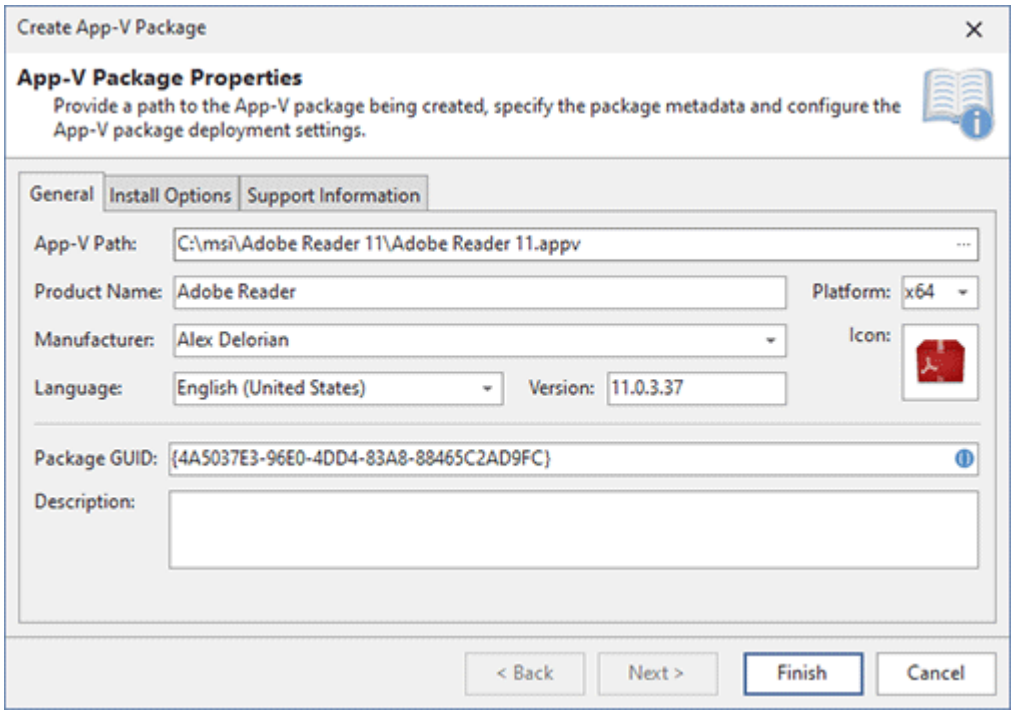
MSI Package Builder supports virtualization by repackaging existing installations into App-V packages. The App-V package can be created during an installation repackaging process or manually from an existing regular MSI Package Builder installation project.




**Create App-V Package**

The **Create App-V Package** button from the **Builder** group on the **Home** Ribbon page should be used to generate an App-V package based on the selected project.

To create an App-V package manually, you can use the **Create App-V Package** item from the **Projects** view pop-up menu or from the **Builder** group on the **Home** Ribbon page. When an App-V package is created based on a project, the options for the package created are filled automatically using the values defined in the **Project Details** view, while during the repackaging process they are defined on-line, and then saved to the created project. Let us take a look at the available options and describe each one starting with the general product information Pic 1.



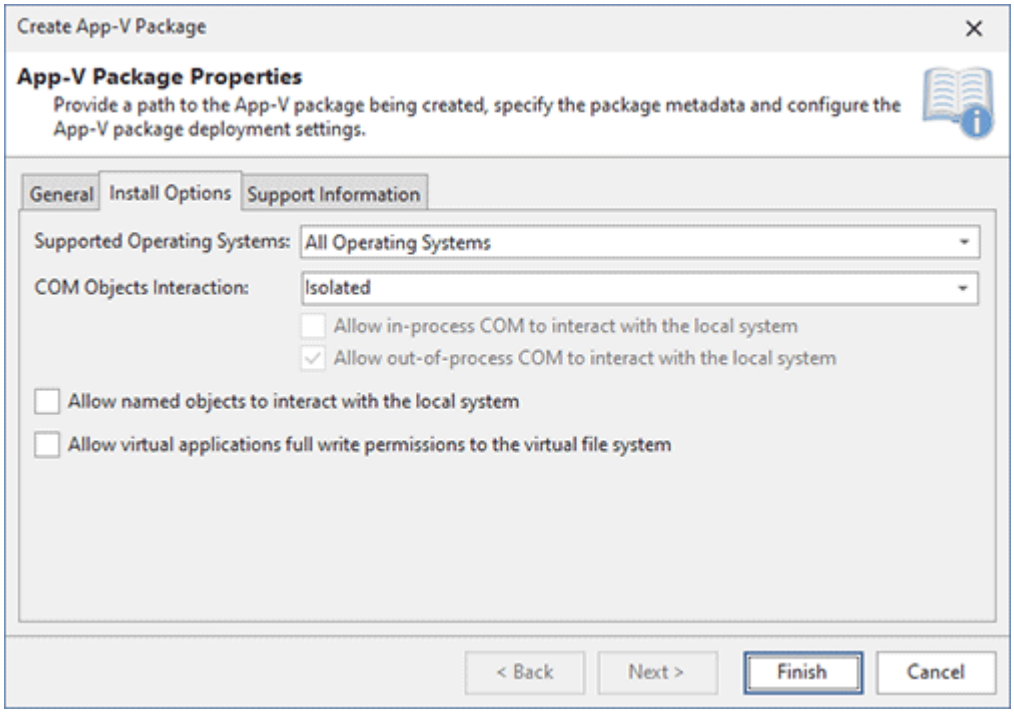

**Pic 1. Specifying general App-V package information**

Copyright © 2001-2023 EMCO. All rights reserved.

The general information consists of the **Product Name**, **Platform**, **Manufacturer**, **Icon**, **Language** and **Version**. The product name, manufacturer and icon are used to display the product on the system. The platform is used to define if the installer is deploying an x86 or an x64 application. The **Language** field is responsible for defining the language and the encoding used by the installer package.

The version is the most important field of the general information because it is responsible for a correct package installation and upgrade. That means that every time you prepare an updated package using the same project you should change its version to a higher one. And it is also displayed on the system.

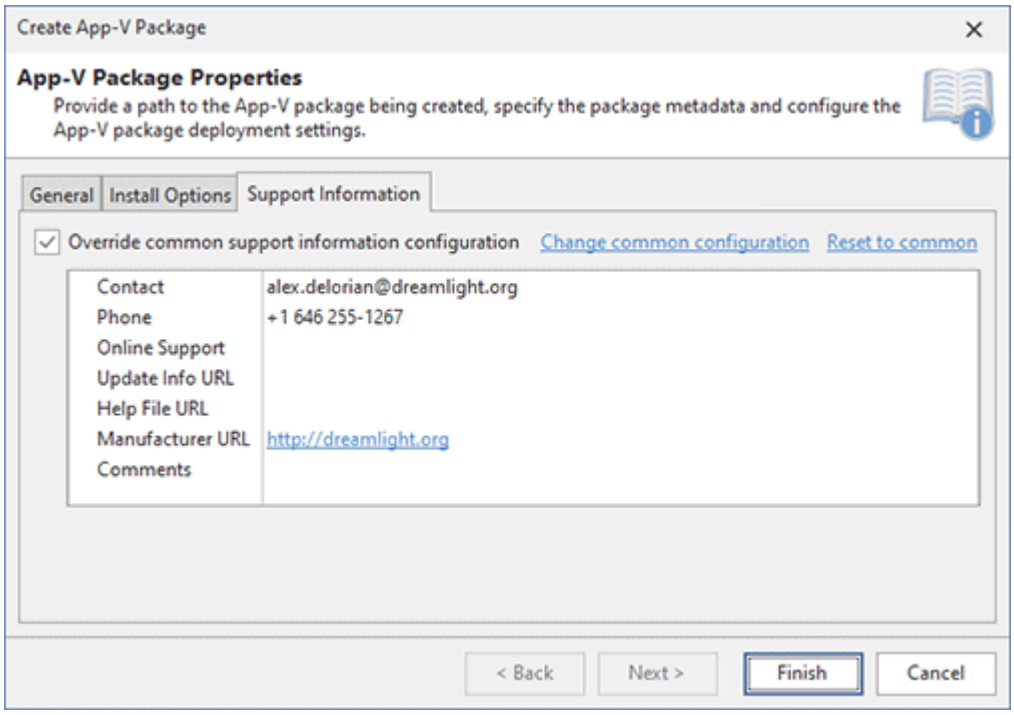
The **Package GUID** is responsible for the package identification. It is generated automatically by default and is never changed in most cases. The **Description** field is used to provide a description for the package.




**Pic 2. Configuring the Install Options**

*Copyright © 2001-2023 EMCO. All rights reserved.*

The next aspect of an App-V package configuration is the install options Pic 2. You can specify the list of supported operating systems and set up the level of interaction of the virtual application with the underlying operating system.




**Pic 3. Defining Support and Summary Information**

The last thing to define, which is also optional, is a set of support and summary properties. Those are available in the package properties and are displayed on the system. You can use the properties defined in the program preferences or override them for this package.

After all the required settings have been configured, either within the **Project Details** view or within the package creation wizard, all you need is to provide a path to the location to save the resulting App-V package to and press **Finish**. As soon as the package is created, it is ready for deployment.

*Copyright © 2001-2023 EMCO. All rights reserved.*

# Chapter 8: Using Placeholders and Filters

With MSI Package Builder, it is possible to define a set of monitoring filters that allow skipping the activity of the defined processes or the changes made to the defined file system items and registry keys during the Live Monitoring process and a set of uninstall filters that allow leaving joint resources on a PC after an MSI is uninstalled. These filters can be defined both as a simple filter value and as a regular expression. The filter condition can contain system folder definition placeholders.

The system folder definition placeholders, as well as the property definition placeholders, can also be used while defining custom actions, wrapped packages, registry changes, etc. In this chapter, we will discuss the ways of specifying filter conditions and provide a detailed description for system folder and property definition placeholders.

## What's Inside

How should I correctly specify the filter value?
System Folder Definition Placeholders
Property Definition Placeholders

Copyright © 2001-2023 EMCO. All rights reserved.

## How should I correctly specify the filter value?

With MSI Package Builder, it is possible to define a set of monitoring filters that allow skipping the activity of the defined processes or the changes made to the defined file system items and registry keys during the Live Monitoring process and a set of uninstall filters that allow leaving joint resources on a PC after an MSI is uninstalled. In this chapter, we will show you how to specify a filter value. There are two methods of specifying a filter value – those are using a simple filter value and using a regular expression. Let us take a close look at each method.

### Simple Filter Value

Simple filter value is a string to be used for matching elements to be filtered. In case of a processes filter this string should be a process name, e.g. *explorer.exe*. For the file system filters this string should be a file system path to filter – all the sub-directories and files in the path specified are also filtered, for example *C:\Windows* filter value will lead to filtering all data inside Windows directory. The registry filter string should match a path to the registry key – all the sub-keys along with registry values in the path specified are filtered, e.g. choosing *HKEY_LOCAL_MACHINE* as root key and setting *Software\Microsoft\Windows NT* as key will filter all data inside Windows registry key.



While specifying a simple filter value you can use wildcard characters same as for file system search, for example setting a file system filter value to *?:\Temp** will filter all the folders that start with *Temp* from all logical drive, such as *C:\Temp*, *D:\Temporary Data*, etc.

### Regular Expressions

Using regular expressions is a more advanced way of a filter value specification. Regular expression is a string that is used to match a set of strings, particular characters, words, or patterns of characters according to certain syntax rules that are described below. Regular expressions usage is a more flexible way of specifying a filter value than a simple filter value, because one regular expression can cover lots of items to be filtered. For example:

*^${CommonUserFolder}$\\[^\\]+\\Local Settings\\!(Application Data\\).+$*

This expression will filter all the folders from the Local Settings folder of each user except the *Application Data* folder.

The following syntax should be used to define a filter value with a help of regular expression.

| Metacharacter | Meaning |
|---|---|
| . | Matches any single character. |
| [ ] | Indicates a character class. Matches any character inside the brackets (for example, *[abc]* matches *a*, *b*, and *c*). |
| ^ | If this metacharacter occurs at the start of a character class, it negates the character class. A negated character class matches any character except those inside the brackets (for example, *[^abc]* matches all characters except *a*, *b*, and *c*).<br>If ^ is at the beginning of the regular expression, it matches the beginning of the input (for example, *^[abc]* will only match input that begins with *a*, *b*, or *c*). |
| - | In a character class, indicates a range of characters (for example, *[0-9]* matches any of the digits *0* through *9*). |

Copyright © 2001-2023 EMCO. All rights reserved.

| Metacharacter | Meaning |
|---|---|
| ? | Indicates that the preceding expression is optional: it matches once or not at all (for example, *[0-9][0-9]?* matches *2* and *12*). |
| + | Indicates that the preceding expression matches one or more times (for example, *[0-9]+* matches *1*, *13*, *456*, and so on). |
| * | Indicates that the preceding expression matches zero or more times. |
| ??, +?, *? | Non-greedy versions of ?, +, and *. These match as little as possible, unlike the greedy versions that match as much as possible (for example, given the input *<abc><def>*, *<.*?>* matches *<abc>* while *<.*>* matches *<abc><def>*). |
| ( ) | Grouping operator (for example *(\d+,)*\d+* matches a list of numbers separated by commas, such as *1* or *1*,*23*,*456*). |
| { } | Indicates a match group. |
| \ | Escape character: interpret the next character literally (for example, *[0-9]+* matches one or more digits, but *[0-9]\+* matches a digit followed by a plus character). Also used for abbreviations (such as *\a* for any alphanumeric character – see the following table for details).<br>If \ is followed by a number n, it matches the nth match group (starting from 0). Example: *<{.*?}>.*?</\0>* matches *<head>Contents</head>*. |
| $ | At the end of a regular expression, this character matches the end of the input (for example, *[0-9]$* matches a digit at the end of the input). |
| \| | Alternation operator: separates two expressions, exactly one of which matches (for example, *T\|the* matches *The* or *the*). |
| ! | Negation operator: the expression following ! does not match the input (for example, *a!b* matches *a* not followed by *b*). |

The following abbreviations can be used in regular expressions.

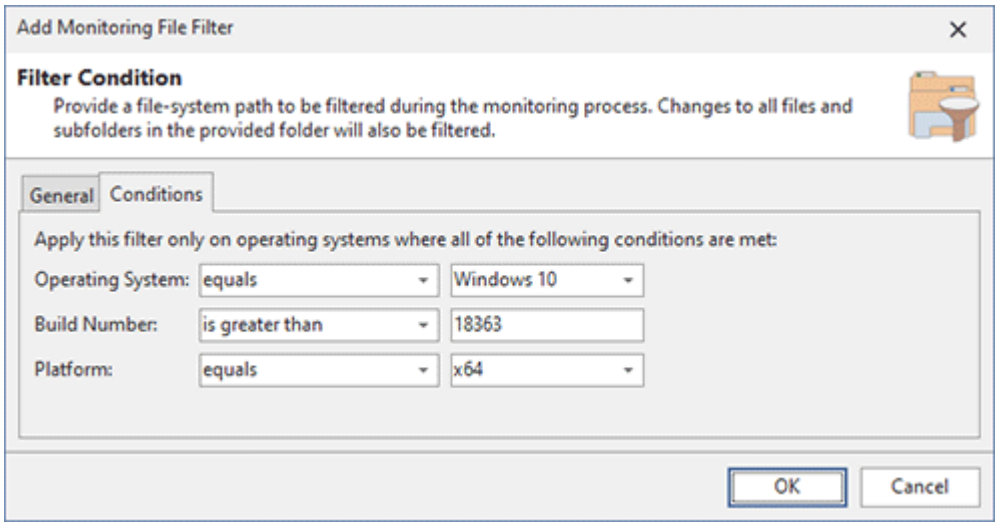
| Metacharacter | Meaning |
|---|---|
| \a | Any alphanumeric character: *([a-zA-Z0-9])* |
| \b | White space (blank): *([ \\t])* |
| \c | Any alphabetic character: *([a-zA-Z])* |
| \d | Any decimal digit: *([0-9])* |
| \h | Any hexadecimal digit: *([0-9a-fA-F])* |
| \n | Newline: *(\r\|(\r?\n))* |
| \q | A quoted string: *(\"[^\"]*\")\|(\'[^\']*\')* |
| \w | A simple word: *([a-zA-Z]+)* |
| \z | An integer: *([0-9]+)* |

Copyright © 2001-2023 EMCO. All rights reserved.

Now you are introduced to the possible ways of specifying a filter value and should be able to provide any filter to solve any problem.

## OS and Platform Conditions

For any filter type, you may configure conditions, including the operating system, the OS build number and the platform. It allows you to configure filters to be used for specific operating systems only. OS and platform conditions can be configured in the **Conditions** tab Pic 1.




**Pic 1. Configuring OS and platform conditions filters**

As a condition, you may specify the required **Operating System**, **Build Number** and **Platform**, or any combination of these fields together with the comparison operators. The applied conditions are displayed in the filters list, so you can find filters that are applicable for the current OS or any specific OS.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## System Folder Definition Placeholders

Different Windows systems may have different locations of standard Windows folders. For example, the location of the Program Files, ProgramData and other system folders may be different and depends on a particular system. You need to take that into account when you create an MSI, you cannot use absolute paths to the standard system folders. An MSI fails to deploy if it uses absolute path to system folders and these folders are missing on the system. Instead, you can use so-called system folders placeholders that represent well-known system locations and are automatically converted to correct file system paths when a package is deployed.

When you use **Repackage Installation Wizard** to monitor an installation, the program automatically converts captured absolute paths to system folders. In the **File System** editor you can select a system folder to add a file into it when adding a file manually. You can automatically covert absolute paths to system folders by executing the **Roll All System Folders** action from the context menu of the project.

System folders placeholders are represented in the project with a set of predefined names to well-known system paths. In the project they are represented with special names started with **${** and ended with **}$**, for example **${ProgramFilesFolder}$**.

System folders are replaced with the file system locations actual for the system where the package is deployed, so file system resources are created in correct locations.

System folder placeholders can be used in different places within the project and in the program settings. In the **File System** section of the project system folders are used to configure paths to the files and folders to be deployed. System folders can be used in **Custom Actions** to specify a path to the deployed script or executable to be invoked after deployment. System folder definition placeholders can be used to configure filters in the program Preferences.

Below is a list of the available placeholders for system folder definitions.

| Placeholder | Replacement |
|---|---|
| AdminToolsFolder | The file system directory that is used to store administrative tools for the current user. |
| AppDataFolder | The **Application Data** folder for the current user. |
| CommonAdminToolsFolder | The file system directory that contains administrative tools for all users of the computer. |
| CommonAppDataFolder | The **Application Data** folder that is common for all users. |
| CommonDesktopFolder | The folder for the **Desktop** content that is common for all users. |
| CommonDocumentsFolder | The file system directory that contains documents that are common to all users. |
| CommonFiles64Folder | The **Common Files** folder for 64-bit applications. |
| CommonFilesFolder | The **Common Files** folder for 32-bit applications. |
| CommonProgramMenuFolder | The folder for the **All Programs** menu content that is common for all users. |
| CommonStartMenuFolder | The folder for the **Start menu** content that is common for all users. |

Copyright © 2001-2023 EMCO. All rights reserved.

| Placeholder | Replacement |
|---|---|
| CommonStartupFolder | The folder for the programs executed on startup for all users. |
| CurrentUserProfileFolder | The current user's profile folder. |
| DesktopFolder | The current user's **Desktop** content folder. |
| FavoritesFolder | The **Favorites** folder for the current user. |
| FontsFolder | The folder used to install fonts to. |
| LocalAppDataFolder | The folder that serves as a data repository for local (nonroaming) applications for the current user. |
| MyPicturesFolder | The **Pictures** folder for the current user. |
| NetHoodFolder | The file system directory that contains the link objects that may exist in the **My Network Places** virtual folder. |
| PersonalFolder | The **Documents** folder for the current user. |
| PrintHoodFolder | The file system directory that contains the link objects that can exist in the **Printers** virtual folder. |
| ProgramFiles64Folder | The **Program Files** folder for 64-bit applications. |
| ProgramFilesFolder | The **Program Files** folder for 32-bit applications. |
| ProgramMenuFolder | The folder for the **All Programs** menu content for the current user. |
| RecentFolder | The file system directory that contains shortcuts to the user's most recently used documents. |
| SendToFolder | The folder tor the **Send To** menu content for the current user. |
| StartMenuFolder | The folder for the **Start** menu content for the current user. |
| StartupFolder | The folder for the programs executed on startup for the current users. |
| System16Folder | The **System folder** for 16-bit libraries. |
| System64Folder | The **System folder** for 64-bit libraries. |
| SystemFolder | The **System folder** for 32-bit libraries. |
| TempFolder | The file system directory that is designated for temporary files. |
| TemplateFolder | The **Templates** folder for the current user. |
| UserProfilesFolder | The system user profiles folder. |
| WindowsFolder | The folder the operating system is installed to. |
| WindowsVolume | The volume the operating system is installed on. |

Please note that specific system folders can be indefinite on specific operating systems, so you should make sure that the operating systems you have defined as target during a deployment package creation process support the folders you have defined.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Property Definition Placeholders

MSI Package Builder allows using standard and user-defined MSI property placeholders when configuring custom actions, registry modifications and other changes in the installation project. Standard property placeholders are defined in Windows Installer specification. You can find a full list of supported standard properties in the **Property Reference** article on MSDN. These properties include system folder placeholders, explained in the system folder definition placeholders article.

Placeholders are replaced with actual values during the package deployment. In the project you can use a property definition placeholder that starts with **${** and ends with **}$** and contains one of the supported property names, e. g. **${Manufacturer}$**. For example, if you specify this placeholder as a value of the registry key, it will be replaced with the name of package manufacturer during package deployment. As another example, you can use a system folder placeholder to specify a path to a script that should be executed in the custom action configuration. The system folder placeholder will be replaced with the actual path during the package deployment.

The program allows using user-defined MSI properties passed to the package externally during deployment. These properties should be spelled with all capital letters and should be passed to the package during deployment as parameters of msiexec.exe command, for example:

*msiexec.exe /i installer.msi AUTOUPDATE=1;REGISTEREDOWNER=Dreamlight;REINSTALLMODE=omus*

The command allow defines **${AUTOUPDATE}$**, **${REGISTEREDOWNER}$** and **${REINSTALLMODE}$** property placeholders that can be used in the project.

In addition to standard and user-defined placeholders, there are additional placeholders available that can be used to configure specific project settings.

## Custom Action Placeholders

Using the following placeholders you can specify locations of file system resources. For example, a custom action includes a reference to a script deployed and executed by the action. To specify a path to the script a special placeholder should be used. It has the same syntax as system folder placeholders starting from **${** and ending by **}$**. Below is a list of placeholders supported by custom actions.

| Placeholder | Replacement |
|---|---|
| ActionFiles | This placeholder can be used in **Custom Actions** to specify a path to the location that stores files deployed by the custom action. To use this placeholder you need to add a file in the custom action. The placeholder can be used to specify a path to the files configured in the custom action. |

## Wrapped Package Placeholders

When you configure a wrapped package, you can use special placeholder to specify paths to the file system locations. Below you can find a list of supported placeholders. They have the same syntax as system folder placeholders starting from **${** and ending by **}$**.

| Placeholder | Replacement |
|---|---|
| PackageSetupFile | This placeholder can be used to specify a full path to the wrapped installation. You can use it in the wrapped package installation parameters configuration. |

Copyright © 2001-2023 EMCO. All rights reserved.

| Placeholder | Replacement |
| --- | --- |
| PackageSetupDir | This placeholder can be used to specify a full path to the directory where the wrapped installation is located. You can use it in the wrapped package installation parameters configuration. |

Copyright © 2001-2023 EMCO. All rights reserved.

# Chapter 9: Command-Line Interface

The MSI Package Builder command-line interface provides a set of commands that allow you to monitor installations, build packages, import/export projects and perform other operations. These commands can be used as alternatives for the features available in the program GUI, so you can run a command from the command-line for the main features instead of performing it in the GUI. However, the command-line interface supports the basic features only, and you need to use the GUI to work with the advanced features.

