



# RANOREX

STUDIO SYSTEM DETAILS  
USERGUIDE

## TABLE OF CONTENTS

<b>RANOREX STUDIO SYSTEM DETAILS.....</b>	<b>3</b>
LICENSING .....	4
<i>Install a node-locked license.....</i>	<i>4</i>
<i>Install a floating license.....</i>	<i>9</i>
<i>Uninstall a license .....</i>	<i>17</i>
<i>Transfer a license .....</i>	<i>20</i>
SETTINGS & CONFIGURATION .....	24
<i>Ranorex Recorder settings .....</i>	<i>28</i>
<i>Repository settings.....</i>	<i>35</i>
<i>Report settings .....</i>	<i>37</i>
<i>Image-based automation settings .....</i>	<i>43</i>
<i>Advanced settings &amp; configurations .....</i>	<i>46</i>
<i>Self-healing.....</i>	<i>51</i>
<i>Plugin-specific settings.....</i>	<i>55</i>
SYSTEM REQUIREMENTS .....	66
64-BIT PLATFORMS .....	76
SILENT INSTALLATION.....	78

# Ranorex Studio system details

The chapter contains all information about tools, application methods and basic knowledge (concepts) referring to Ranorex system details, i.e. installation of Ranorex, system requirements, version release notes and licensing including not only the different types of licensing but also the possibilities to transfer licenses from one machine to another.



Licensing



Settings and Configuration



System Requirements



64-bit platforms



Silent Installation

# Licensing

In this and the following chapters, we'll explain how licensing works in Ranorex Studio. You'll find out which licenses there are, how to install them, how to uninstall them, and how to transfer them.

## Licenses

There are two basic kinds of Ranorex Studio licenses: node-locked and floating.

### Node-locked license

Node-locked licenses are bound to a single machine (host), on which they can be used without expiring.

### Floating license

Floating licenses are not bound to a single machine. They “float” on a server, waiting to be leased by a user. When a license is leased, it becomes unavailable to other users until it is returned. Then the license floats on the server again, ready for lease by another user.

Floating licenses are managed in the Ranorex License Manager on a machine. The machine that the License Manager is installed on is called the server in the following chapters. The License Manager distributes floating licenses to users automatically as needed.



#### Note

Both kinds of license can be → [transferred](#) from one machine to another once every 90 days from the last installation.