The command-line interface is provided by the *PackageBuilderCMD.exe* executable available in the program installation folder. You can run a command using the following format:

*PackageBuilderCMD.exe <command>*

| Command | Description |
|---|---|
| -monitor | Monitor system activity and create a project based on the captured changes. |
| -build-msi | Build an MSI package using an existing project. |
| -build-appv | Build an App-V package using an existing project. |
| -build-appx | Build an MSIX/AppX package using an existing project. |
| -sign-package | Sign the package with a digital certificate. |
| -import-project | Import the project from an archive file. |
| -export-project | Export the project to an archive file. |
| -help <command> | Display detailed help for the specified command. |

The command-line interface requires that you should have a license for the program. It is possible to use the command-line interface in the trial mode, but the generated MSI packages will have the same restrictions as those applied to packages generated in the GUI.



You need to have local administrative permissions to run MSI Package Builder commands, so it is recommended to open the command-line prompt in Windows "As Administrator".

If you would like to get help for a specific command, you can run *PackageBuilderCMD.exe* with the *-help* command, for example:

*PackageBuilderCMD.exe -help monitor*

Detailed information on these commands and examples of their usage are available in the next chapters.

## What's Inside

Monitoring Command
Package Building and Signing Commands
Project Import and Export Commands

Copyright © 2001-2023 EMCO. All rights reserved.

## Monitoring Command

The monitoring command allows repackaging an installation and creating a project based on the captured changes. As a result of the execution of this command, you get a project with captured changes, which you can use to generate a package in different formats. When you run the monitoring command, the repackaging process is performed on the local machine, so you need to make sure your repackaging environment is clean and you follow the repackaging best practices.

To run monitoring, you need to execute the following command:

*PackageBuilderCMD.exe -monitor <project-name> [-i <application-path> [-p <application-params>] | -m <applications-list>] [-l <path>] [-e <project-archive>] [-c]*

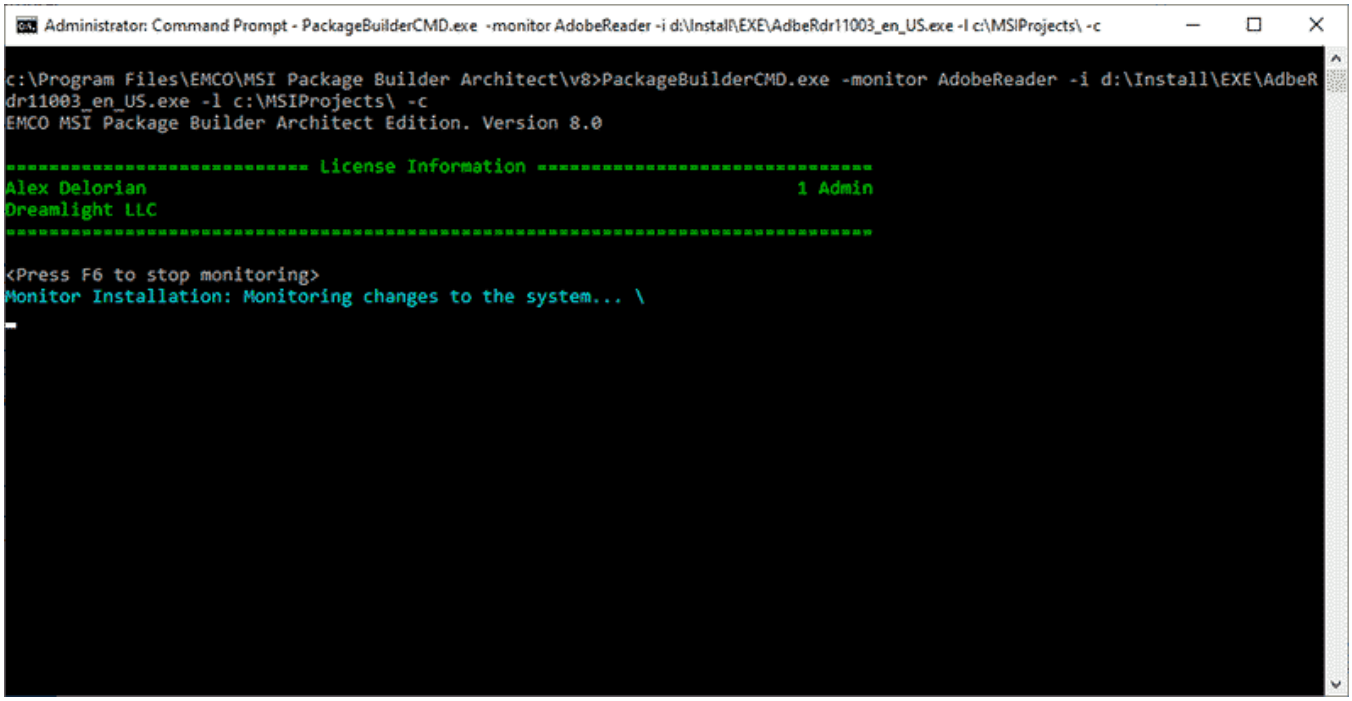
| Parameter | Optional | Description |
|---|---|---|
| <project-name> | No | The name of the project to be created. |
| -i <application-path> | Yes | The path to the installation to be monitored (*.exe, *.bat, *.cmd, *.msi, *.msp file types are supported). |
| -p <application-params> | Yes | Command-line parameters of the installation. This parameter can be used if the installation path is specified with the *–i* option. |
| -m <applications-list> | Yes | The path to the .csv file with a list of installations to be captured and their command-line parameters. |
| -l <path> | Yes | The path to the project to be created. If this option is skipped, the project is created in the current folder. |
| -e <project-archive> | Yes | The path to the archive file to export the project to. |
| -c | Yes | Prepare the project after monitoring. |

The monitoring command can be used for capturing installations and system changes. For example, if you need to repackage an executable installation, you can run the following command:

*PackageBuilderCMD.exe -monitor AdobeReader –i c:\downloads\reader-install.exe –c*

When you monitor an installation, the specified installation file is executed automatically, and you need to follow the installation steps as usual. The monitoring progress and additional information are displayed in the console window Pic 1. Once the installation is completed, the monitoring stops automatically and the project is created.

Copyright © 2001-2023 EMCO. All rights reserved.




**Pic 1. Monitoring console**

If you need to repackage multiple installations into a single package, you can run the following command using the path to a .csv file where the installations and their command-line parameters are specified:

*PackageBuilderCMD.exe –monitor OfficeTools –m c:\config\office-tools.csv –c*

The program starts monitoring and runs the installations specified in the .csv file. When the last installation is completed, the monitoring stops automatically and the project is created. You can use the created project to generate an output package by running other commands.

Using the monitoring command, you can also capture system changes to generate a project including those changes. In this case, you don't have an installation to be repackaged, so you need to start monitoring and apply the required changes manually. Run the following command that enables that:

*PackageBuilderCMD.exe –monitor FontsInstall –c*

When monitoring is enabled, you need to apply the required changes to be captured. To stop monitoring and create a project, you need to press **F6** in the monitoring console window.

## Final Notes

In the examples above, the *–c* parameter is used. It allows preparing a project after monitoring by saving all captured files in the project folder, so that the project can be transferred to another computer and used for generating a package. It's recommended to use the *–c* if there are no reasons to skip it.

If you don't use the *–l* parameter to specify a path to the project location, the program creates a folder with the project name locally, i.e. in the folder where *PackageBuilderCMD.exe* is executed. Use the *–l* parameter to specify a path to the project location if you need to save the project to a specific directory.

You can save the project content as a single archive file using the *–e* option.

Copyright © 2001-2023 EMCO. All rights reserved.

## Package Building and Signing Commands

If you have a project, you can generate a package using the project content. It is possible to generate a package in the MSI, App-V and AppX/MSIX formats. To generate an MSI package, you can run the following command:

*PackageBuilderCMD.exe -build-msi <project-path> <package-path> [-g <guid>] [-u <guid>] [-m <manufacturer>] [-n <product-name>] [-l <language-id>] [-v <version>] [-p <x86|x64>]*

| Parameter | Optional | Description |
|---|---|---|
| <project-path> | No | The path to the project used to generate an MSI package. |
| <package-path> | No | The path to the MSI package output file. |
| -g <guid> | Yes | Set the product GUID for the generated MSI package. |
| -u <guid> | Yes | Enable an upgrade for the generated MSI package with the specified GUID. |
| -m <manufacturer> | Yes | Set the manufacturer for the generated MSI package. |
| -n <product-name> | Yes | Set the product name for the generated MSI package. |
| -l <language-id> | Yes | Set the MSI package language. |
| -v <version> | Yes | Set the MSI package version. |
| -p <x86\|x64> | Yes | Set the MSI package platform. |

To generate an MSI, you need to specify a path to the project and the output file only. Other parameters are optional. For example, you can use the following command to generate an MSI package:

*PackageBuilderCMD.exe -build-msi c:\Projects\AdobeReader c:\Projects\AdobeReader.msi*

The project name, manufacturer, version and other settings should be specified in the project, so that you don't have to specify them when you generate a package from the command-line. If those settings are missing for some reason, you can specify them using the corresponding command-line parameters.

To generate an App-V package, you can run the following command:

*PackageBuilderCMD.exe -build-appv <project-path> <package-path> [-g <guid>] [-m <manufacturer>] [-n <product-name>] [-l <language-id>] [-v <version>] [-p <x86|x64>]*

| Parameter | Optional | Description |
|---|---|---|
| <project-path> | No | The path to the project used to generate an App-V package. |
| <package-path> | No | The path to the App-V package output file. |
| -g <guid> | Yes | Set the product GUID for the generated App-V package. |
| -u <guid> | Yes | Enable an upgrade for the generated App-V package with the specified GUID. |
| -m <manufacturer> | Yes | Set the manufacturer for the generated App-V package. |

Copyright © 2001-2023 EMCO. All rights reserved.

| Parameter | Optional | Description |
|---|---|---|
| -n <product-name> | Yes | Set the product name for the generated App-V package. |
| -l <language-id> | Yes | Set the App-V package language. |
| -v <version> | Yes | Set the App-V package version. |
| -p <x86\|x64> | Yes | Set the App-V package platform. |

Example:

*PackageBuilderCMD.exe -build-appv c:\Projects\AdobeReader c:\Projects\AdobeReader.appv*

To generate an MSIX/AppX package, you can run the following command:

*PackageBuilderCMD.exe -build-appx <project-path> <package-path> [-a <package-name>] [-u <publisher>] [-d <package-description>] [-m <manufacturer>] [-n <product-name>] [-l <language-id>] [-v <version>] [-p <x86|x64>]*

| Parameter | Optional | Description |
|---|---|---|
| <project-path> | No | The path to the project used to generate an MSIX/AppX package. |
| <package-path> | No | The path to the MSIX/AppX package output file. |
| -a <package-name> | Yes | The package name. |
| -u <publisher> | Yes | The name of the package publisher (in the Relative Distinguished Name (RDN) format) |
| -d <package-description> | Yes | The package description. |
| -m <manufacturer> | Yes | Set the manufacturer for the generated MSIX/AppX package. |
| -n <product-name> | Yes | Set the product name for the generated MSIX/AppX package. |
| -l <language-id> | Yes | Set the MSIX/AppX package language. |
| -v <version> | Yes | Set the MSIX/AppX package version. |
| -p <x86\|x64> | Yes | Set the MSIX/AppX package platform. |

Example:

*PackageBuilderCMD.exe -build-appx c:\Projects\AdobeReader c:\Projects\AdobeReader.appx*

You can digitally sign a package using the sign command. You can find detailed information on digital signatures in the signing packages chapter. To digitally sign a package, run the following command:

*PackageBuilderCMD.exe -sign-package <package-path> <certificate> [-p <password>] [-t <timestamp-url>]*

| Parameter | Optional | Description |
|---|---|---|
| <package-path> | No | The path to the AppX or MSI package to sign. |
| <certificate> | No | The path to the certificate pfx file to sign the package with. |
| -p <password> | Yes | The certificate password. |

Copyright © 2001-2023 EMCO. All rights reserved.

| Parameter | Optional | Description |
|---|---|---|
| -t <timestamp-url> | Yes | The timestamp URL to use in the signing process. |

Example:

*PackageBuilderCMD.exe -sign-package c:\Projects\AdobeReader.msi c:\certificate.pfx*

## Project Import and Export Commands

MSI Package Builder has commands that allow you to export and import projects to/from the project archive file. Using the project archive file, you can transfer a project to another computer or back up a project as a single file.

You can use the following command to export a project:

*PackageBuilderCMD.exe -export-project <project-path> <project-archive>*

| Parameter | Optional | Description |
|---|---|---|
| <project-path> | No | The path to the project to export. |
| <project-archive> | No | The path to the destination project archive file *.pbproj. |

Example:

*PackageBuilderCMD.exe -export-project c:\Projects\AdobeReader c:\Projects\AdobeReader.pbproj*

To import a project from the project archive file, you can run the following command:

*PackageBuilderCMD.exe -import-project <project-archive> <project-location> [-n <project-name>]*

| Parameter | Optional | Description |
|---|---|---|
| <project-archive> | No | The path to the project archive file *.pbproj to import. |
| <project-location> | No | The path to the folder to import the project content to. |
| -n <project-name> | Yes | The name of the project to be created to store the imported data. |

Example:

*PackageBuilderCMD.exe -import-project c:\Projects\AdobeReader.pbproj c:\Projects\AdobeReader*

Copyright © 2001-2023 EMCO. All rights reserved.

# Chapter 10: Log

The log is designed to store information on the events taking place while the program is running. Most of them are messages sent by the operations and operations results. Such events are displayed in the **Log** view and can significantly help you to analyze the operation execution results and troubleshoot problems taking place while the program is in use. The log may grow continually, thus slowing down the program loading and response time. To prevent this, it can be cleared manually at any moment.

In this chapter, we will explain how to analyze the log to ensure that the operations are completed successfully or to troubleshoot possible problems and describe the option of exporting the log to a simple format.

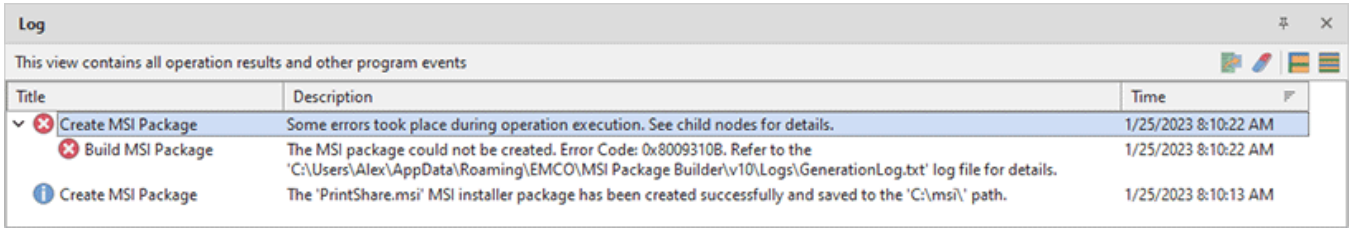
## What's Inside

Analyzing Log
Exporting Log

## Analyzing Log

The main purpose of the **Log** view is to help you understand if the operation execution has succeeded and troubleshoot problems if any have occurred. Each entry in the log has a severity icon, a title and a description. From the title, you can understand which operation has been performed; the description provides you with the result message and a hint on solving the problem, if any; and the severity icon can be used to quickly understand if the operation has fully succeeded.

For example, let us take a closer look at the following result set in the log Pic 1.




**Pic 1. Sample logged events**

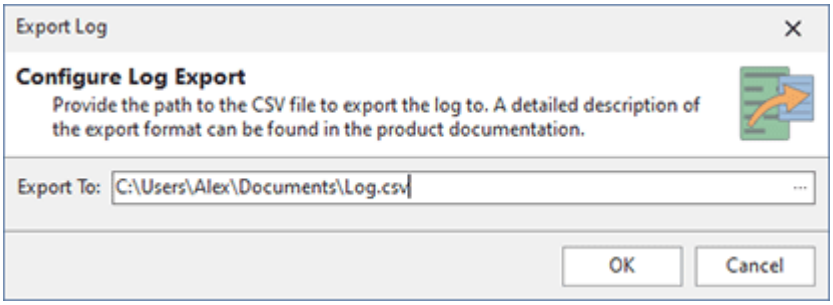
The picture above shows the set of result we received after performing an operation. As we can see, most of the logged events are informational, but some stand for problems. We need to find out what caused the problem and what should be done to avoid it in future. In addition, it may be interesting to go through the warnings to see if anything wrong is going on.

After the results have been reviewed and all the problems have been solved, you can run the operation again and ensure that it completes successfully.

Copyright © 2001-2023 EMCO. All rights reserved.

## Exporting Log

With MSI Package Builder, you can easily export the log to the CSV file format for future analysis or processing by an automated tool. To export the log, click the **Export** button from the **Log** view toolbar or choose the **Export** item from the pop-up menu. The **Export Log** dialog will appear on the screen Pic 1.




**Pic 1. Exporting Log**

In the **Export Log** dialog, you are offered to choose a file you are going to save the log data to. The file path should be provided to the **Export To** field. After you are ready with configuring the export, press **OK** to proceed. The file containing the log data will be created in the path specified.

The CSV file with the exported data consists of four columns, which are the following:

| Column Index | Header | Description |
|---|---|---|
| 1 | Title | The logged event title (including the path). |
| 2 | Severity | The logged event severity level. |
| 3 | Description | The logged event description. |
| 4 | Time | The time when the event occurred. |

### Sample exported log data in the CSV format

"Title","Severity","Description","Time"

"Create MSI Package","Information","The operation was completed successfully.","12/3/2013 11:14:50 AM"

"Create MSI Package","Cancel","The operation was canceled by the user.","12/3/2013 11:12:51 AM"

"Create MSI Package\Code Page Verification","Cancel","All code page errors should be resolved for a successful MSI package creation. To find out which items could not pass the code page verification process, review the Application Log.","12/3/2013 11:12:51 AM"

"Create MSI Package\Code Page Verification","Warning","The characters contained in the 'HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Adobe\PDF Owner\PreviousOwner\Acroexch.Document.7\shell\Open' Registry item are not available in the code page selected for the MSI package.","12/3/2013 11:12:43 AM"

"Create MSI Package\Code Page Verification","Warning","The characters contained in the 'HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Adobe\PDF Owner\PreviousOwner\CLSID\{B801CA65-A1FC-11D0-85AD-444553540000}\Verb\0' Registry item are not available in the code page selected for the MSI package.","12/3/2013 11:12:43 AM"

Copyright © 2001-2023 EMCO. All rights reserved.

"Create MSI Package\Code Page Verification","Warning","The characters contained in the 'HKEY_LOCAL_MACHINE\SOFTWARE\Classes\AcroExch.Document.7\shell\Open' Registry item are not available in the code page selected for the MSI package.","12/3/2013 11:12:43 AM"

Now you are introduced to the log export and export data file format and can use the export feature for the log analysis without any misunderstanding.

Copyright © 2001-2023 EMCO. All rights reserved.

# Chapter 11: Program Preferences

MSI Package Builder comes with a wide range of settings available for changing by any user. Every preference page has a detailed description of its content and of the feature it is used to configure. To reach the program preferences, click the **Preferences** button available from the **Application Menu**. Besides, the clickable Ribbon groups' glyphs open the preference pages that configure the functionality incorporated in the respective group.

### What's Inside

MSI Package Builder Part
Filters Part
Miscellaneous Part

## MSI Package Builder Part

The **MSI Package Builder** part of the program preferences is used to configure the main program settings, such as the projects storage location, project signing options and user interface settings. To open the **Preferences** dialog, click the **Preferences** button available from the **Application Menu**. Feel free to configure the available settings to suit your needs best.

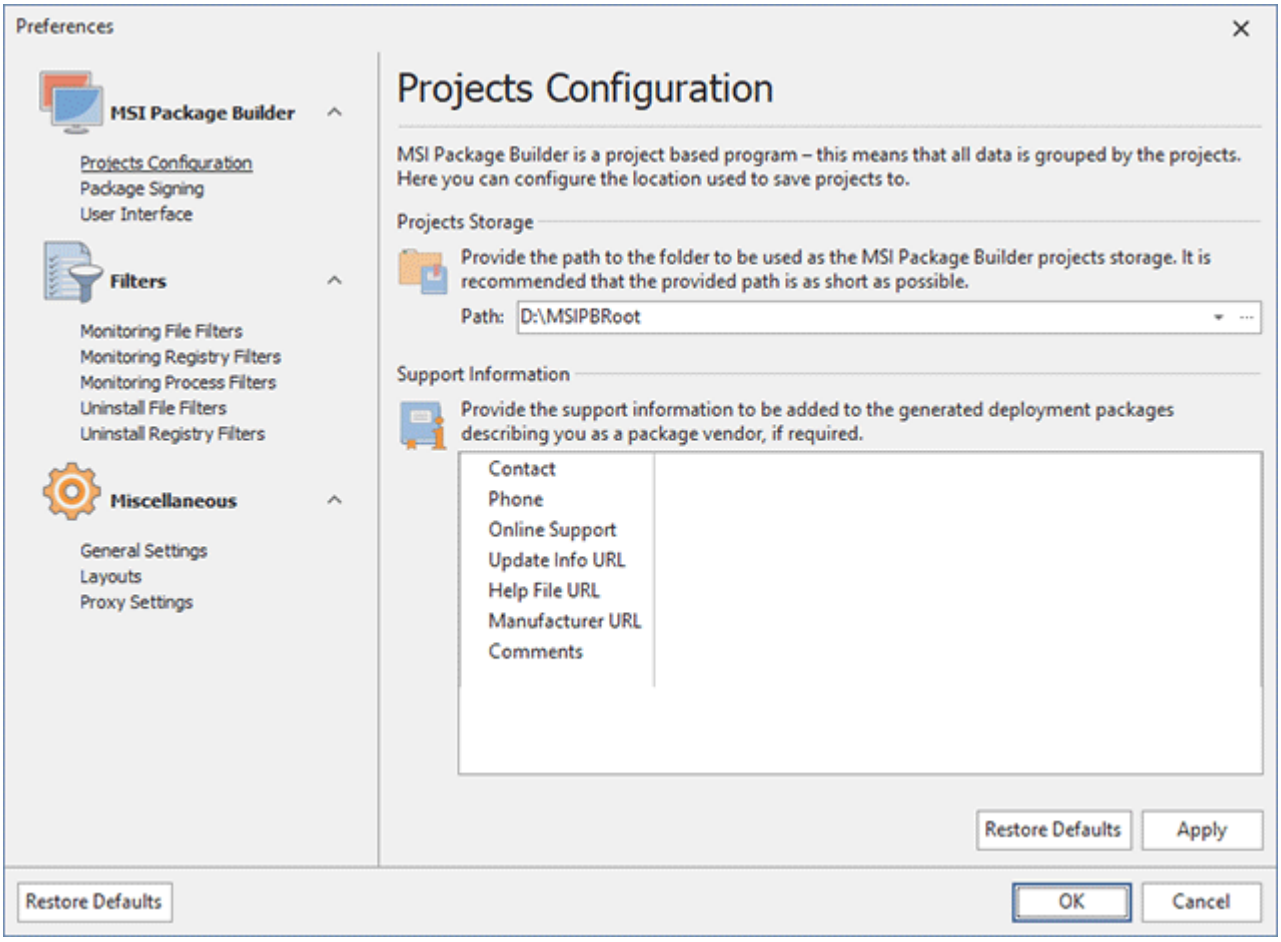
### What's Inside

Projects Configuration Page
Package Signing Page
User Interface Page

Copyright © 2001-2023 EMCO. All rights reserved.

## Projects Configuration Page

MSI Package Builder is a project based program – this means that all data is grouped by the projects. Each project contains one or several packages. Such data structure allows grouping the coupled installations, storing them and managing them as the unit. The projects are saved to the projects storage. To define a projects storage location, open the program preferences using the **Preferences** button from the **Application Menu** and click the **Projects Configuration** link on the navigation bar on the left of the **Preferences** dialog within the **MSI Package Builder** group Pic 1.




**Pic 1. The Projects Configuration preference page**

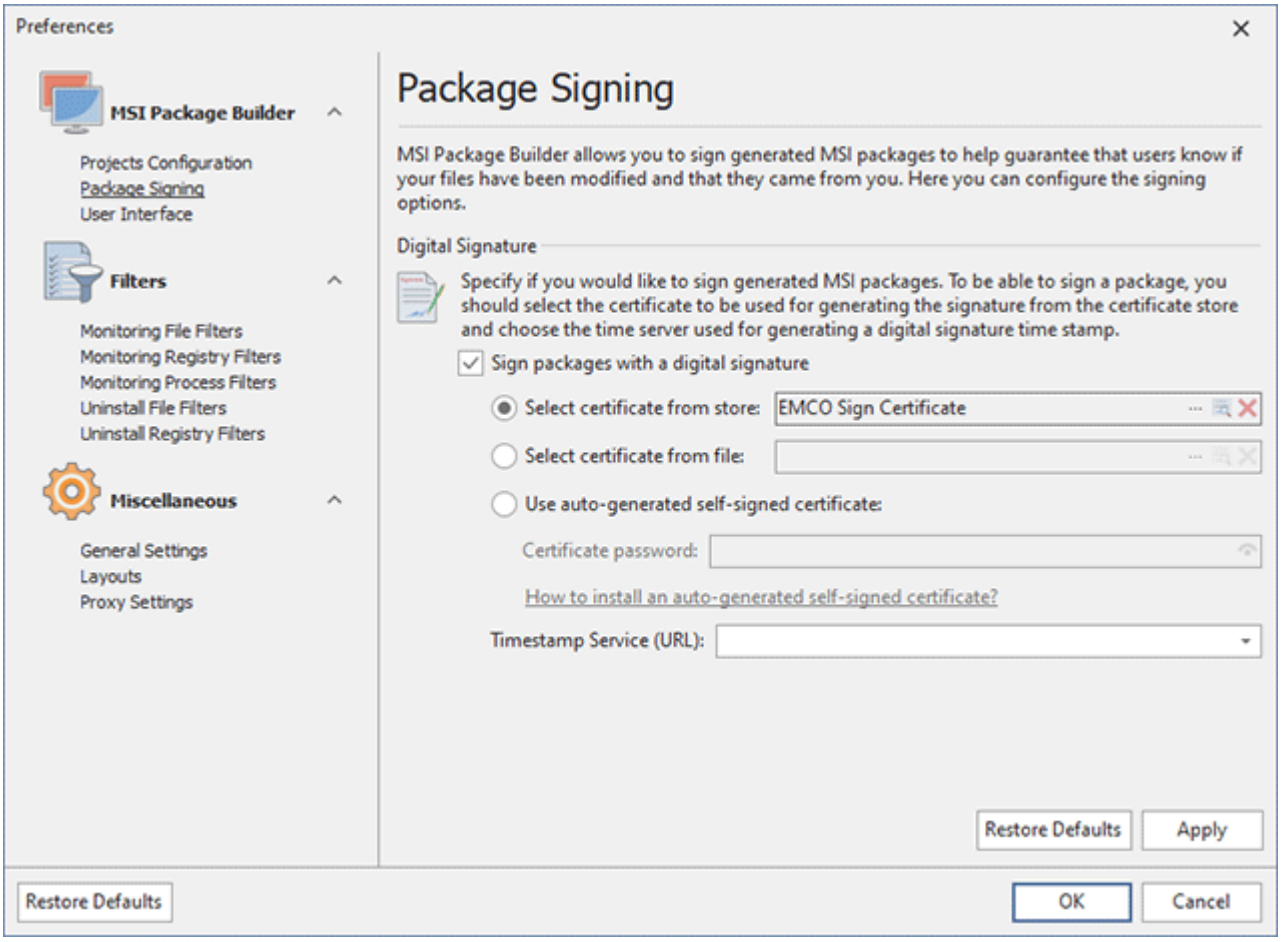
The path to the folder to organize the MSI Package Builder projects storage in should be provided to the **Path** field within the **Projects Root Directory** group.

On the **Projects Configuration** preference page, you can provide the default support information to be used for all deployment packages, if not overridden for a specific project.

Copyright © 2001-2023 EMCO. All rights reserved.

## Package Signing Page

MSI Package Builder allows you to sign generated deployment packages to help guarantee that users know if your packages have been modified and that they came from you, the publisher. To configure the signing options, open the program preferences using the **Preferences** button from the **Application Menu** and click the **Package Signing** link on the navigation bar on the left of the **Preferences** dialog within the **MSI Package Builder** group Pic 1.




**Pic 1. Configuring the package signing options**

Within the **Digital Signature** group, you can define if generated deployment packages should be signed, select the certificate to be used for a digital signature (either from those available in the certificate storage, or from the file, or use auto-generated one) and choose the time server that should be used for creating a time stamp for the digital signature.

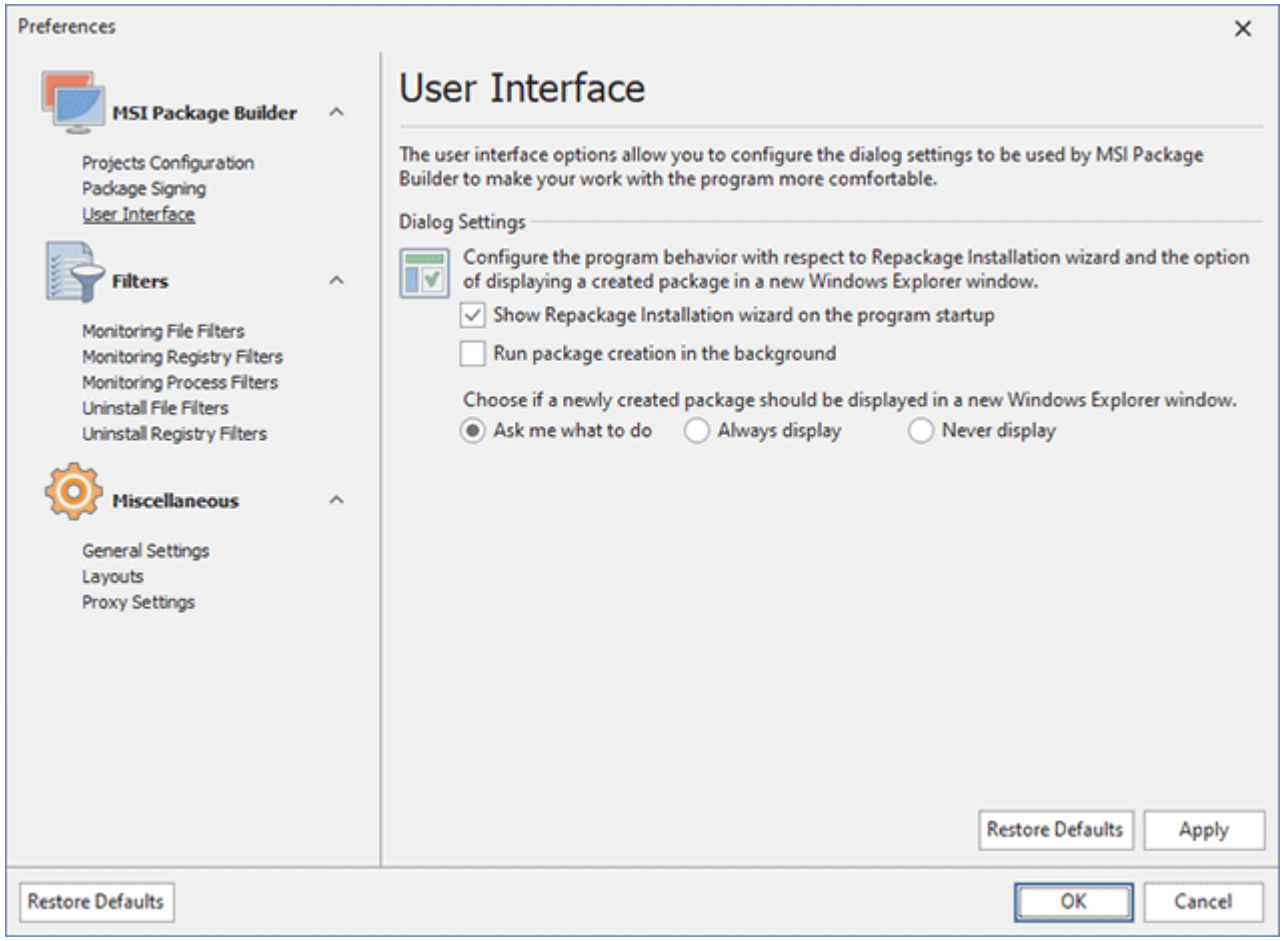
The package signing options defined in the program preferences are used as default for each project, but it is possible to override those options for each specific project on the **Project Details** view or while creating an MSI package or creating an MSIX/AppX package from the project.

Refer to the Signing Packages section of this document for detailed information on the purpose and the process of packages signing.

Copyright © 2001-2023 EMCO. All rights reserved.

## User Interface Page

MSI Package Builder is a multifunctional tool, which can be used as an easy-to-use installations repackager and as an advanced packages editor. That is why different user interface configuration may be required for different persons. The configuration is performed on the **User Interface** preference page Pic 1. To access this page, click the **Preferences** button from the **Application Menu** and select the appropriate link in the navigation bar, on the left in the **Preferences** dialog within the **MSI Package Builder** group.




**Pic 1. Configuring user interface**

For those who use it as a simple repackager, it might be enough to walk through the repackaging wizard and build a resulting deployment package, so it would be convenient for them to display the **Repackage Installation** wizard on start up. However, it is not convenient for those who are using the program also for managing the software packages or creating their own packages. It is also possible to define if the deployment package creation is always run in background.

The other aspect is a decision making. By default, after each deployment package creation the program asks if the folder containing the newly created package should be opened in Windows Explorer. If your answer is always the same, you can answer this question only once, on the **User Interface** page.

Copyright © 2001-2023 EMCO. All rights reserved.

## Filters Part

During the monitoring process MSI Package Builder records the modifications performed by all processes that operate in the system. Thus monitoring results can include the modifications performed by system processes. These modifications do not refer to the application or installation that you have monitored and that is why they should be excluded from monitoring results. This goal can be reached using the Monitoring Processes Filters configuration. While using process filters, the changes made by those processes are automatically excluded from the resulting deployment package. In the same manner, there are registry keys and file system paths that should be excluded from the resulting deployment package. The Monitoring Registry Filters and Monitoring File Filters configuration allow you to fulfill this.

When the installation creates a joint resource that is commonly used by several applications, such resource should be indicated as permanent. Permanent files, keys and registry entries are not deleted during application uninstall process to prevent functionality violation of applications that use these resources. You are able to set permanency option for each resource in separate during its edit but it seems to be quite a complex way. To simplify the operation with permanent resources MSI Package Builder comes with a set of predefined uninstall filters and allows you to add some.

All the filters in MSI Package Builder are set up in the program preferences. Press the **Preferences** button from the **Application Menu** and you'll find the filtration settings under the **Filters** group.

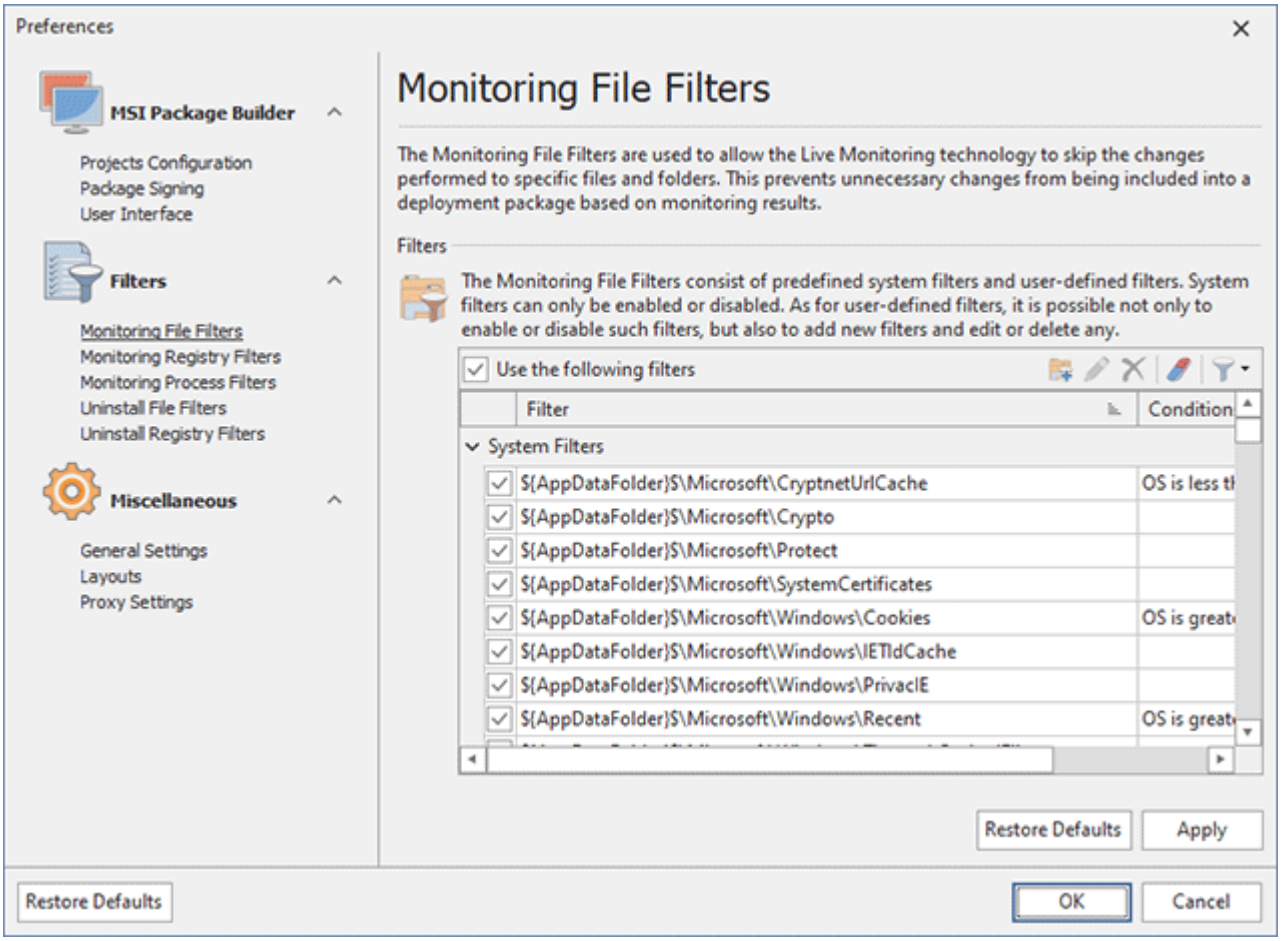
### What's Inside

Monitoring File Filters Page
Monitoring Registry Filters Page
Monitoring Processes Filters Page
Uninstall File Filters Page
Uninstall Registry Filters Page

Copyright © 2001-2023 EMCO. All rights reserved.

## Monitoring File Filters Page

The Monitoring File Filters are used to allow the Live Monitoring technology to skip the changes performed to specific files and folders. This prevents unnecessary changes from being included into a deployment package based on monitoring results. To configure the Monitoring File Filters open the program preferences using the **Preferences** button from the **Application Menu** and click the **Monitoring File Filters** link on the navigation bar on the left of the **Preferences** dialog within the **Filters** group Pic 1.




**Pic 1. Configuring Monitoring File Filters**

The filters are divided into two categories; those are **User Filters** and **System Filters**. The system filters are predefined ones and cannot be deleted, but can be disabled, if required, using the **Disable Selected** item from the pop-up menu. You can then re-enable any filter using the **Enable Selected** menu item. To enable/disable a filter and to check if it is enabled, you can use the check boxes located on the left of each row representing the filter. It is also possible to refuse filtering file system items by disabling the **Use the following filters** option.

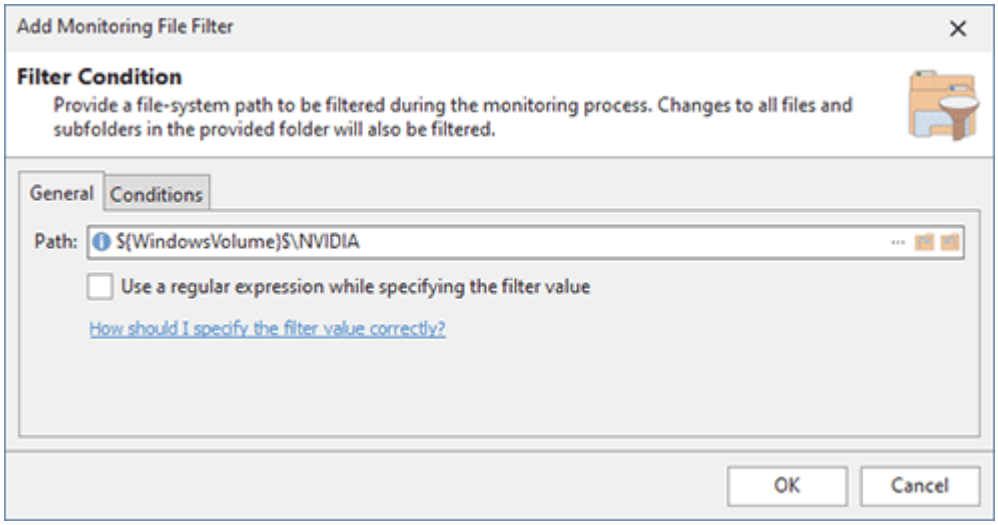
### Toolbar Overview

| | |
|---|---|
|  | **Add Filter**<br>The **Add Filter** button should be used to add a new user-defined **Monitoring File Filter**. |
|  | **Edit**<br>The **Edit** button allows you to change the selected user-defined **Monitoring File Filter**. |

Copyright © 2001-2023 EMCO. All rights reserved.

| | |
|---|---|
|  | **Delete**<br>The **Delete** button allows you to delete the selected user-defined **Monitoring File Filter** items. |
|  | **Clear User Filters**<br>The **Clear User Filters** button should be used to delete all user-defined **Monitoring File Filter** items. |
|  | **View Mode**<br>The **View Mode** button is intended to show only the filters that meet specific criteria. You may show all the filters, the filters applicable for the current OS only, or specify a custom criterion for filters. |

As for the user filters, those are the user-defined ones. To add a new filter, use the **Add Filter** item from the pop-up menu or press the **Add Filter** button on the toolbar.




**Pic 2. Configuring a Monitoring File Filter**

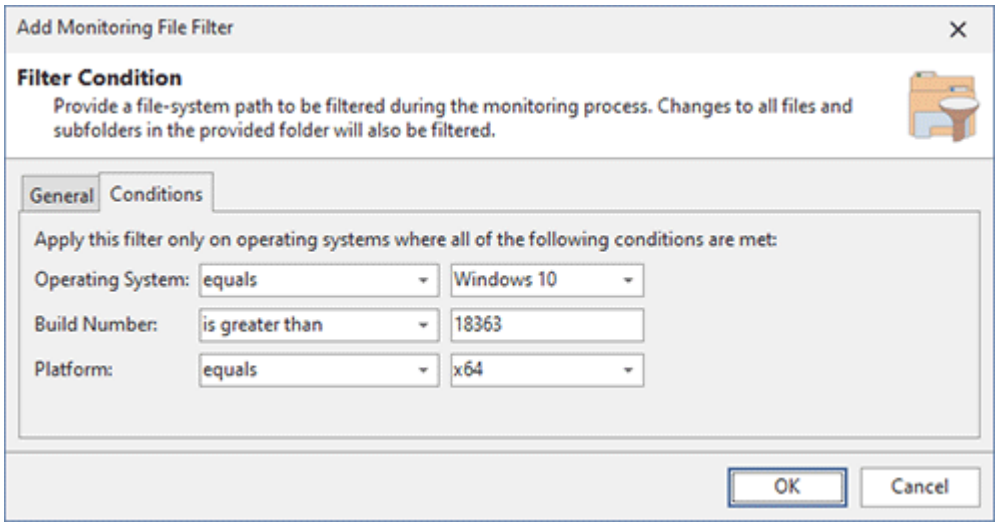
When configuring a Monitoring File Filter Pic 2, you should provide a file system path to be filtered. All changes to the specified path are ignored during a monitoring process. You can provide the path manually or select it through the file picker using the ... button built into the edit box. It is possible to use the system folder definition placeholders while specifying the filter, these placeholders are replaced automatically with corresponding file system paths during an installation monitoring process. See System Folder Definition Placeholders section of this document for the list of available placeholders.

Each system folder definition should be proceeded with **${** and succeeded with **}$**, e. g. **${ProgramFilesFolder}$**.

Copyright © 2001-2023 EMCO. All rights reserved.

The and buttons built into the edit box can help you with replacing specific paths to their system folder definitions and vice versa. If you enable the **Use a regular expression while specifying the filter value** option, you can define a filter that will exclude all paths matching the expression from monitoring results. See the How should I correctly specify the filter value? section of this document for a detailed description of possible values.

A filter can be used in specific environments if you configure conditions for it. On the **Conditions** tab Pic 3 you can specify the **Operating System**, the **Build Number** (of the OS) and the **Platform** where the filter should be applied. You can use one or multiple criteria together with the comparison operators to specify environments for the filter. If you specify a condition, you can see its value in the filters list. Using conditions can be helpful if you plan to perform monitoring on different OSs so that you can configure OS/platform-specific filters, if required.



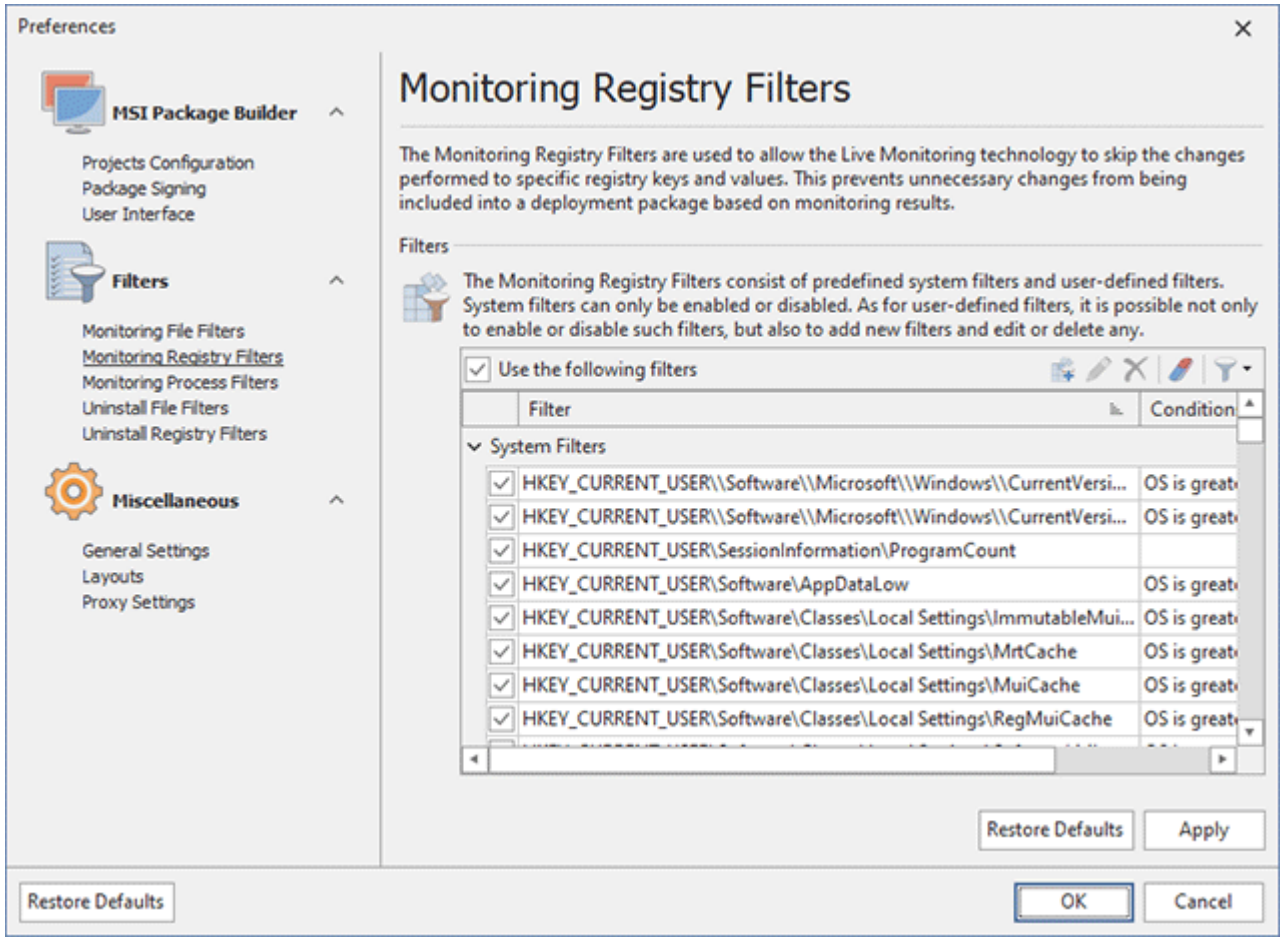

**Pic 3. Configuring OS and platform conditions**

To edit a user-defined Monitoring File Filter select it in the filters table and choose the **Edit** menu item from the pop-up menu or press the **Edit** button on the toolbar. The filter editing process is similar to the above-stated creation process. To delete the user-defined filters that are no longer needed, you can select those items in the filters table and choose the **Delete** menu item from the pop-up menu or press the **Delete** button on the toolbar. It is also possible to delete all user-defined filters using the **Clear User Filters** menu item and the corresponding button on the toolbar.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Monitoring Registry Filters Page

The Monitoring Registry Filters are used to allow the Live Monitoring technology to skip the changes performed to specific registry keys. This prevents unnecessary changes from being included into a deployment package based on monitoring results. To configure the Monitoring Registry Filters open the program preferences using the **Preferences** button from the **Application Menu** and click the **Monitoring Registry Filters** link on the navigation bar on the left of the **Preferences** dialog within the **Filters** group Pic 1.




**Pic 1. Configuring Monitoring Registry Filters**

The filters are divided into two categories; those are **User Filters** and **System Filters**. The system filters are predefined ones and cannot be deleted, but can be disabled, if required, using the **Disable Selected** item from the pop-up menu. You can then re-enable any filter using the **Enable Selected** menu item. To enable/disable a filter and to check if it is enabled, you can use the check boxes located on the left of each row representing the filter. It is also possible to refuse filtering registry keys by disabling the **Use the following filters** option.

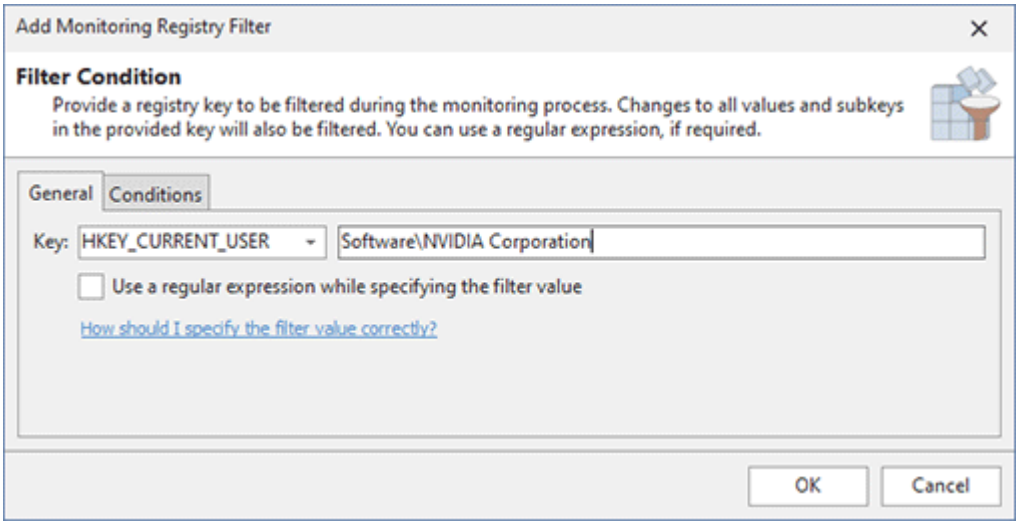
### Toolbar Overview

| | |
|---|---|
| | **Add Filter**<br>The **Add Filter** button should be used to add a new user-defined **Monitoring Registry Filter**. |
| | **Edit**<br>The **Edit** button allows you to change the selected user-defined **Monitoring Registry Filter**. |

Copyright © 2001-2023 EMCO. All rights reserved.

| | |
|---|---|
|  | **Delete**<br>The **Delete** button allows you to delete the selected user-defined **Monitoring Registry Filter** items. |
|  | **Clear User Filters**<br>The **Clear User Filters** button should be used to delete all user-defined **Monitoring Registry Filter** items. |
|  | **View Mode**<br>The **View Mode** button is intended to show only the filters that meet specific criteria. You may show all the filters, the filters applicable for the current OS only, or specify a custom criterion for filters. |

As for the user filters, those are the user-defined ones. To add a new filter, use the **Add Filter** item from the pop-up menu or press the **Add Filter** button on the toolbar.

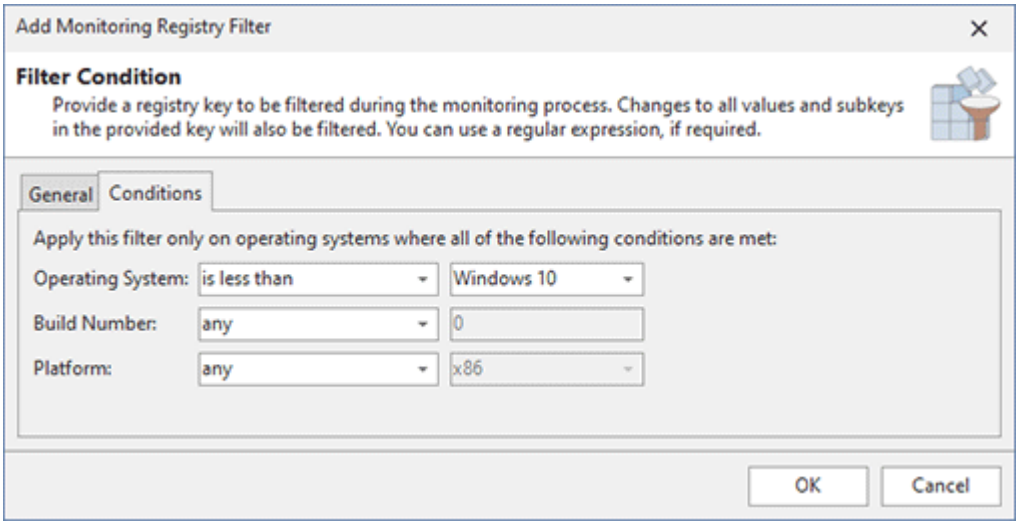



**Pic 2. Configuring a Monitoring Registry Filter**

Copyright © 2001-2023 EMCO. All rights reserved.

When configuring a Monitoring Registry Filter Pic 2, you should define a registry key to be filtered. All changes to the specified key and its sub-keys are ignored during a monitoring process. To define a registry key you first choose the root key from the drop down list and then type in the path to the key in the text edit. If you enable the **Use a regular expression while specifying the filter value** option, you can define a filter that will exclude changes to all keys matching the expression from monitoring results. See the How should I correctly specify the filter value? section of this document for a detailed description of possible values.

A filter can be used in specific environments if you configure conditions for it. On the **Conditions** tab Pic 3 you can specify the **Operating System**, the **Build Number** (of the OS) and the **Platform** where the filter should be applied. You can use one or multiple criteria together with the comparison operators to specify environments for the filter. If you specify a condition, you can see its value in the filters list. Using conditions can be helpful if you plan to perform monitoring on different OSs so that you can configure OS/platform-specific filters, if required.



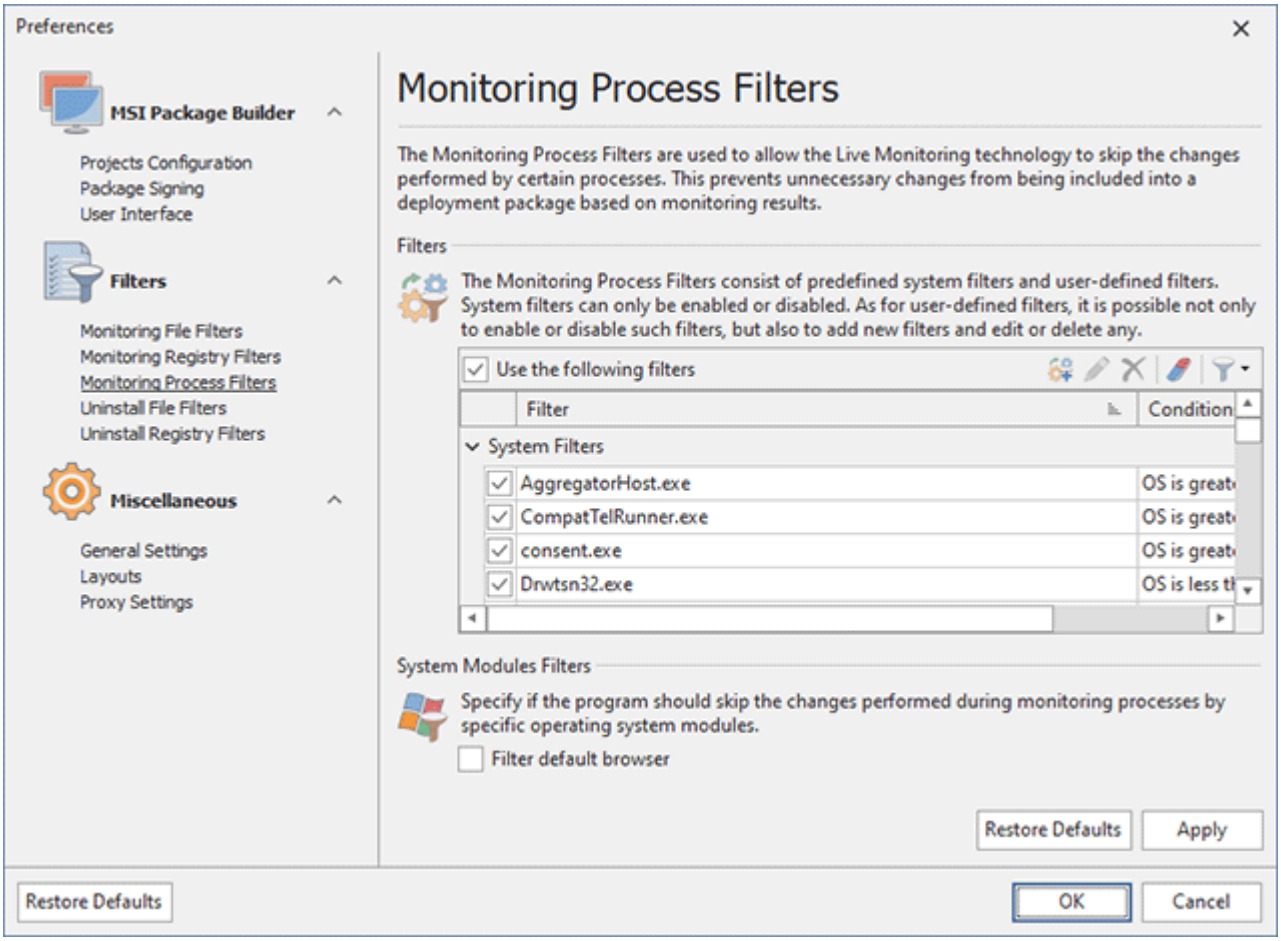

**Pic 3. Configuring OS and platform conditions**

To edit a user-defined Monitoring Registry Filter select it in the filters table and choose the **Edit** menu item from the pop-up menu or press the **Edit** button on the toolbar. The filter editing process is similar to the above-stated creation process. To delete the user-defined filters that are no longer needed, you can select those items in the filters table and choose the **Delete** menu item from the pop-up menu or press the **Delete** button on the toolbar. It is also possible to delete all user-defined filters using the **Clear User Filters** menu item and the corresponding button on the toolbar.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Monitoring Processes Filters Page

The Monitoring Process Filters are used to allow the Live Monitoring technology to skip the changes performed by certain processes. This prevents unnecessary changes from being included into a deployment package based on monitoring results. To configure the Monitoring Process Filters open the program preferences using the **Preferences** button from the **Application Menu** and click the **Monitoring Process Filters** link on the navigation bar on the left of the **Preferences** dialog within the **Filters** group Pic 1.




**Pic 1. Configuring Monitoring Process Filters**

The filters are divided into two categories; those are **User Filters** and **System Filters**. The system filters are predefined ones and cannot be deleted, but can be disabled, if required, using the **Disable Selected** item from the pop-up menu. You can then re-enable any filter using the **Enable Selected** menu item. To enable/disable a filter and to check if it is enabled, you can use the check boxes located on the left of each row representing the filter. It is also possible to refuse filtering processes by disabling the **Use the following filters** option.

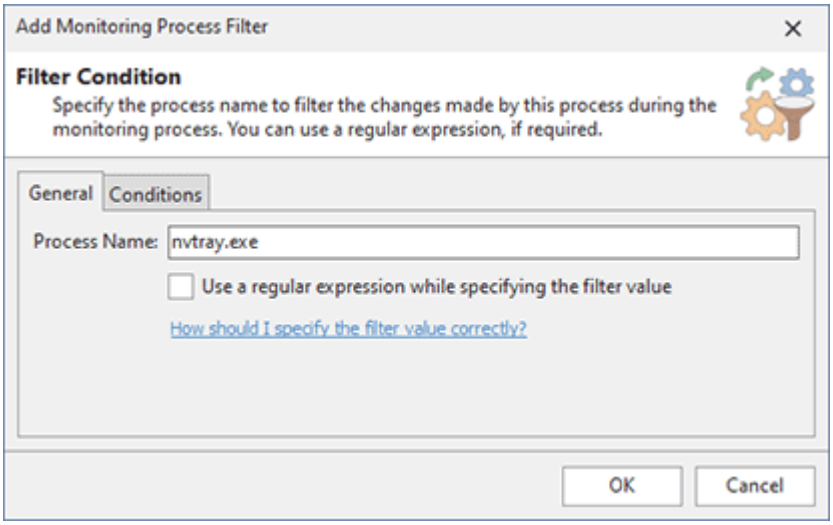
### Toolbar Overview

| | |
|---|---|
| | **Add Filter**<br>The **Add Filter** button should be used to add a new user-defined **Monitoring Process Filter**. |
| | **Edit**<br>The **Edit** button allows you to change the selected user-defined **Monitoring Process Filter**. |

Copyright © 2001-2023 EMCO. All rights reserved.

| | |
|---|---|
|  | **Delete**<br>The **Delete** button allows you to delete the selected user-defined **Monitoring Process Filter** items. |
|  | **Clear User Filters**<br>The **Clear User Filters** button should be used to delete all user-defined **Monitoring Process Filter** items. |
|  | **View Mode**<br>The **View Mode** button is intended to show only the filters that meet specific criteria. You may show all the filters, the filters applicable for the current OS only, or specify a custom criterion for filters. |

As for the user filters, those are the user-defined ones. To add a new filter, use the **Add Filter** item from the pop-up menu or press the **Add Filter** button on the toolbar.

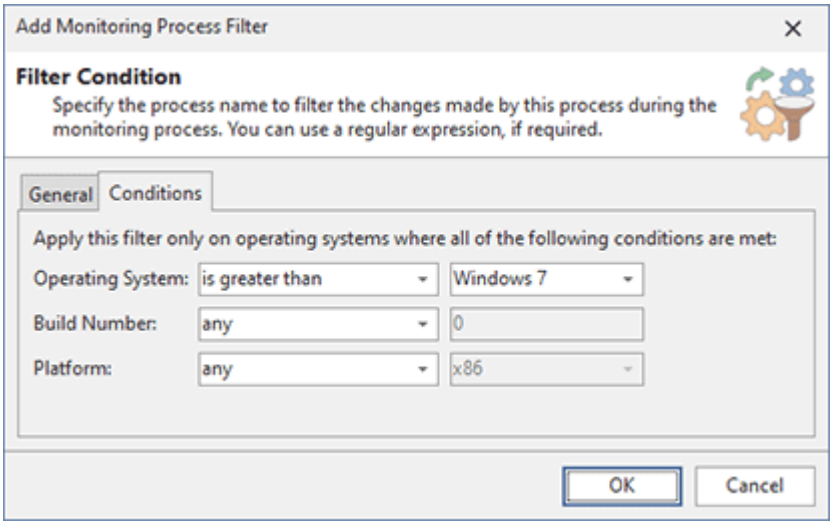



**Pic 2. Configuring a Monitoring Process Filter**

Copyright © 2001-2023 EMCO. All rights reserved.

When configuring a Monitoring Process Filter Pic 2, you should define a process name to be filtered. All changes performed by the specified process are ignored during a monitoring. If you enable the **Use a regular expression while specifying the filter value** option, you can define a filter that will exclude changes by all processes which names are matching the expression from monitoring results. See the How should I correctly specify the filter value? section of this document for a detailed description of possible values.

A filter can be used in specific environments if you configure conditions for it. On the **Conditions** tab Pic 3 you can specify the **Operating System**, the **Build Number** (of the OS) and the **Platform** where the filter should be applied. You can use one or multiple criteria together with the comparison operators to specify environments for the filter. If you specify a condition, you can see its value in the filters list. Using conditions can be helpful if you plan to perform monitoring on different OSs so that you can configure OS/platform-specific filters, if required.




**Pic 3. Configuring OS and platform conditions**

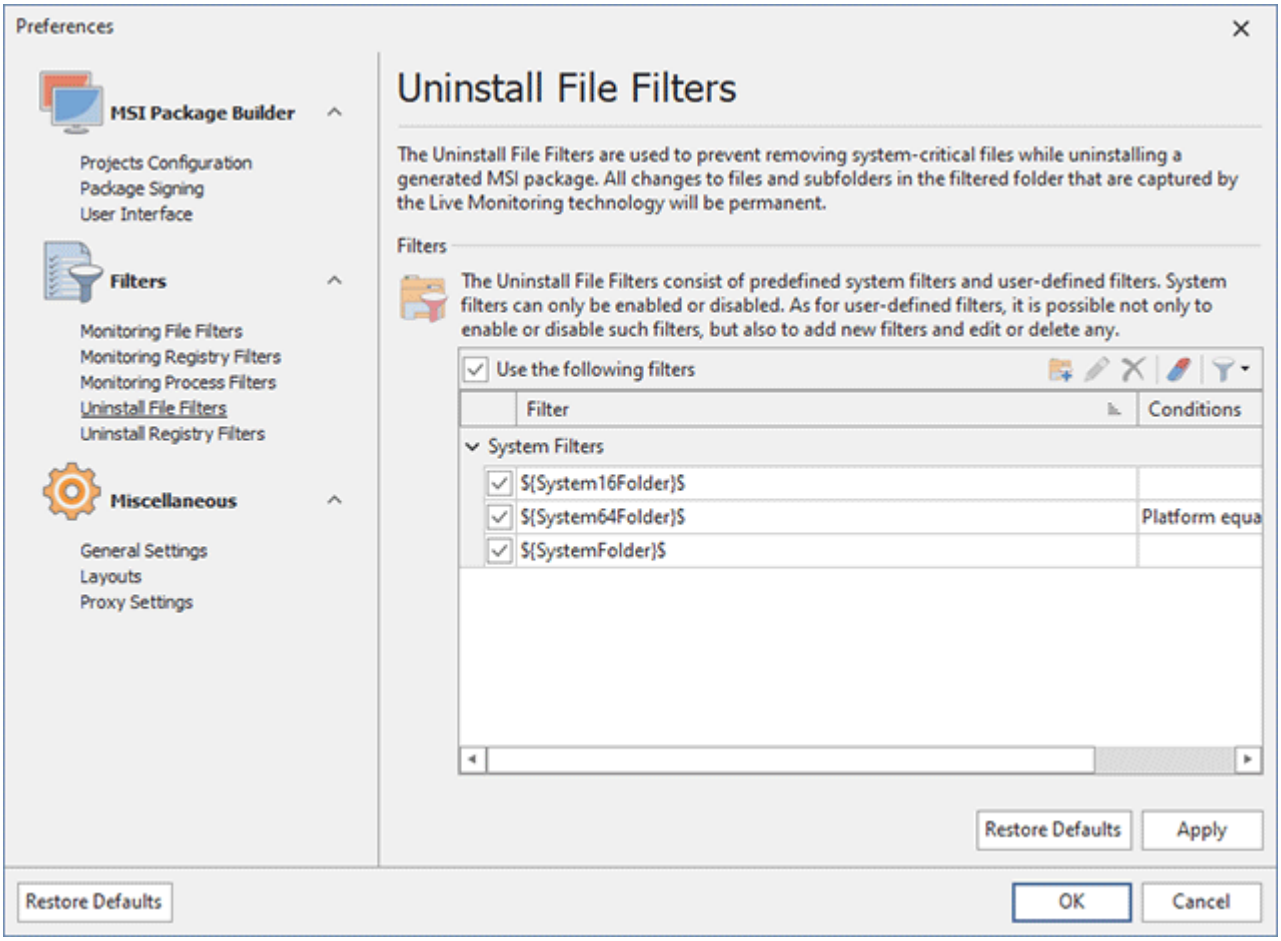
To edit a user-defined Monitoring Process Filter select it in the filters table and choose the **Edit** menu item from the pop-up menu or press the **Edit** button on the toolbar. The filter editing process is similar to the above-stated creation process. To delete the user-defined filters that are no longer needed, you can select those items in the filters table and choose the **Delete** menu item from the pop-up menu or press the **Delete** button on the toolbar. It is also possible to delete all user-defined filters using the **Clear User Filters** menu item and the corresponding button on the toolbar.

In addition to using the generic process monitoring filters, you are suggested to define if the changes performed by specific system modules should be filtered or not. Currently, you can choose if the activity of a default browser should be taken into account during the monitoring process using the **Filter default browser** option.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Uninstall File Filters Page

The Uninstall File Filters allow you to prevent removing system-critical files while uninstalling a generated deployment package. All changes to files and folders in the filtered folder that are captured by the Live Monitoring technology are treated as permanent. To configure the Uninstall File Filters open the program preferences using the **Preferences** button from the **Application Menu** and click the **Uninstall File Filters** link on the navigation bar on the left of the **Preferences** dialog within the **Filters** group Pic 1.




**Pic 1. Configuring Uninstall File Filters**

The filters are divided into two categories; those are **User Filters** and **System Filters**. The system filters are predefined ones and cannot be deleted, but can be disabled, if required, using the **Disable Selected** item from the pop-up menu. You can then re-enable any filter using the **Enable Selected** menu item. To enable/disable a filter and to check if it is enabled, you can use the check boxes located on the left of each row representing the filter. It is also possible to refuse filtering files and folders from uninstall by disabling the **Use the following filters** option.

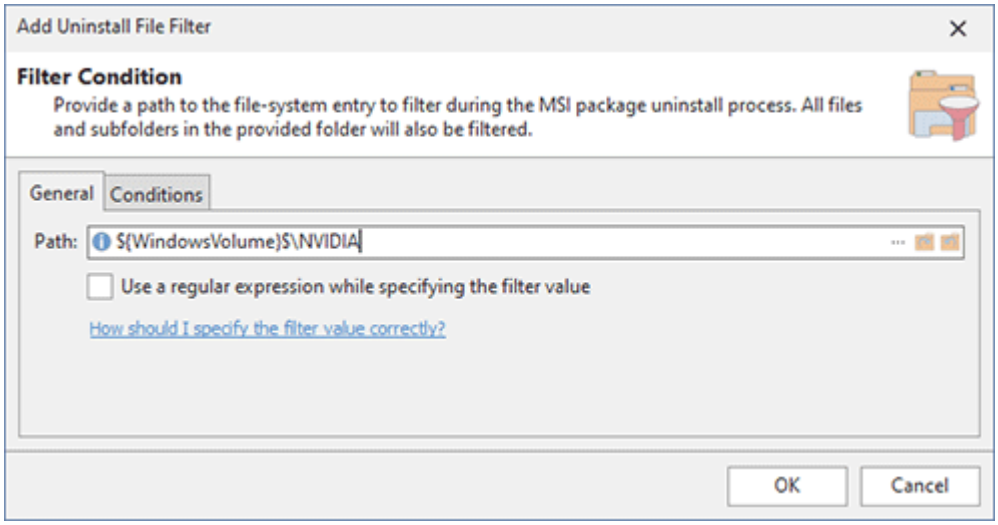
### Toolbar Overview

| | |
|---|---|
|  | **Add Filter**<br>The **Add Filter** button should be used to add a new user-defined **Uninstall File Filter**. |
|  | **Edit**<br>The **Edit** button allows you to change the selected user-defined **Uninstall File Filter**. |

Copyright © 2001-2023 EMCO. All rights reserved.

| | |
|---|---|
|  | **Delete**<br>The **Delete** button allows you to delete the selected user-defined **Uninstall File Filter** items. |
|  | **Clear User Filters**<br>The **Clear User Filters** button should be used to delete all user-defined **Uninstall File Filter** items. |
|  | **View Mode**<br>The **View Mode** button is intended to show only the filters that meet specific criteria. You may show all the filters, the filters applicable for the current OS only, or specify a custom criterion for filters. |

As for the user filters, those are the user-defined ones. To add a new filter, use the **Add Filter** item from the pop-up menu or press the **Add Filter** button on the toolbar.




**Pic 2. Configuring an Uninstall File Filter**

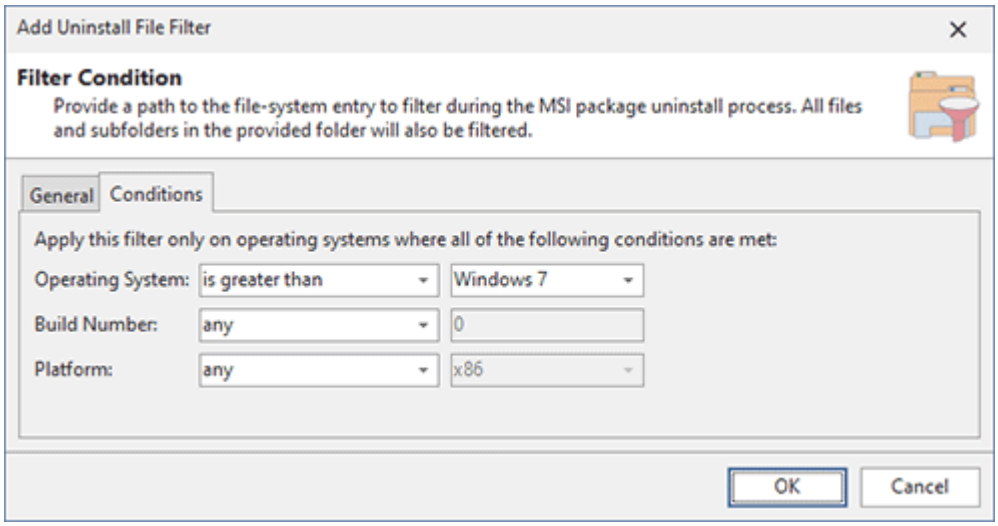
When configuring an Uninstall File Filter Pic 2, you should provide a file system path to be treated as persistent. No files and folders in the specified path, including the one represented with this path, will be deleted while uninstalling a generated deployment package. You can provide the path manually or select it through the file picker using the ... button built into the edit box. It is possible to use the system folder definition placeholders while specifying the filter, these placeholders are automatically replaced with the corresponding file system paths during a deployment package uninstall process. See System Folder Definition Placeholders section of this document for the list of available placeholders.

Each system folder definition should be proceeded with **${** and succeeded with **}$**, e. g. **${SystemFolder}$**.

Copyright © 2001-2023 EMCO. All rights reserved.

The  and  buttons built into the edit box can help you with replacing specific paths to their system folder definitions and vice versa. If you enable the **Use a regular expression while specifying the filter value** option, you can define a filter that will treat as permanent all paths matching the expression. See the How should I correctly specify the filter value? section of this document for a detailed description of possible values.

A filter can be used in specific environments if you configure conditions for it. On the **Conditions** tab Pic 3 you can specify the **Operating System**, the **Build Number** (of the OS) and the **Platform** where the filter should be applied. You can use one or multiple criteria together with the comparison operators to specify environments for the filter. If you specify a condition, you can see its value in the filters list. Using conditions can be helpful if you plan to perform monitoring on different OSs so that you can configure OS/platform-specific filters, if required.



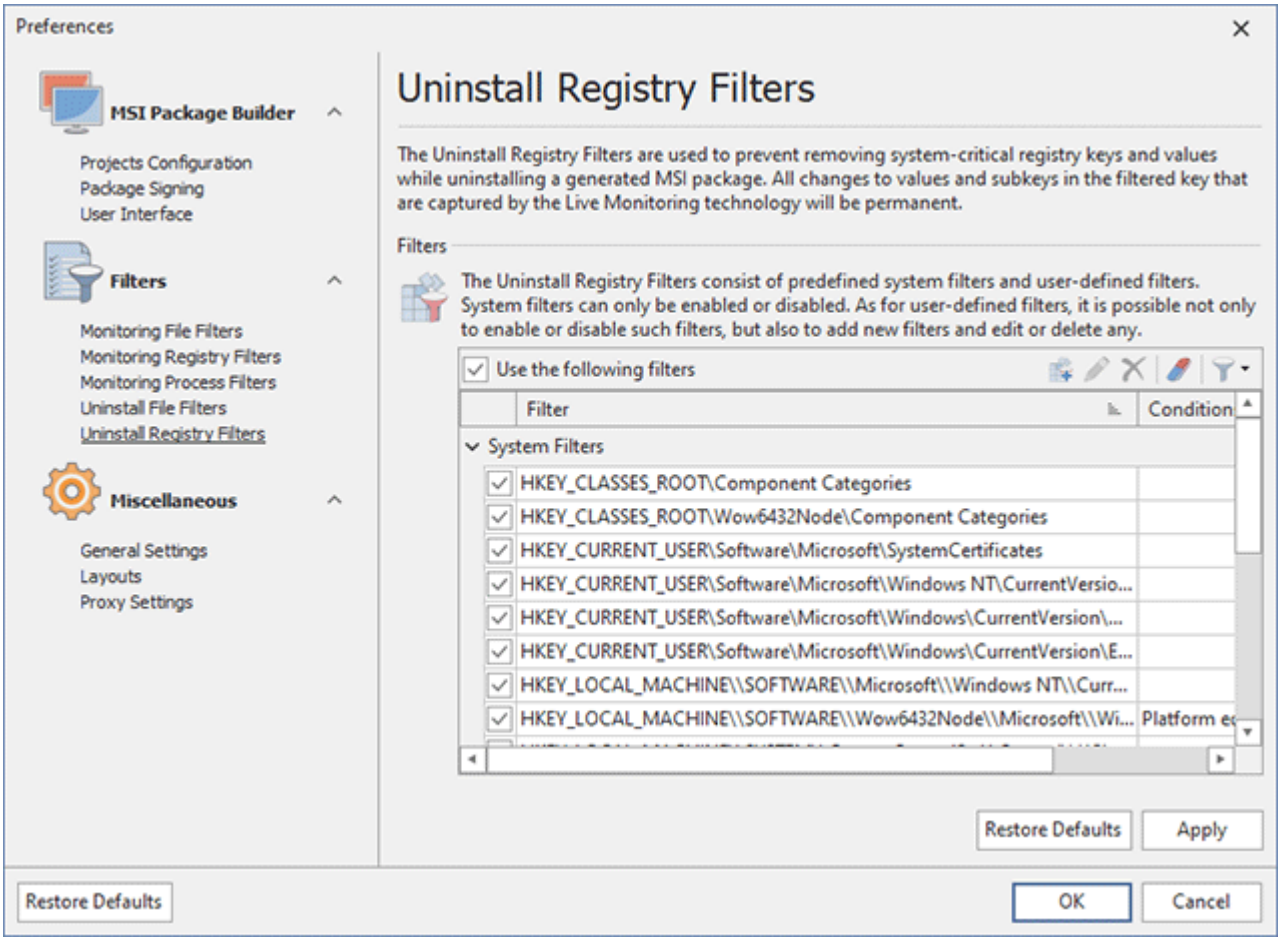

**Pic 3. Configuring OS and platform conditions**

To edit a user-defined Uninstall File Filter select it in the filters table and choose the **Edit** menu item from the pop-up menu or press the **Edit** button on the toolbar. The filter editing process is similar to the above-stated creation process. To delete the user-defined filters that are no longer needed, you can select those items in the filters table and choose the **Delete** menu item from the pop-up menu or press the **Delete** button on the toolbar. It is also possible to delete all user-defined filters using the **Clear User Filters** menu item and the corresponding button on the toolbar.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Uninstall Registry Filters Page

The Uninstall Registry Filters allow you to prevent removing system-critical registry keys and values while uninstalling a generated deployment package. All changes to values and sub-keys in the filtered key that are captured by the Live Monitoring technology are treated as permanent. To configure the Uninstall Registry Filters open the program preferences using the **Preferences** button from the **Application Menu** and click the **Uninstall Registry Filters** link on the navigation bar on the left of the **Preferences** dialog within the **Filters** group Pic 1.




**Pic 1. Configuring Uninstall Registry Filters**

The filters are divided into two categories; those are **User Filters** and **System Filters**. The system filters are predefined ones and cannot be deleted, but can be disabled, if required, using the **Disable Selected** item from the pop-up menu. You can then re-enable any filter using the **Enable Selected** menu item. To enable/disable a filter and to check if it is enabled, you can use the check boxes located on the left of each row representing the filter. It is also possible to refuse filtering registry keys from uninstall by disabling the **Use the following filters** option.

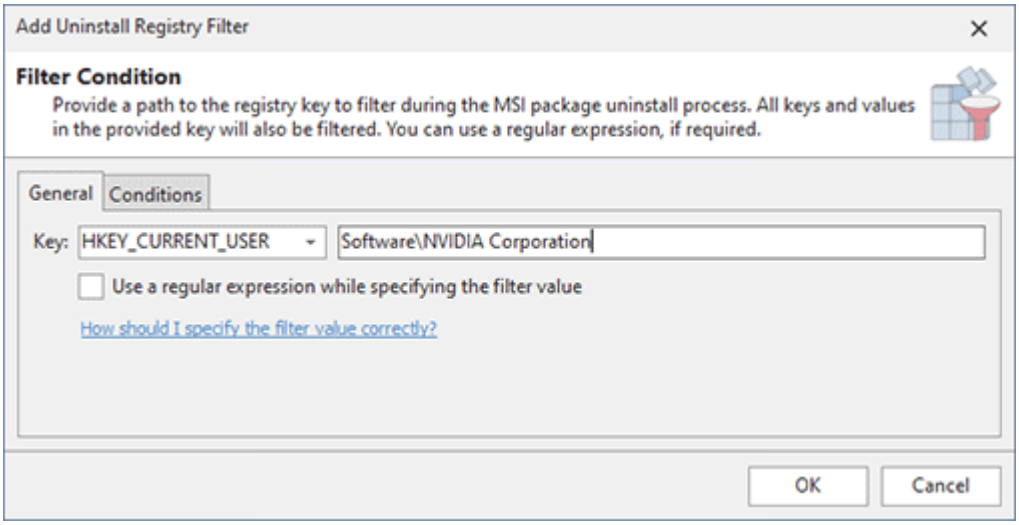
### Toolbar Overview

| | |
|---|---|
|  | **Add Filter**<br>The **Add Filter** button should be used to add a new user-defined **Uninstall Registry Filter**. |
|  | **Edit**<br>The **Edit** button allows you to change the selected user-defined **Uninstall Registry Filter**. |

Copyright © 2001-2023 EMCO. All rights reserved.

| | |
|---|---|
|  | **Delete**<br>The **Delete** button allows you to delete the selected user-defined **Uninstall Registry Filter** items. |
|  | **Clear User Filters**<br>The **Clear User Filters** button should be used to delete all user-defined **Uninstall Registry Filter** items. |
|  | **View Mode**<br>The **View Mode** button is intended to show only the filters that meet specific criteria. You may show all the filters, the filters applicable for the current OS only, or specify a custom criterion for filters. |

As for the user filters, those are the user-defined ones. To add a new filter, use the **Add Filter** item from the pop-up menu or press the **Add Filter** button on the toolbar.

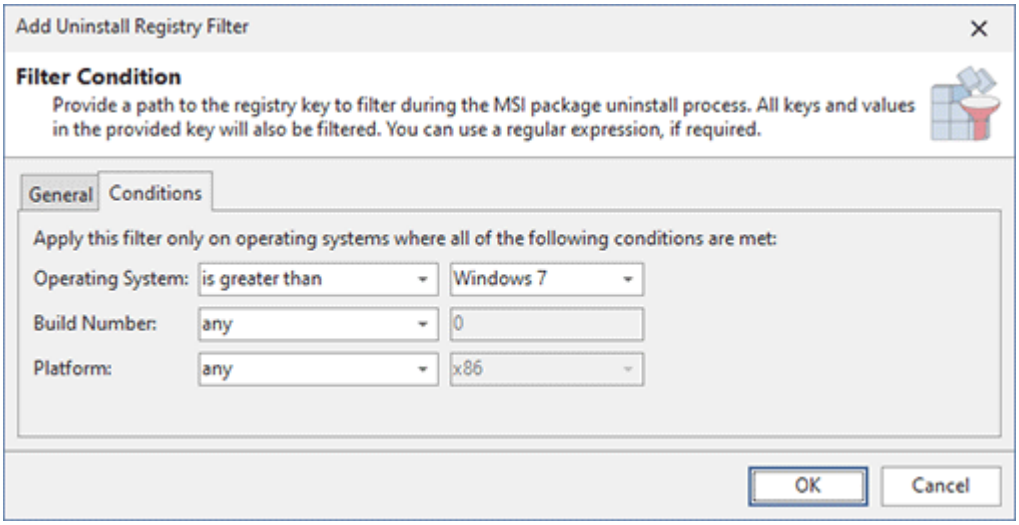



**Pic 2. Configuring an Uninstall Registry Filter**

Copyright © 2001-2023 EMCO. All rights reserved.

When configuring an Uninstall Registry Filter Pic 2, you should define a registry key to be treated as persistent. No keys and values from the specified key, including the key itself, will be deleted while uninstalling a generated deployment package. To define a registry key, you should first choose the root key from the drop down list and then type in the path to the key in the text edit. If you enable the **Use a regular expression while specifying the filter value** option, you can define a filter that will treat as permanent all keys matching the expression. See the How should I correctly specify the filter value? section of this document for a detailed description of possible values.

A filter can be used in specific environments if you configure conditions for it. On the **Conditions** tab Pic 3 you can specify the **Operating System**, the **Build Number** (of the OS) and the **Platform** where the filter should be applied. You can use one or multiple criteria together with the comparison operators to specify environments for the filter. If you specify a condition, you can see its value in the filters list. Using conditions can be helpful if you plan to perform monitoring on different OSs so that you can configure OS/platform-specific filters, if required.




**Pic 3. Configuring OS and platform conditions**

To edit a user-defined Uninstall Registry Filter select it in the filters table and choose the **Edit** menu item from the pop-up menu or press the **Edit** button on the toolbar. The filter editing process is similar to the above-stated creation process. To delete the user-defined filters that are no longer needed, you can select those items in the filters table and choose the **Delete** menu item from the pop-up menu or press the **Delete** button on the toolbar. It is also possible to delete all user-defined filters using the **Clear User Filters** menu item and the corresponding button on the toolbar.

## Miscellaneous Part

The **Miscellaneous** part of the program preferences should be used to configure the common MSI Package Builder options, such as automatic update settings, layouts configuration, the proxy settings to be used to connect to the Internet, etc. To open the **Preferences** dialog, click the **Preferences** button available from the **Application Menu**. Configure the available settings to best suit your needs.

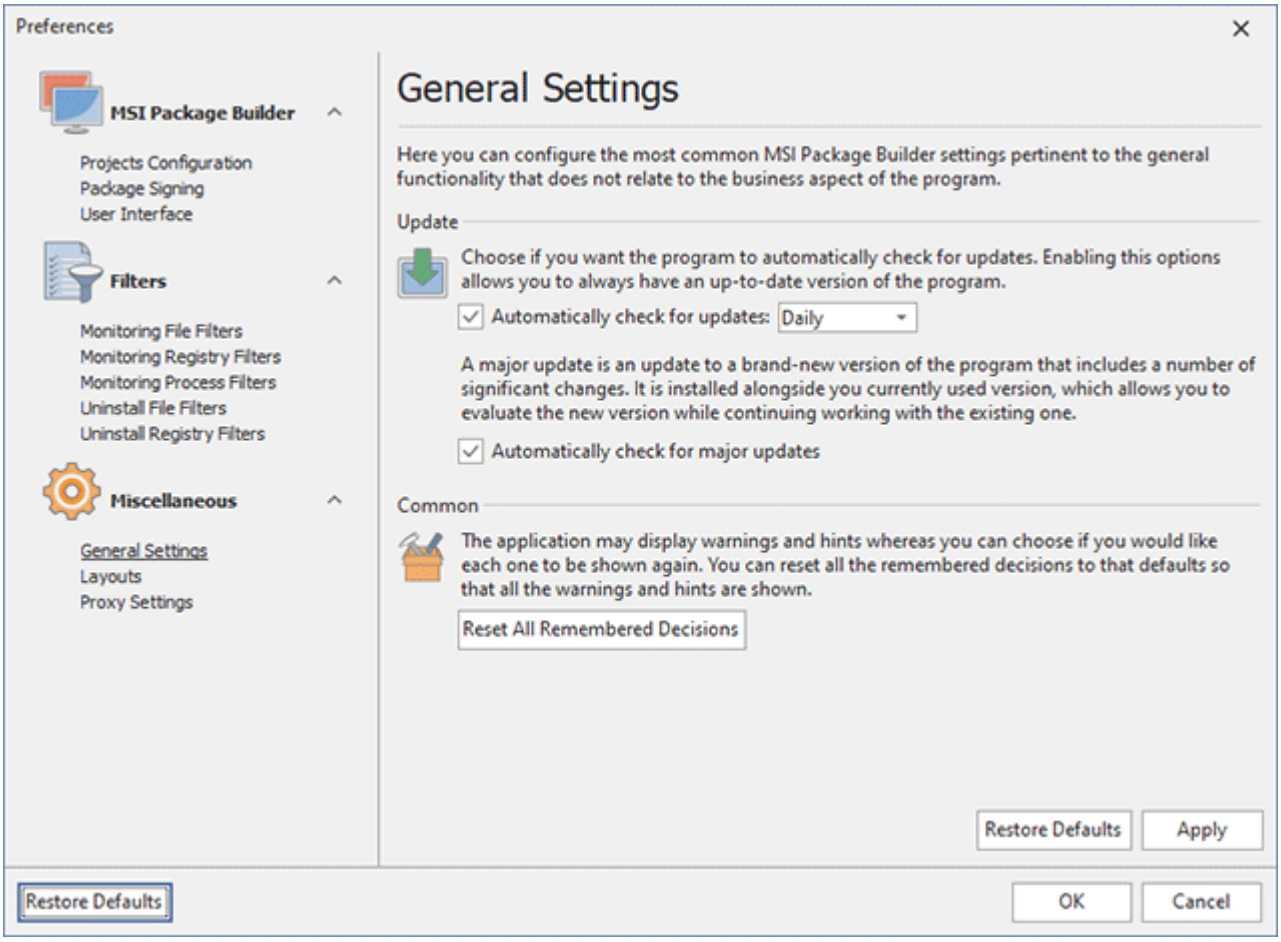
### What's Inside

General Settings Page
Layouts Page
Proxy Settings Page

Copyright © 2001-2023 EMCO. All rights reserved.

## General Settings Page

MSI Package Builder can automatically check for updates for you to always have the latest version of the program. You can configure this feature from the **General Settings** preference page. To open this page, click the **Preferences** button from the **Application Menu** and select the **General Settings** link in the navigation bar on the left in the **Preferences** dialog within the **Miscellaneous** group Pic 1.




**Pic 1. Configuring general settings**

MSI Package Builder can check for updates automatically every day or once a week. To enable an automatic checking for updates, check the **Automatically check for updates** option and choose the checking frequency between **Daily** and **Weekly**. You can also define if the program should check for major updates by changing the **Automatically check for major updates** option value.

If you use a proxy server to connect to the Internet and the required proxy settings are not provided, an automatic check for updates will not take place.

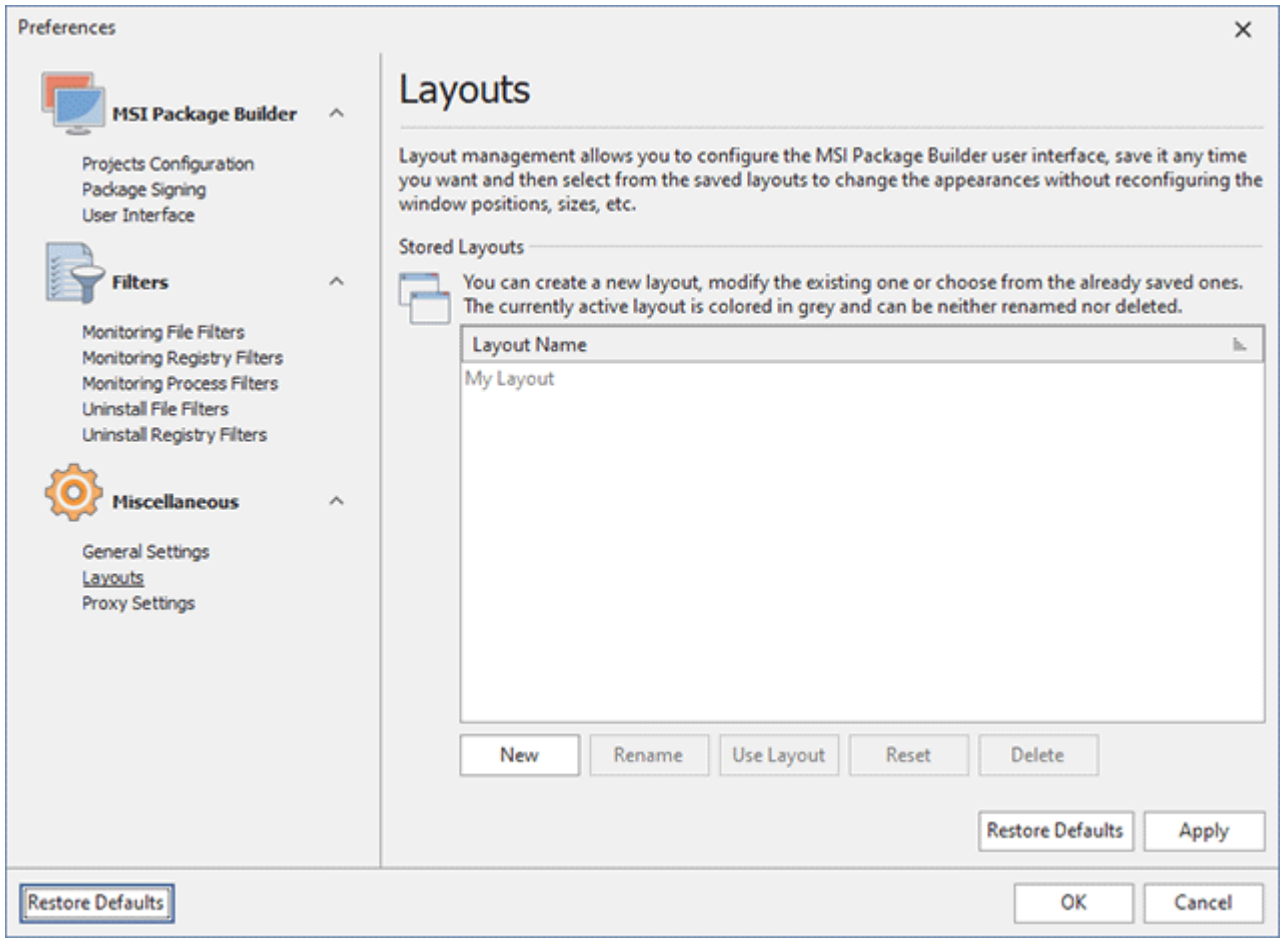
The application may display warnings and hints, and you can choose if you would like each one to be shown again. On this page, you can reset all the remembered decisions to the defaults so that all the warnings and hints are shown. Use the **Reset All Remembered Decisions** button to this purpose.

Copyright © 2001-2023 EMCO. All rights reserved.

## Layouts Page

Layout management technology allows you to configure MSI Package Builder user interface, save it any time you want and then select between saved layouts to change the appearance without reconfiguring windows positions, sizes, etc.

The **Layouts** preference page Pic 1 is designed to help you with windows layout management. To access this page click **Preferences** button from the **Application Menu** and select the appropriate link in the navigation bar on the left of the **Preferences** dialog.



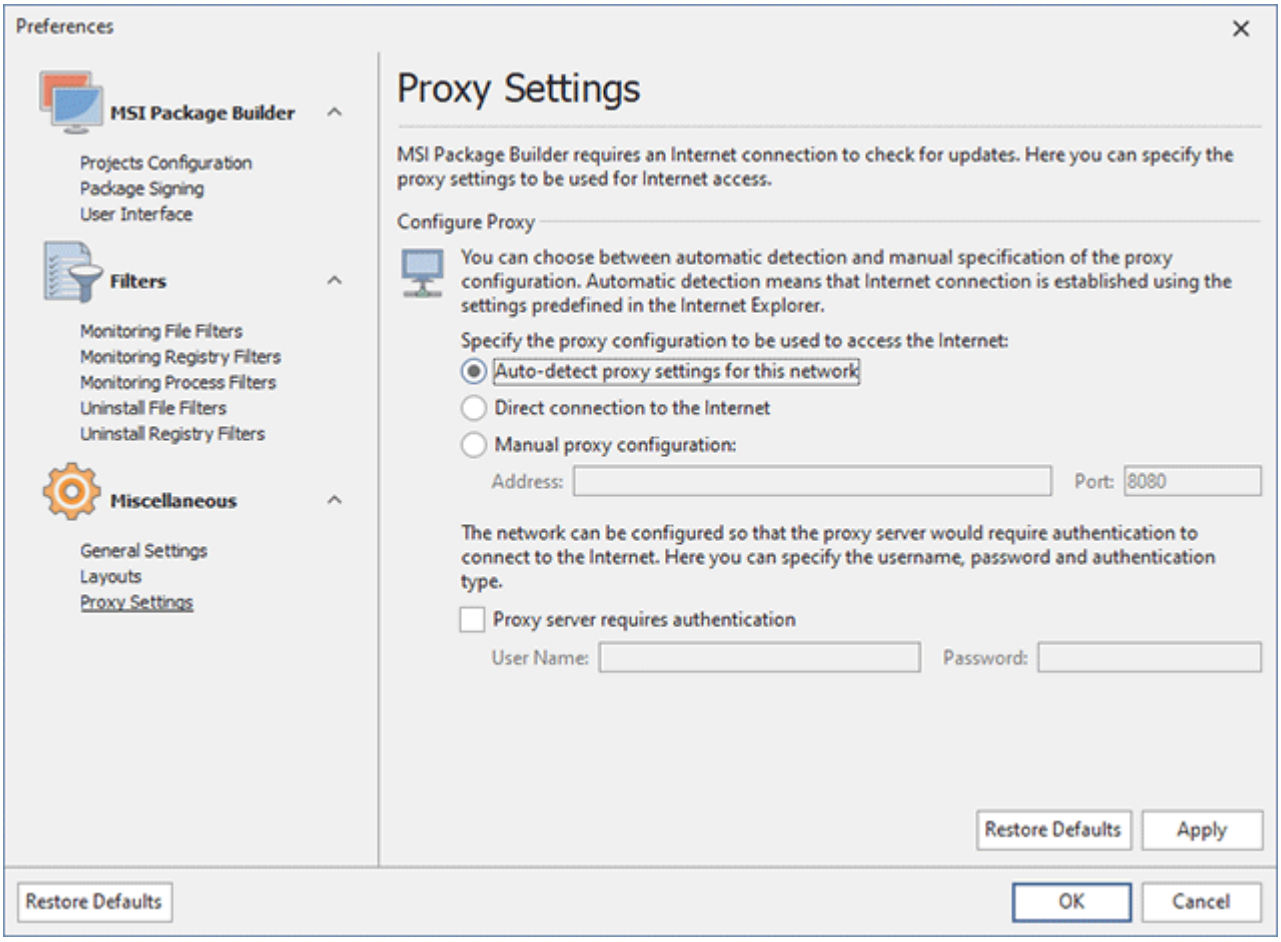

**Pic 1. Managing Layouts**

On the **Layouts** preference page Pic 1 you can create new layout, modify existing one or choose from already saved layouts. Currently active layout is colored gray and cannot be either renamed or deleted.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Proxy Settings Page

MSI Package Builder requires an Internet connection to support the **Live Update** and **Feedback** features. Therefore, if a proxy server has to be used to connect to the Internet, it should be configured on the **Proxy Settings** preference page Pic 1. To access this page, click the **Preferences** button from the **Application Menu** and select the appropriate link in the navigation bar on the left in the **Preferences** dialog within the **Miscellaneous** group.




**Pic 1. Proxy Settings**

On this page Pic 1, you may choose among three variants of the proxy configuration to be used by the program. If **Auto-detect proxy settings for this network** is chosen, the program uses the settings predefined in the **Internet Explorer**. If MSI Package Builder does not have to use a proxy server to connect to the Internet, the **Direct connection to the Internet** option should be chosen. The **Manual proxy configuration** option allows you to provide the proxy server address and port manually.

Both for the automatic detection and manual configuration, it is possible to specify if the proxy server requires authentication and what credentials should be used to connect to the proxy server.

*Copyright © 2001-2023 EMCO. All rights reserved.*

# Chapter 12: Evaluation of the Program

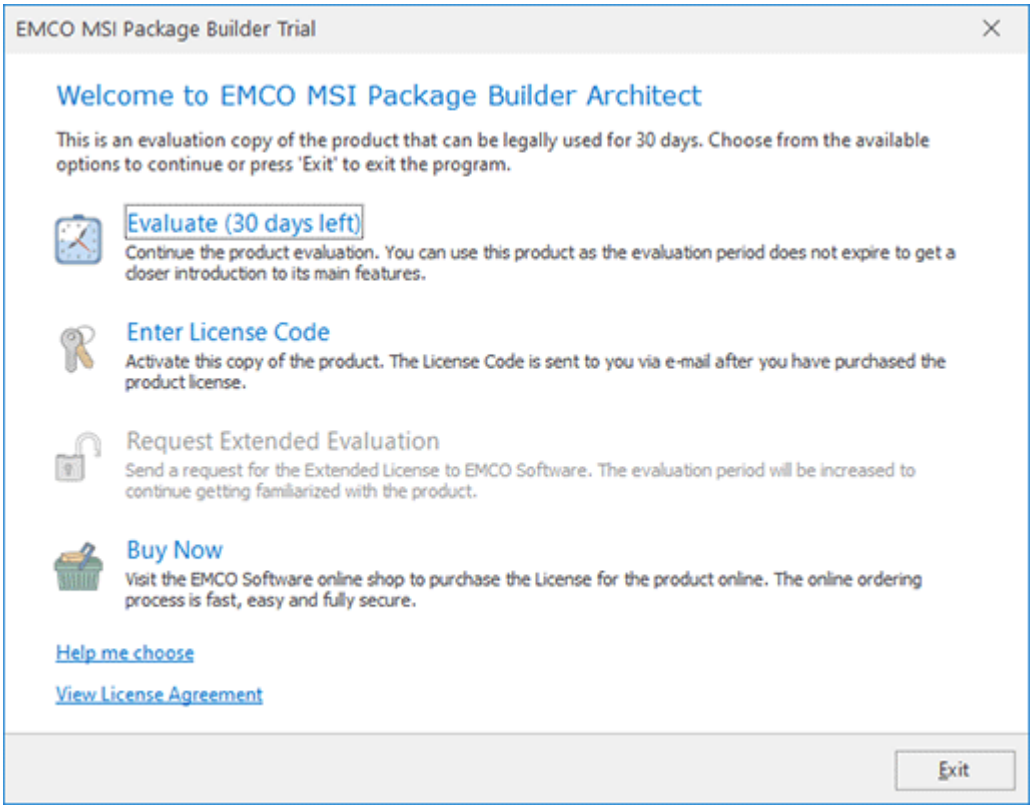
MSI Package Builder is a shareware product, so you can try it before ordering a license. The download includes a free 30-days trial of the Enterprise edition of the program. It provides access to all the available features. After the trial expires, the program reverts to the Free edition, and you can keep using it as long as you like. You can order a license for a commercial edition of the program to register it and use the program's commercial features.

This chapter will cover the particularities of the evaluation mode, tell you how and where you can get the license code and how you can request the extended evaluation. Read this chapter carefully to face no difficulties during the EMCO MSI Package Builder evaluation.

Copyright © 2001-2023 EMCO. All rights reserved.

## Evaluation Wizard

As long as the EMCO MSI Package Builder is not activated on each program startup the **Evaluation Wizard** Pic 1 is displayed on the screen, showing you the information about the evaluation process and providing with quick links for the program activation, purchase and extended evaluation request.



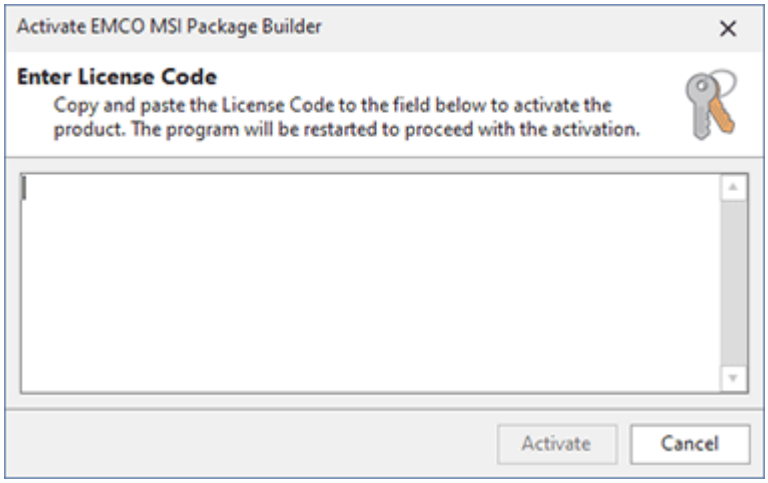

**Pic 1. The EMCO MSI Package Builder Evaluation Wizard welcome page**

*Copyright © 2001-2023 EMCO. All rights reserved.*

The welcome page of the **Evaluation Wizard** allows you to choose between four options to continue. Those are **Evaluate**, **Enter License Code**, **Request extended evaluation** and **Buy now**. Optionally you can press **Exit** button to close the program. In this section we will help you to choose the option that will best fit your needs.

The **Evaluate** option shows you the time left until the evaluation period expires. You should choose this option to continue the evaluation process – the wizard will be closed and you can start working with MSI Package Builder. You can use the program as long as the evaluation period does not expire to get a closer introduction to its main features.

It you have already purchased the license for using the EMCO MSI Package Builder you should choose the **Enter License Code** option to activate the program. This options also should be chosen if the extended evaluation request has been approved by EMCO Software and you have been provided with the Extended License. If you are having problems with finding the License Code refer to the Where can I get my License Code? section of this document. After choosing the **Enter License Code** option the program activation page is displayed on the screen Pic 2.

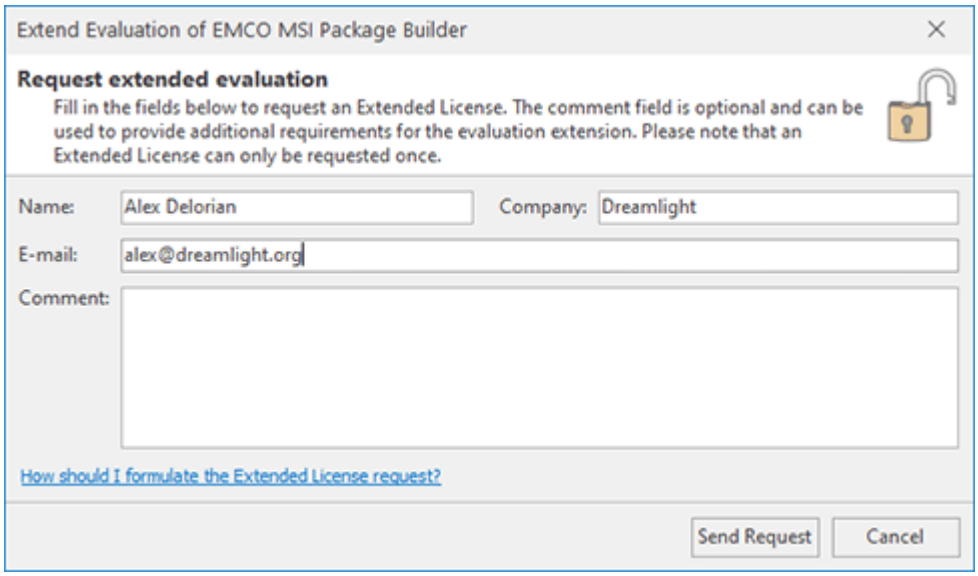



**Pic 2. Activating EMCO MSI Package Builder**

Copyright © 2001-2023 EMCO. All rights reserved.

To activate EMCO MSI Package Builder copy and paste the License Code to the input field on this page and press **Activate** - the program will be restarted to activate.

If the evaluation period has expired and you are not sure you have fully introduced yourself to EMCO MSI Package Builder main features you can once request the extended evaluation. As soon as the request is processed by EMCO Software you are provided with the Extended License to prolong the evaluation period. To request the Extended License you should choose the **Request extended evaluation** option. After choosing this option the request form will appear on the screen Pic 3.




**Pic 3. Requesting an extended evaluation**

To request the Extended License fill the fields on the request form regarding the recommendations and press the **Send Request** button.

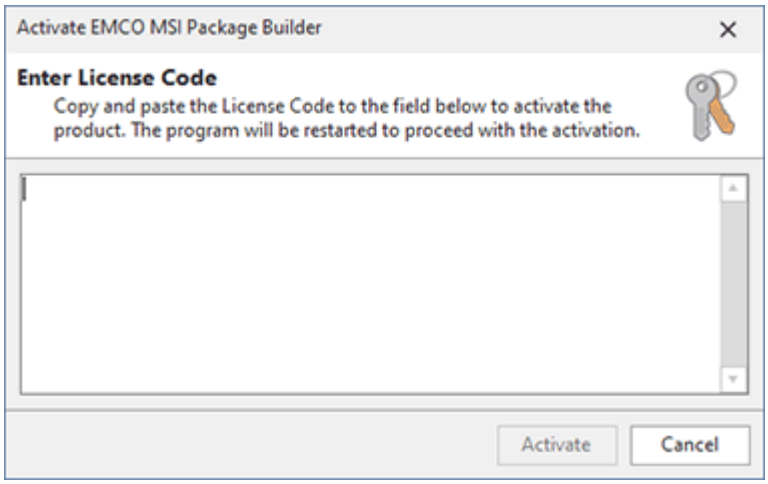
The **Evaluation Wizard** also provides you with the short cut action that allows you to visit EMCO Software web store. To use this feature choose the **Buy Now** option. The on-line ordering process is fast, easy, and fully secure.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Where can I get my License Code?

After you have purchased the license for using EMCO MSI Package Builder our experts will generate the License Code and send it to you via e-mail to the address you have specified during the purchasing process. You are supposed to receive two e-mail messages – one with the License Code written in the message body and one with the attached text file (license.txt), containing the license. It is your choice to use any message because both License Codes are identical.

To activate the program the License Code received via e-mail should be copied and pasted to the program activation form Pic 1. This form can be reached using the **Enter License Code** button from the **Information** group on the Ribbon bar or by choosing the appropriate option in the **Evaluation Wizard**.



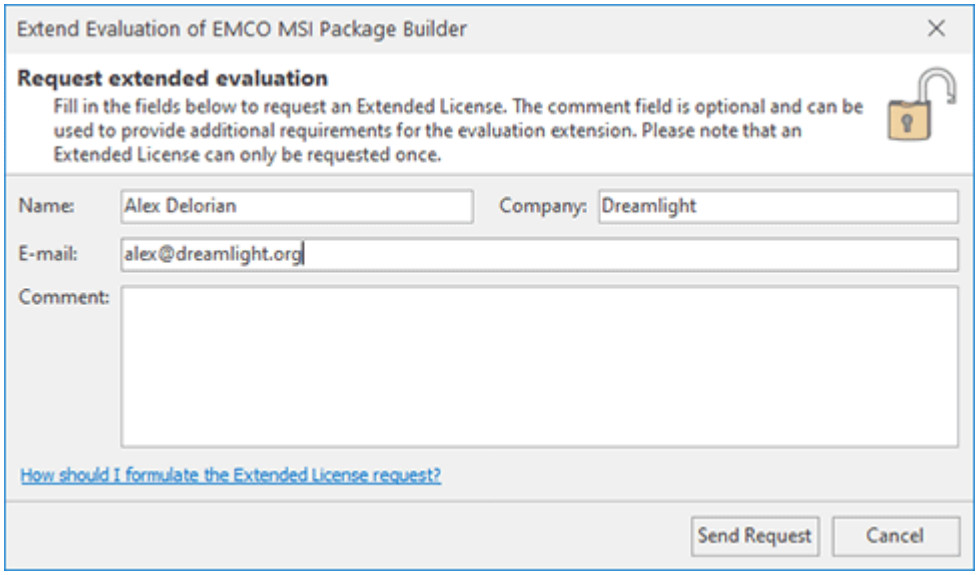

**Pic 1. Activating EMCO MSI Package Builder**

To activate MSI Package Builder, copy and paste the License Code to the input field and press **Activate** - the program will be restarted to activate.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## How should I formulate the Extended License request?

The Extended License is used to prolong the evaluation period to get a closer look at EMCO MSI Package Builder. This feature can be reached by choosing the appropriate option in the **Evaluation Wizard**. Here we'll give you the recommendations on filling the **Request extended evaluation** form Pic 1.




**Pic 1. The Request Extended Evaluation form**

In the **Request extended evaluation** form you should obligatory specify your name in the **Name** field, company name in the **Company** field and your e-mail address in the **E-mail** field.

Though the **Comment** field is optional it is strongly recommended to use this field for providing EMCO Software with the reason of requesting the Extended License. Please notice that EMCO Software reserves the right to decline the request without providing a requester with any explanations.

If the extended license request is approved by EMCO Software experts you'll receive the License Code to the e-mail address specified.

Copyright © 2001-2023 EMCO. All rights reserved.

# Chapter 13: Program Updates

EMCO Software cares for versatile needs of the users of EMCO programs and fully understands their wish to have the most up-to-date software installed on there PCs. That is why we provide you with an easy update feature. You do not need to browse the Internet again and again to find out if any updates are available – MSI Package Builder will do this work for you. Checking for updates can be performed both manually and automatically. This chapter describes the Live Update process for the current major version of the program and the Major Update feature which allows you to get a brand new version of MSI Package Builder quickly and easily.

Copyright © 2001-2023 EMCO. All rights reserved.

## Live Update

MSI Package Builder can be easily updated with just a few clicks. The update process is performed via an Internet connection using preconfigured proxy settings.




**Check for Updates**

The **Check for Updates** button from the **Update** Ribbon group should be used to check for new versions of MSI Package Builder.

MSI Package Builder can check for updates automatically. You can configure the program behavior regarding the automatic check for updates on the **General Settings** preference page.

To check for updates, click the **Check for Updates** button from the **Application Menu** or from the **Update** group of the **Program** Ribbon page. MSI Package Builder will check if any updates are available and if so, the **Live Update Wizard** Pic 1 will appear on the screen.




**Pic 1. The Live Update Wizard welcome page**

The **Live Update Wizard** will introduce you to the changes made in the newer version and guide you through the whole updating process while showing the detailed download progress. When the download is finished, the program will be restarted to perform the actual update.

Copyright © 2001-2023 EMCO. All rights reserved.

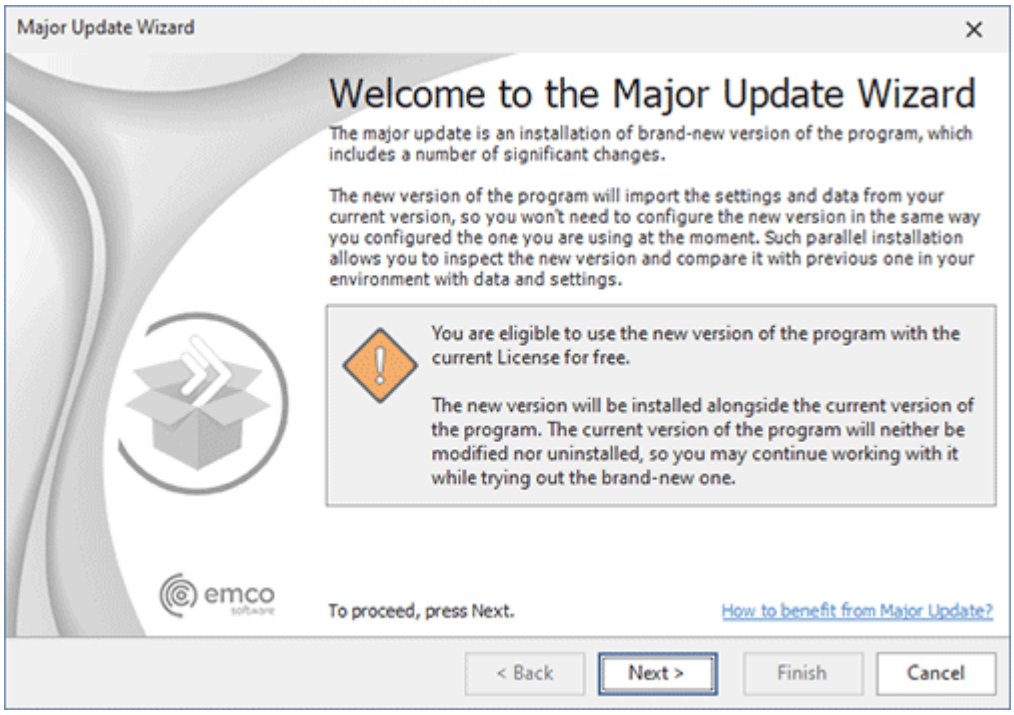
## Major Update

Along with the **Live Update** feature, MSI Package Builder comes with a built-in function of automatic checking for Major Updates. The Major Update is an update to a brand-new version of MSI Package Builder that includes a number of significant changes.

You can install this version alongside the version you are using now. It will import the settings and data from your current version, so that you won't need to configure the new version in the same way you configured the one you are using at the moment. Such parallel installation allows you to inspect the new version and compare it with the previous one in your environment with your data and settings.

The Major Update is installed alongside the version you currently use. The existing version is not automatically uninstalled from your PC, and you can continue using the program version you are accustomed to while having a look at the brand new one.

If the program detects availability of a Major Update, the **Major Update Wizard** Pic 1 will appear on the screen.




**Pic 1. The Major Update Wizard welcome page**

The **Major Update Wizard** will introduce you to the features available in the brand new version of MSI Package Builder and guide you through the update process. The message displayed at the bottom of the welcome page will let you know if the current License allows you to install and use the Major Update for free. When the download is finished, the new version installation will be run automatically.

Copyright © 2001-2023 EMCO. All rights reserved.

# Chapter 14: Main Program Actions

The main program actions are all gathered on the **Ribbon** bar and are grouped by the functions performed into pages. There are static pages, that are always displayed, and categories, containing contextual pages, that are displayed only when a specific context is active. As for the static pages, they are the following: **Home**, **Project**, **View** and **Program**. These static pages are filled up with the **File System**, **Registry**, **Environment Variables**, **Services**, **Assemblies**, **Drivers**, **Printers**, **Windows Firewall**, **Scheduled Tasks**, **Custom Actions** and **Wrapped Packages** pages.

This chapter is to describe the actions available on all the pages and can be used as a glossary while working with MSI Package Builder.

*Copyright © 2001-2023 EMCO. All rights reserved.*

## Home Ribbon Page

The **Home** Ribbon page contains the actions for repackaging, projects management and deployment package creation. This section describes each group from this page and all available actions.

### Builder Ribbon Group

The **Builder** Ribbon group is used to provide access to wizards specially designed to simplify the installation repackaging and projects management workflow.




**Repackage Installation**

The **Repackage Installation** button from the **Builder** group on the **Home** Ribbon page should be used to open the Repackage Installation wizard, that is designed to help you with the repackaging process.



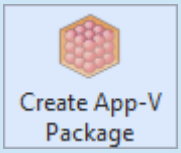

**Project Setup**

The **Project Setup** button from the **Builder** group on the **Home** Ribbon page opens the project setup wizard, that can be used as an entry point both for managing MSI Package Builder projects and creating MSI packages on the basis of those projects.




**Create MSI Package**

The **Create MSI Package** button from the **Builder** group on the **Home** Ribbon page should be used to generate an MSI package based on the selected project.




**Create App-V Package**

The **Create App-V Package** button from the **Builder** group on the **Home** Ribbon page should be used to generate an App-V package based on the selected project.




**Create MSIX/AppX Package**

The **Create MSIX/AppX Package** button from the **Builder** group on the **Home** Ribbon page should be used to generated an MSIX/AppX package based on the selected project.

### Project Management Ribbon Group

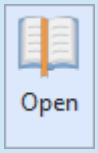
The **Project Management** Ribbon group contains the actions on managing the projects storage and a set of projects available in the **Projects** view.




**New**

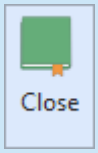
The **New** button from the **Project Management** group on the **Home** Ribbon page should be used to create a new project with a single empty package in it.

Copyright © 2001-2023 EMCO. All rights reserved.




**Open**

The **Open** button from the **Project Management** group on the **Home** Ribbon page allows you to open projects available in the projects storage.



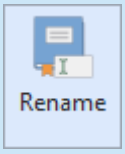

**Close**

The **Close** button from the **Project Management** group on the **Home** Ribbon page should be used to close the selected projects, thus remove them from the **Projects** view.




**Close All**

The **Close All** button from the **Project Management** group on the **Home** ribbon page allows you to close all currently opened projects, thus remove them from the **Projects** view.




**Rename**

The **Rename** button from the **Project Management** group on the **Home** Ribbon page should be used to provide a new name for the selected project.



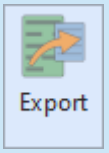

**Delete**

The **Delete** button from the **Project Management** group on the **Home** Ribbon page allows you to delete the selected projects from the projects storage.




**Import**

The **Import** button from the **Project Management** group on the **Home** Ribbon page should be used to create a new MSI Package Builder project based on an existing deployment package or a project archive file.




**Export**

The **Export** button from the **Project Management** group on the **Home** Ribbon page should be used to export all project contents to a compressed file.

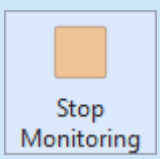
## Capture Ribbon Group

The **Capture** Ribbon group should be used to control the manually initiated advanced monitoring of changes to create a project for further generation of deployment packages based on those changes.




**Start Monitoring**

The **Start Monitoring** button from the **Capture** group on the **Home** Ribbon page should be used to start a new session for tracking changes performed to the system.




**Stop Monitoring**

The **Stop Monitoring** button from the **Capture** group on the **Home** Ribbon page should be used to stop the currently running monitoring session.

Copyright © 2001-2023 EMCO. All rights reserved.




**Manage VM Profiles**

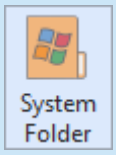
The **Manage VM Profiles** button from the **Capture** group on the **Home** Ribbon page allows managing virtual machines used for installations monitoring.

## Project Ribbon Page

The **Project** Ribbon page contains the actions for creating new project structure units and preparing a project. This section describes each group from this page and all available actions.

### New Ribbon Group

The **New** Ribbon group is used to create new items within a project.



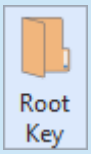

**System Folder**

The **System Folder** button from the **New** group on the **Project** Ribbon page should be used to add a new system folder object to the currently configured project.



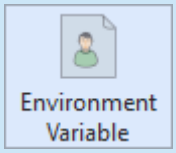

**Drive**

The **Drive** button from the **New** group on the **Project** Ribbon page should be used to add a new logical drive to the currently configured project.



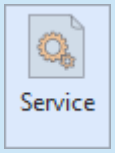

**Root Key**

The **Root Key** button from the **New** group on the **Project** Ribbon page should be used to add a new root key to the currently configured project.




**Environment Variable**

The **Environment Variable** button from the **New** group on the **Project** Ribbon page should be used to add a new environment variable modification to the currently configured project.



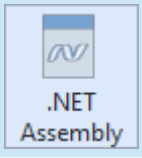

**Service**

The **Service** button from the **New** group on the **Project** Ribbon page should be used to add a new service modification to the currently configured project.




**Win32 Assembly**

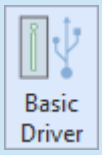
The **Win32 Assembly** button from the **New** group on the **Project** Ribbon page should be used to add a new Win32 assembly to the currently configured project.




**.NET Assembly**

The **.NET Assembly** button from the **New** group on the **Project** Ribbon page should be used to add a new .NET assembly to the currently configured project.

Copyright © 2001-2023 EMCO. All rights reserved.




**Basic Driver**

The **Basic Driver** button from the **New** group on the **Project** Ribbon page should be used to add a new basic driver deployment action to the currently configured project.



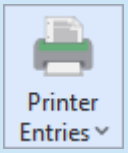

**Driver Package**

The **Driver Package** button from the **New** group on the **Project** Ribbon page should be used to add a new driver package to the currently configured project to be either installed or pre-installed by a resulting deployment package.



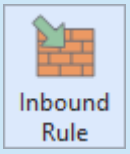

**DIFx Driver Package**

The **DIFx Driver Package** button from the **New** group on the **Project** Ribbon page should be used to add a new Driver Installation Framework package to the currently configured project to be either installed or pre-installed by a resulting deployment package.




**Printer Entries**

The **Printer Entries** drop-down button from the **New** group on the **Project** Ribbon page can be used to create modifications to the printing system to be performed by a resulting deployment package.



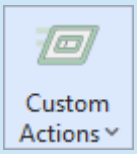

**Inbound Rule**

The **Inbound Rule** button from the **New** group on the **Project** Ribbon page should be used to create a new inbound Windows Firewall rule and add it to the currently configured project.



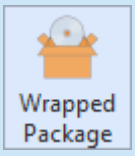

**Outbound Rule**

The **Outbound Rule** button from the **New** group on the **Project** Ribbon page should be used to create a new outbound Windows Firewall rule and add it to the currently configured project.




**Custom Actions**

The **Custom Actions** drop-down button from the **New** group on the **Project** Ribbon page should be used to create a new action of a specific type and add it to the selected project.



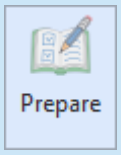

**Wrapped Package**

The **Wrapped Package** button from the **New** group on the **Project** Ribbon page should be used to create a new wrapped package and add it to the selected project.

*Copyright © 2001-2023 EMCO. All rights reserved.*

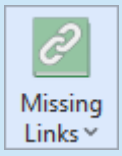
## Links Ribbon Group

The **Links** Ribbon group contains the actions for preparing projects for a deployment package creation.




**Prepare**

The **Prepare** button from the **Links** group on the **Project** Ribbon page should be used to prepare the selected projects.




**Missing Links**

The **Missing Links** drop-down button from the **Links** group on the **Project** Ribbon page allows you to choose the method for resolving the problems with missing files in the selected projects. You can either remove those files, or try to check if they became available.

## Import Ribbon Group

The **Import** Ribbon group should be used to import file system or registry changes into projects.




**File System**

The **File System** drop-down button from the **Import** group on the **Project** Ribbon page should be used to add files and folders from the file system to the selected project.




**Registry**

The **Registry** button from the **Import** group on the **Project** Ribbon page allows you to add the changes defined in the specific registration entries (.reg) file to the selected project.

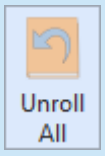
## System Folders Ribbon Group

The **System Folders** Ribbon group contains the actions for rolling and unrolling the system folders for a project.




**Roll All**

The **Roll All** button from the **System Folders** group on the **Project** Ribbon page should be used to replace all system folders with their definitions in the selected projects.




**Unroll All**

The **Unroll All** button from the **System Folders** group on the **Project** Ribbon page allows you to expand the system folder definitions to their absolute local paths in the selected projects.

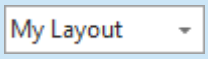
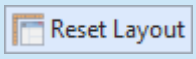
Copyright © 2001-2023 EMCO. All rights reserved.

## View Ribbon Page

The **View** Ribbon page is used to control the program representation, such as the currently applied skin, visible views and their layout.

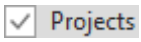
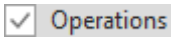
### Layout Ribbon Group

The **Layout** Ribbon group should be used for a workspace layout management.

| | |
|---|---|
|  | **Layout Chooser**<br>The **Layout Chooser** is represented by a drop-down list filled with available window layouts. You can choose one of the existing layouts to apply it or input a name for a new layout to be saved. |
|  | **Reset Workspace**<br>The **Reset Workspace** action should be used to restore the default windows', views' and dock panels' layout. |

### Show Ribbon Group

The **Show** Ribbon group should be used for managing the currently visible MSI Package Builder views.

| | |
|---|---|
|  | **Projects**<br>Shows or hides the **Projects** view. The **Projects** view is designed to display a set of currently opened MSI Package Builder projects. |
|  | **Log**<br>Shows or hides the **Log** view. The **Log** view shows information on the events taking place during the program execution. |
|  | **Operations**<br>Shows or hides the **Operations** view. The **Operations** view shows detailed progress of each operation currently performed and allows canceling some or all running operations. |

### Skins Ribbon Group

The **Skins** Ribbon group provides you with an option of quick changing of the program skins.

| | |
|---|---|
|  | **Skin Chooser**<br>The **Skin Chooser** is a drop-down button that should be used to select the skin from the available skins to be applied to the program. The currently active skin is the checked one. |

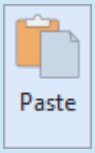
Copyright © 2001-2023 EMCO. All rights reserved.

## Program Ribbon Page

The **Program** Ribbon page contains service actions that are not connected to the business area of the program.

### Clipboard Ribbon Group

The **Clipboard** Ribbon group contains the actions to copy/move objects to the clipboard and paste data from the clipboard.



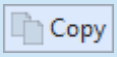

**Paste**

The **Paste** button should be used to add the objects that are currently available from the clipboard to a selected location in the currently focused view.




**Cut**

The **Cut** button should be used to put the selected objects from the currently focused view to the clipboard and remove them from the source after pasting them to the destination.




**Copy**

The **Copy** button allows you to put the selected objects from the currently focused view to the clipboard to be copied to the destination when pasting.

### Update Ribbon Group

The **Update** Ribbon group is responsible for the product updates function. The glyph of this group opens the **General Settings** preference page enabling you to configure the auto-update options.

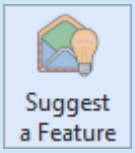



**Check for Updates**

The **Check for Updates** button should be used to check for new versions of MSI Package Builder.

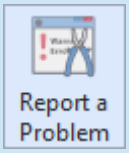
### Feedback Ribbon Group

You can use the **Feedback** Ribbon group to send EMCO Software a request for a new feature you are missing in MSI Package Builder or to report on problems you faced while working with the program.




**Suggest a Feature**

The **Suggest a Feature** button should be used to suggest a functionality you would like to see in the next versions of MSI Package Builder.



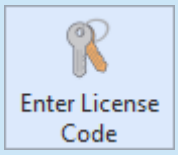

**Report a Problem**

The **Report a Problem** button should be used for reporting the problem you faced while working with MSI Package Builder.

Copyright © 2001-2023 EMCO. All rights reserved.

## Information Ribbon Group

The **Information** Ribbon group has a range of useful actions to manage licensing issues, get help or information you may require.



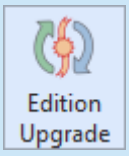

**Enter License Code**

The **Enter License Code** action should be used to enter the license code you received from EMCO to activate MSI Package Builder.




**Buy Now**

The **Buy Now** button is only visible if MSI Package Builder is not activated. It provides you with a quick access to the EMCO web store.




**Edition Upgrade**

The **Edition Upgrade** provides you with an ability of benefiting from update to a more comprehensive edition of MSI Package Builder with a help of the **Edition Upgrade Wizard** that will help you choose an appropriate edition and purchase a license for using it.




**Contact Support**

The **Contact Support** button should be used to get efficient technical assistance from EMCO Software support team.




**EMCO on the Web**

The **EMCO on the Web** button provides you with a short-cut to the EMCO website.




**About**

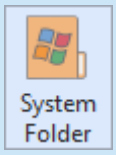
It opens the **About** dialog to review the MSI Package Builder information, licensing information and the End-User License Agreement.

## File System Contextual Ribbon Page

The contextual **File System** Ribbon page is displayed when the **File System** view is active and contains the actions for file system modifications management. This section describes each group from this page and all available actions.

## New Ribbon Group

The **New** Ribbon group is used to create file system modifications within a project.




**System Folder**

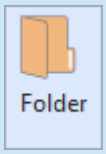
The **System Folder** button from the **New** group on the contextual **File System** Ribbon page should be used to add a new system folder object to the currently configured project.

Copyright © 2001-2023 EMCO. All rights reserved.




**Drive**

The **Drive** button from the **New** group on the contextual **File System** Ribbon page should be used to add a new logical drive to the currently configured project.



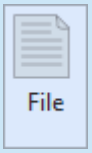

**Folder**

The **Folder** button from the **New** group on the contextual **File System** Ribbon page should be used to create a new folder modification within the selected path.




**Sync Folder**

The **Sync Folder** button from the **New** group on the contextual **File System** Ribbon page should be used to create a new synchronized folder modification within the selected path.




**File**

The **File** button from the **New** group on the contextual **File System** Ribbon page should be used to create a new file modification within the selected path.




**Shortcut**

The **Shortcut** button from the **New** group on the contextual **File System** Ribbon page allows you to create a new shortcut to any target from this project in the selected path.

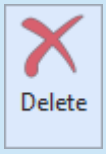
## Management Ribbon Group

The **Management** Ribbon group allows you to edit, delete and transfer file system modifications.



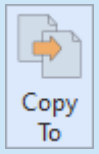

**Edit**

The **Edit** button from the **Management** group on the contextual **File System** Ribbon page should be used to change the selected file system modification.



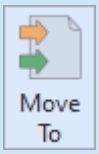

**Delete**

The **Delete** button from the **Management** group on the contextual **File System** Ribbon page allows you to delete the selected file system modification from the project.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **File System** Ribbon page should be used to copy the selected file system modifications to another location.



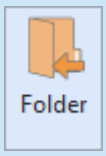

**Move To**

The **Move To** button from the **Management** group on the contextual **File System** Ribbon page allows you to move the selected file system modifications to another location.

Copyright © 2001-2023 EMCO. All rights reserved.

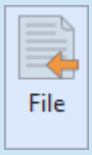
## Import Ribbon Group

The **Import** Ribbon group should be used to import file system changes into a project.




**Folder**

The **Folder** button from the **Import** group on the contextual **File System** Ribbon page allows you to add a folder from the file system with or without its content to the selected project.




**File**

The **File** button from the **Import** group on the contextual **File System** Ribbon page allows you to add a file from the file system to the selected project.

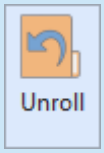
## System Folders Ribbon Group

The **System Folders** Ribbon group contains the actions for rolling and unrolling specific system folders within the file system modifications.




**Roll**

The **Roll** button from the **System Folders** group on the contextual **File System** Ribbon page should be used to replace the selected system folders with their definitions in the selected project.

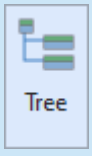



**Unroll**

The **Unroll** button from the **System Folders** group on the contextual **File System** Ribbon page allows you to expand the selected system folders to their absolute local paths in the selected project.

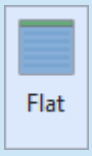
## Presentation Ribbon Group

The **Presentation** Ribbon group should be used to switch between tree and flat presentation of file system modifications.




**Tree**

The **Tree** button from the **Presentation** group of the contextual **File System** tab should be used to display the file system modifications in form of a tree, similar to Windows Explorer.




**Flat**

The **Flat** button from the **Presentation** group of the contextual **File System** tab allows you to display the file system modifications as a raw list, where each element is represented with its full path.

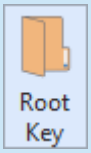
Copyright © 2001-2023 EMCO. All rights reserved.

## Registry Contextual Ribbon Page

The contextual **Registry** Ribbon page is displayed when the **Registry** view is active and contains the actions for registry modifications management. This section describes each group from this page and all available actions.

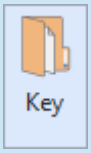
### New Ribbon Group

The **New** Ribbon group is used to create registry modifications within a project.



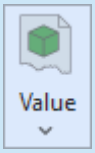

**Root Key**

The **Root Key** button from the **New** group on the contextual **Registry** Ribbon page should be used to add a new root key to the currently configured project.




**Key**

The **Key** button from the **New** group on the contextual **Registry** Ribbon page should be used to create a new registry key modification within the selected registry key.




**Value**

The **Value** drop-down button from the **New** group on the contextual **Registry** Ribbon page should be used to create a new registry value modification within the selected registry key.

### Management Ribbon Group

The **Management** Ribbon group allows you to edit, delete and transfer registry modifications.



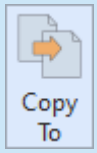

**Edit**

The **Edit** button from the **Management** group on the contextual **Registry** Ribbon page should be used to change the selected registry modification.



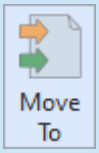

**Delete**

The **Delete** button from the **Management** group on the contextual **Registry** Ribbon page allows you to delete the selected registry modification from the selected project.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **Registry** Ribbon page should be used to copy the selected registry modifications to another location.




**Move To**

The **Move To** button from the **Management** group on the contextual **Registry** Ribbon page allows you to move the selected registry modifications to another location.

Copyright © 2001-2023 EMCO. All rights reserved.

## Import Ribbon Group

The **Import** Ribbon group should be used to import registry changes into project.

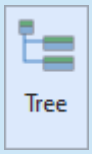



**Registry**

The **Registry** button from the **Import** group on the contextual **Registry** Ribbon page allows you to add the changes defined in the specific registration entries (.reg) file to the selected project.

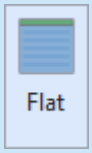
## Presentation Ribbon Group

The **Presentation** Ribbon group should be used to switch between tree and flat presentation of registry modifications.




**Tree**

The **Tree** button from the **Presentation** group of the contextual **Registry** tab should be used to display the registry modifications in form of a tree, similar to **Registry Editor**.




**Flat**

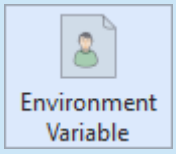
The **Flat** button from the **Presentation** group of the contextual **Registry** tab allows you to display the registry modifications as a raw list, where each element is represented with its full path.

## Environment Variables Contextual Ribbon Page

The contextual **Environment Variables** Ribbon page is displayed when the **Environment Variables** view is active and contains the actions for environment variables modifications management. This section describes each group from this page and all available actions.

## New Ribbon Group

The **New** Ribbon group is used to create environment variables modifications within a project.




**Environment Variable**

The **Environment Variable** button from the **New** group on the contextual **Environment Variables** Ribbon page should be used to add a new environment variable modification to the currently configured project.

Copyright © 2001-2023 EMCO. All rights reserved.

## Management Ribbon Group

The **Management** Ribbon group allows you to edit, delete and transfer environment variables modifications.



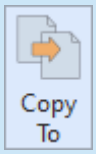

**Edit**

The **Edit** button from the **Management** group on the contextual **Environment Variables** Ribbon page should be used to change the selected environment variables modification.



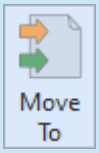

**Delete**

The **Delete** button from the **Management** group on the contextual **Environment Variables** Ribbon page allows you to delete the selected environment variables modifications from the selected project.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **Environment Variables** Ribbon page should be used to copy the selected environment variables modifications to another location.




**Move To**

The **Move To** button from the **Management** group on the contextual **Environment Variables** Ribbon page allows you to move the selected environment variables modifications to another location.

## Services Contextual Ribbon Page

The contextual **Services** Ribbon page is displayed when the **Services** view is active and contains the actions for services modifications management. This section describes each group from this page and all available actions.

## New Ribbon Group

The **New** Ribbon group is used to create services modifications within a project.




**Service**

The **Service** button from the **New** group on the contextual **Services** Ribbon page should be used to add a new service modification to the currently configured project.

Copyright © 2001-2023 EMCO. All rights reserved.

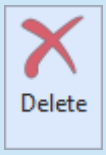
## Management Ribbon Group

The **Management** Ribbon group allows you to edit, delete and transfer services modifications.



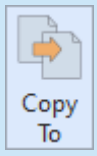

**Edit**

The **Edit** button from the Management group on the contextual Services Ribbon page should be used to change the selected services modification.



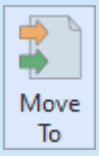

**Delete**

The **Delete** button from the **Management** group on the contextual **Services** Ribbon page allows you to delete the selected services modifications from the selected project.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **Services** Ribbon page should be used to copy the selected services modifications to another location.




**Move To**

The **Move To** button from the **Management** group on the contextual **Services** Ribbon page allows you to move the selected services modifications to another location.

## Assemblies Contextual Ribbon Page

The contextual **Assemblies** Ribbon page is displayed when the **Assemblies** view is active and contains actions for side-by-side assemblies management. This section describes each group from this page and all available actions.

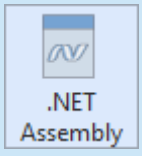
## New Ribbon Group

The **New** Ribbon group is used to create side-by-side assemblies to be included into a project.




**Win32 Assembly**

The **Win32 Assembly** button from the **New** group on the contextual **Assemblies** Ribbon page should be used to add a new Win32 assembly to the currently configured project.



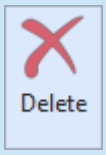
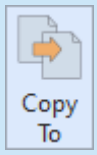
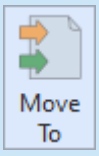

**.NET Assembly**

The **.NET Assembly** button from the **New** group on the contextual **Assemblies** Ribbon page should be used to add a new .NET assembly to the currently configured project.

Copyright © 2001-2023 EMCO. All rights reserved.

## Management Ribbon Group

The **Management** Ribbon group allows you to edit, delete and transfer side-by-side assemlbies.

| | |
|---|---|
|   | **Edit**<br>The **Edit** button from the **Management** group on the contextual **Assemblies** Ribbon page should be used to change the selected assembly. |
|   | **Delete**<br>The **Delete** button from the **Management** group on the contextual **Assemblies** Ribbon page allows you to delete the selected assemblies from the selected project. |
|   | **Copy To**<br>The **Copy To** button from the **Management** group on the contextual **Assemblies** Ribbon page should be used to copy the selected assemblies to another project. |
|   | **Move To**<br>The **Move To** button from the **Management** group on the contextual **Assemblies** Ribbon page allows you to move the selected assemblies to another project. |

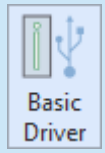
Copyright © 2001-2023 EMCO. All rights reserved.

## Drivers Contextual Ribbon Page

The contextual **Drivers** Ribbon page is displayed when the **Drivers** view is active and contains actions for driver modifications management. This section describes each group from this page and all available actions.

### New Ribbon Group

The **New** Ribbon group is used to create a new basic driver or a driver package deployment action within a project.




**Basic Driver**

The **Basic Driver** button from the **New** group on the contextual **Drivers** Ribbon page should be used to add a new basic driver deployment action to the currently configured project.




**Driver Package**

The **Driver Package** button from the **New** group on the contextual **Drivers** Ribbon page should be used to add a new driver package to the currently configured project to be either installed or pre-installed by a resulting deployment package.




**DIFx Driver Package**

The **DIFx Driver** Package button from the **New** group on the **Drivers** contextual Ribbon page should be used to add a new Driver Installation Framework package to the currently configured project to be either installed or pre-installed by a resulting deployment package.

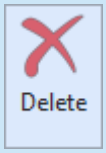
### Management Ribbon Group

The **Management** Ribbon group allows you to edit, delete and transfer driver modifications.



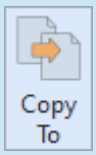

**Edit**

The **Edit** button from the **Management** group on the contextual **Drivers** Ribbon page should be used to change the selected driver deployment action.




**Delete**

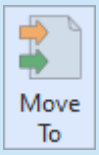
The **Delete** button from the **Management** group on the contextual **Drivers** Ribbon page allows you to delete the selected driver deployment actions from the selected project.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **Drivers** Ribbon page should be used to copy the selected driver deployment actions to another location.

Copyright © 2001-2023 EMCO. All rights reserved.




**Move To**

The **Move To** button from the **Management** group on the contextual **Drivers** Ribbon page allows you to move the selected driver deployment actions to another location.

## Order Ribbon Group

The **Order** Ribbon group is used to organize the driver deployment order.




**Move Up**

The **Move Up** button from the **Order** group on the contextual **Drivers** ribbon page should be used to move the selected drivers up the deployment order.




**Move Down**

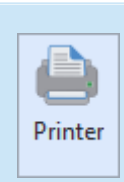
The **Move Down** button from the **Order** group on the contextual **Drivers** ribbon page should be used to move the selected drivers down the deployment order.

## Printers Contextual Ribbon Page

The **Printers** contextual Ribbon page is displayed when the **Printers** view is active. It contains the actions for managing the printing system configuration. This section describes each group from this page and all the available actions.

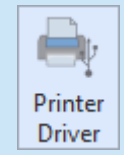
## New Ribbon Group

The **New** Ribbon group is used to create a new entry for configuring the printing system.



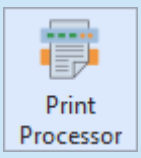

**Printer**

The **Printer** button from the **New** group on the **Printers** contextual Ribbon page should be used to add a new deployment action for creating, modifying or deleting a printer during the package deployment.




**Printer Driver**

The **Printer Driver** button from the **New** group on the **Printers** contextual Ribbon page should be used to add a new deployment action for installing or removing a printer driver during the package deployment.




**Print Processor**

The **Print Processor** button from the **New** group on the **Printers** contextual Ribbon page should be used to add a new deployment action for adding or removing a print processor during the package deployment.

Copyright © 2001-2023 EMCO. All rights reserved.




**Print Monitor**

The **Print Monitor** button from the **New** group on the **Printers** contextual Ribbon page should be used to add a new deployment action for adding or removing a print monitor (either a language monitor or a port monitor) during the package deployment.




**Printer Port**

The **Printer Port** button from the **New** group on the **Printers** contextual Ribbon page should be used to add a new deployment action for creating, modifying or deleting a printer port during the package deployment.

## Management Ribbon Group

The **Management** Ribbon group allows you to edit, delete and transfer printing system configuration entries.



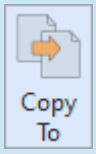

**Edit**

The **Edit** button from the **Management** group on the **Printers** contextual Ribbon page should be used to change the selected printing system configuration action.



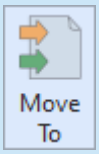

**Delete**

The **Delete** button from the **Management** group on the **Printers** contextual Ribbon page allows you to delete the selected printing system configuration actions from the selected project.




**Copy To**

The **Copy To** button from the **Management** group on the **Printers** contextual Ribbon page should be used to copy the selected printing system configuration actions to another location.




**Move To**

The **Move To** button from the **Management** group on the **Printers** contextual Ribbon page allows you to move the selected printing system configuration actions to another location.

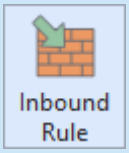
Copyright © 2001-2023 EMCO. All rights reserved.

## Windows Firewall Contextual Ribbon Page

The contextual **Windows Firewall** Ribbon page is displayed when the **Windows Firewall** view is active and contains the items that used for firewall rules management. This section describes each group from this page and all available actions.

### New Ribbon Group

The **New** Ribbon group is used to create a new action for a specific project.




**Inbound Rule**

The **Inbound Rule** button from the **New** group on the contextual **Windows Firewall** Ribbon page should be used to create a new inbound firewall rule and add it to the selected project.




**Outbound Rule**

The **Outbound Rule** button from the **New** group on the contextual **Windows Firewall** Ribbon page should be used to create a new outbound firewall rule and add it to the selected project.

### Management Ribbon Group

The **Management** Ribbon group allows you to edit, delete and transfer custom actions.



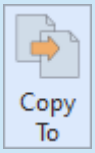

**Edit**

The **Edit** button from the **Management** group on the contextual **Windows Firewall** Ribbon page should be used to change the selected firewall rule configuration.



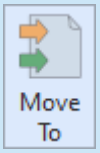

**Delete**

The **Delete** button from the **Management** group on the contextual **Windows Firewall** Ribbon page allows you to delete the selected firewall rules from the selected project.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **Windows Firewall** Ribbon page should be used to copy the selected firewall rules to another project.




**Move To**

The **Move To** button from the **Management** group on the contextual **Windows Firewall** Ribbon page allows you to move the selected firewall rules to another project.

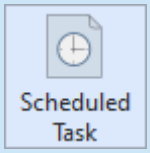
Copyright © 2001-2023 EMCO. All rights reserved.

## Scheduled Tasks Contextual Ribbon Page

The contextual **Scheduled Tasks** Ribbon page is displayed when the **Scheduled Tasks** view is active and contains the actions for scheduled tasks modifications management. This section describes each group from this page and all available actions.

### New Ribbon Group

The **New** Ribbon group is used to create scheduled tasks modifications within a project.




**Scheduled Task**

The **Scheduled Task** button from the **New** group on the contextual **Scheduled Tasks** Ribbon page should be used to add a new scheduled task modification to the currently configured project.

### Management Ribbon Group

The **Management** Ribbon group allows you to edit, delete and transfer scheduled tasks modifications.



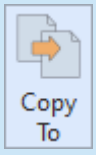

**Edit**

The **Edit** button from the **Management** group on the contextual **Scheduled Tasks** Ribbon page should be used to change the selected scheduled tasks modification.



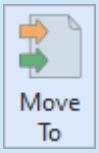

**Delete**

The **Delete** button from the **Management** group on the contextual **Scheduled Tasks** Ribbon page allows you to delete the selected scheduled tasks modifications from the selected project.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **Scheduled Tasks** Ribbon page should be used to copy the selected scheduled tasks modifications to another location.




**Move To**

The **Move To** button from the **Management** group on the contextual **Scheduled Tasks** Ribbon page allows you to move the selected scheduled tasks modifications to another location.

Copyright © 2001-2023 EMCO. All rights reserved.

## Import Ribbon Group

The **Import** Ribbon group is used to import scheduled tasks to a project.




**Scheduled Task**

The **Scheduled Task** button from the **Import** group on the contextual **Scheduled Tasks** Ribbon page should be used to import a scheduled task configuration from an XML file to the currently configured project.

## Custom Actions Contextual Ribbon Page

The contextual **Custom Actions** Ribbon page is displayed when the **Custom Actions** view is active and contains the items that used for custom actions management. This section describes each group from this page and all available actions.

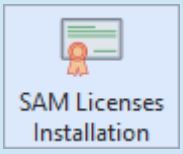
## New Ribbon Group

The **New** Ribbon group is used to create a new action for a specific project.




**Pre & Post Action**

The **Pre & Post Action** button from the New group on the contextual **Custom Actions** Ribbon page should be used to create a new generic action and add it to the selected project.




**SAM Licenses Installation**

The **SAM Licenses Installation** button from the New group on the **Custom Actions** contextual Ribbon page should be used to create a new action for installing So**ftware Assets Management (SAM)** licenses to the **Software Licensing Service (SLS)** and to add it to the selected project.




**Pin Application Action**

The **Pin Application Action** button from the **New** group on the **Custom Actions** contextual Ribbon page allows you to create a new action to pin any application to or unpin it from the **Start Menu** and/or the **Task Bar**.

## Management Ribbon Group

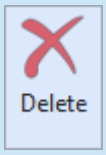
The **Management** Ribbon group allows you to edit, delete and transfer custom actions.




**Edit**

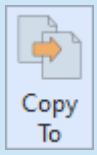
The **Edit** button from the **Management** group on the contextual **Custom Actions** Ribbon page should be used to change the selected action configuration.

Copyright © 2001-2023 EMCO. All rights reserved.



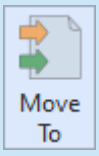

**Delete**

The **Delete** button from the **Management** group on the contextual **Custom Actions** Ribbon page allows you to delete the selected actions from the selected project.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **Custom Actions** Ribbon page should be used to copy the selected actions to another project.

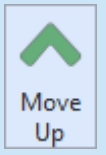



**Move To**

The **Move To** button from the **Management** group on the contextual **Custom Actions** Ribbon page allows you to move the selected actions to another project.

## Order Ribbon Group

The **Order** Ribbon group is used to organize the custom actions execution order.




**Move Up**

The **Move Up** button from the **Order** group on the contextual **Custom Actions** Ribbon page should be used to move the selected user actions up the execution order.

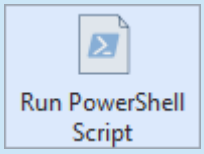



**Move Down**

The **Move Down** button from the **Order** group on the contextual **Custom Actions** Ribbon page should be used to move the selected user actions down the execution order.

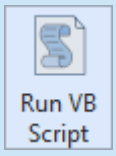
## Action Templates

The **Action Templates** group allows you to create typical custom actions based on preconfigured templates.




**Run Power Shell Script**

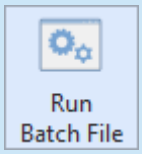
The **Run PowerShell Script** button from the **Action Templates** group allows you to create a new custom action that runs a selected PowerShell script. The program allows you to select a script that is added to a set of action files. Command-line parameters to run the script are set automatically.




**Run VB Script**

The **Run VB Script** button from the **Action Templates** group allows you to create a new custom action that runs a selected VB script. The program allows you to select a script that is added to a set of action files. Command-line parameters to run the script are set automatically.

Copyright © 2001-2023 EMCO. All rights reserved.




**Run Batch File**

The **Run Batch File** button from the **Action Templates** group allows you to create a new custom action that runs a selected batch file. The program allows you to select a batch file that is added to a set of action files. Command-line parameters to run the batch file are set automatically.




**Run Installed Application**

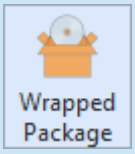
The **Run Installed Application** button from the **Action Templates** group allows you to create a new custom action that runs an application deployed by the package. The program allows you to select an executable file from the project to run it as a custom action. Command-line parameters to run the batch file are set automatically.

## Wrapped Packages Contextual Ribbon Page

The contextual **Wrapped Packages** Ribbon page is displayed when the **Wrapped Packages** view is active and contains the actions for wrapped packages management. This section describes each group from this page and all available actions.

## New Ribbon Group

The **New** Ribbon group is used to create wrapped packages to be included into a project.




**Wrapped Package**

The **Wrapped Package** button from the **New** group on the contextual **Wrapped Packages** Ribbon page should be used to create a new wrapped package and add it to the selected project.

## Management Ribbon Group

The **Management** Ribbon group allows you to edit, delete and transfer wrapped packages.



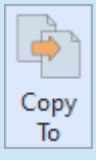

**Edit**

The **Edit** button from the **Management** group on the contextual **Wrapped Packages** Ribbon page should be used to change the selected wrapped package configuration.




**Delete**

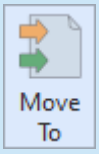
The **Delete** button from the **Management** group on the contextual **Wrapped Packages** Ribbon page allows you to delete the selected wrapped packages from the selected project.




**Copy To**

The **Copy To** button from the **Management** group on the contextual **Wrapped Packages** Ribbon page should be used to copy the selected wrapped packages to another project.

Copyright © 2001-2023 EMCO. All rights reserved.




**Move To**

The **Move To** button from the **Management** group on the contextual **Wrapped Packages** Ribbon page allows you to move the selected wrapped packages to another project.

## Order Ribbon Group

The **Order** Ribbon group is used to organize the wrapped packages deployment order.




**Move Up**

The **Move Up** button from the **Order** group on the contextual **Wrapped Packages** ribbon page should be used to move the selected packages up the deployment order.




**Move Down**

The **Move Down** button from the **Order** group on the contextual **Wrapped Packages** ribbon page should be used to move the selected packages down the deployment order.

Copyright © 2001-2023 EMCO. All rights reserved.

# Chapter 15: Requirements

Please carefully read and follow all requirements, listed here, or you may not be able to successfully use the product. You can contact our support if you experience a problem during the product use.

## System Requirements

Computer running MSI Package Builder must meet the following requirements:

### Minimum Hardware Requirements

- Intel Core iX Processor or equivalent
- 4 GB of RAM
- 2 GB of free disk space

### Recommended Hardware Requirements

- 6th Gen Intel Core iX Processor or equivalent
- 8 GB of RAM
- 10 GB of free disk space

### Supported Platforms (x86/x64)

- Windows 11, 10, 8.1, 8, 7 (with SP1 or later)
- Windows Server 2022, 2019, 2016, 2012 R2, 2012, 2008 R2 (with SP1 or later)

### Requirements

- Administrative rights on the local computer
- Microsoft .NET Framework 3.5 SP1 or above

*Copyright © 2001-2023 EMCO. All rights reserved.*

# Chapter 16: Edition Upgrade

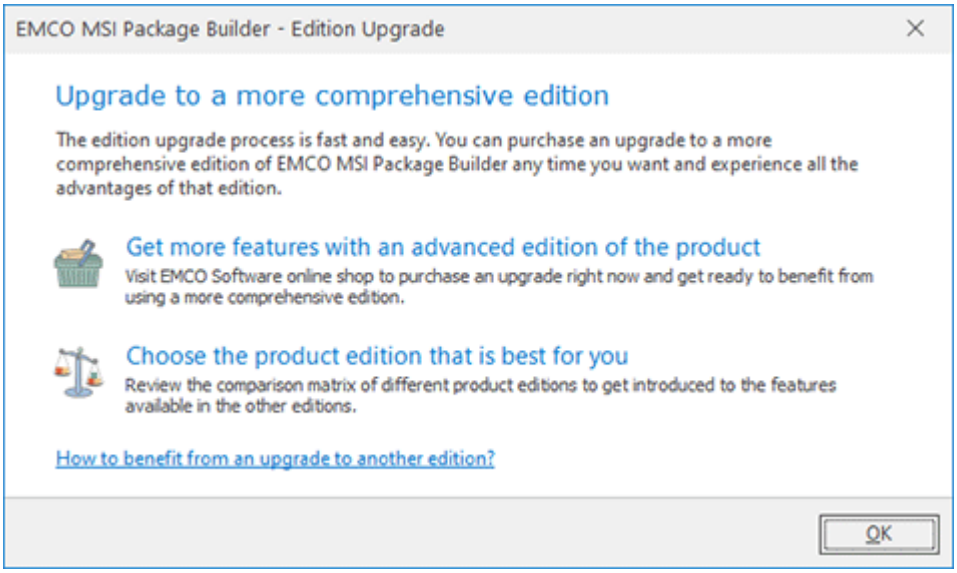
The program has multiple editions with different features, so you can select the one that suits your needs.




**Edition Upgrade**

The **Edition Upgrade** provides you with an ability of benefiting from update to a more comprehensive edition of MSI Package Builder with a help of the **Edition Upgrade Wizard** that will help you choose an appropriate edition and purchase a license for using it.

The **Edition Upgrade Wizard** Pic 1 was designed to make the upgrade process easier. This wizard can be reached by clicking an appropriate hyper link in the **About** dialog or by using the **Edition Upgrade** button from the **Program** Ribbon page.



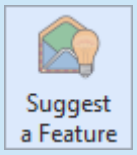

**Pic 1. The Edition Upgrade Wizard**

With a help of the **Edition Upgrade Wizard**, you can purchase an upgrade to more comprehensive edition of MSI Package Builder with a single click on the **Get more features with an advanced edition of the program** option or introduce yourself to the features available in the other edition of the program using the **Choose the program edition that is best for you** option. This option will open a feature list web page that shows you the detailed comparison matrix of the features available in different MSI Package Builder editions so that you can review all the features of each edition before choosing the one that best fits your needs.

Copyright © 2001-2023 EMCO. All rights reserved.

# Chapter 17: How can I leave my Feedback?

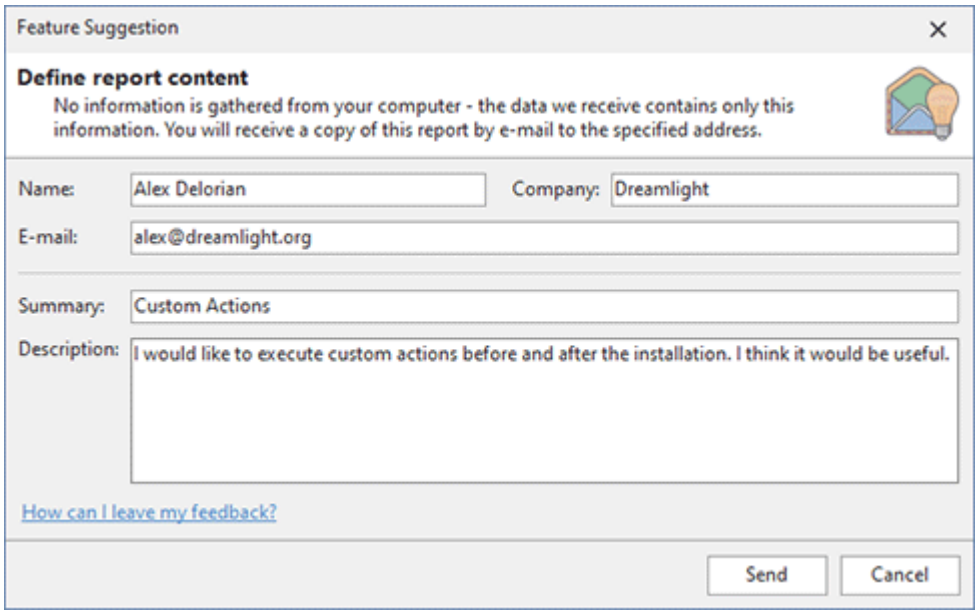
EMCO Software always takes care of its customers, and your opinion means a lot to us. For this reason, our programs have built-in features for your feedback. You can suggest a feature you want to see in new program versions or report a technical problem you have faced using the program. Specifying your contact information on the feedback forms ensures that you will be informed of any changes with regard to the reported issue, our plans for implementing the suggested feature or fixing the reported bug. Those actions can be found in the **Feedback** Ribbon group of the Program page.




**Suggest a Feature**

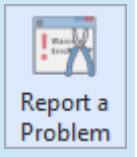
The **Suggest a Feature** button from the **Feedback** Ribbon group should be used to suggest a functionality you would like to see in the next versions of MSI Package Builder.

MSI Package Builder comes with a wide range of features, but if you feel some functionality is missing, you can always suggest a new feature to us that you want to see in the program. To suggest a new feature, you should press the **Suggest a Feature** button from the **Feedback** Ribbon group. After pressing this button, you will see the **Feature Suggestion** dialog Pic 1 on the screen where you are offered to enter your contact information and describe your suggestion.




**Pic 1. The feature suggestion form**

Press **Send** when you are done with filling out the form to send your suggestion.

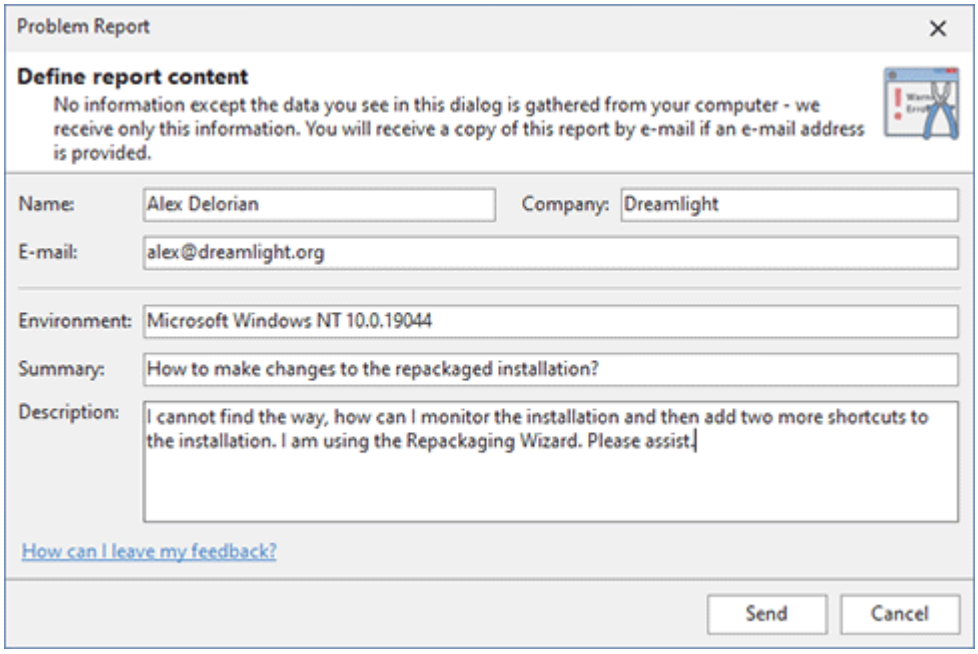



**Report a Problem**

The **Report a Problem** button should be used to report a problem you have faced while working with MSI Package Builder.

Copyright © 2001-2023 EMCO. All rights reserved.

MSI Package Builder is easy to use and very stable. Nevertheless, if you have faced any difficulty or problem while working with it, you can send us a problem report. To send such a report, you should press the **Report a Problem** button from the **Feedback** Ribbon group. When this button is pressed, the **Problem Report** dialog Pic 2 will appear on the screen where you are offered to enter your contact information and describe your problem.




**Pic 2. The problem report form**

In the **Environment** field, you can provide us with a description of the specific environment used while working with the program. Press **Send** when you are done with filling out the form to send your report.

Do not hesitate to contact EMCO Software - we are always glad to receive your feedback and are doing our best to satisfy our customers' preferences.

*Copyright © 2001-2023 EMCO. All rights reserved.*

# Chapter 18: About EMCO Software

EMCO creates mission-critical software to manage network computers remotely and automate network administration.

Our company was founded in 2001 in Reykjavik, Iceland. Managing Windows networks as network administrators, we could not find tools that would help us automate our routine network administration tasks, so we decided to create such tools for fellow administrators and ourselves.

Today we offer innovative software that help IT specialists and network administrators to automate their Windows network management tasks. Our software tools are focused on remote management of Windows computers across networks and allow administrators to perform routine tasks on all managed computers with a few mouse clicks. We automate software audit and deployment, power management, hosts monitoring and other computer administration tasks.

Learn more: http://emcosoftware.com.

## Our Customers

Being suitable for managing networks of any size, our products cater to the needs of 25,000+ customers in 85 countries around the globe. They are Fortune 100 corporations and small businesses, as well as governments, military institutions, universities, public schools, libraries and charities.

Our customers rely on EMCO products for managing their mission-critical network infrastructure. Using our products, network administrators monitor, audit, deploy and manage 3,000,000+ network devices every day.

*Copyright © 2001-2023 EMCO. All rights reserved.*

# Chapter 19: Contact Information

We would be glad to help you with any questions and problems you might have. Use the contact information below.

| Contact Sales | Contact Support |
| --- | --- |
| Our sales team is standing by to answer your questions about purchasing or licensing EMCO products. Submit a request, send us an e-mail or call us: Contact Sales. | Our support team is here to help you with any technical product-related issues you may have. We provide free technical support for all our products, including freeware. Submit a request or send us an e-mail: Contact Support. |

Copyright © 2001-2023 EMCO. All rights reserved.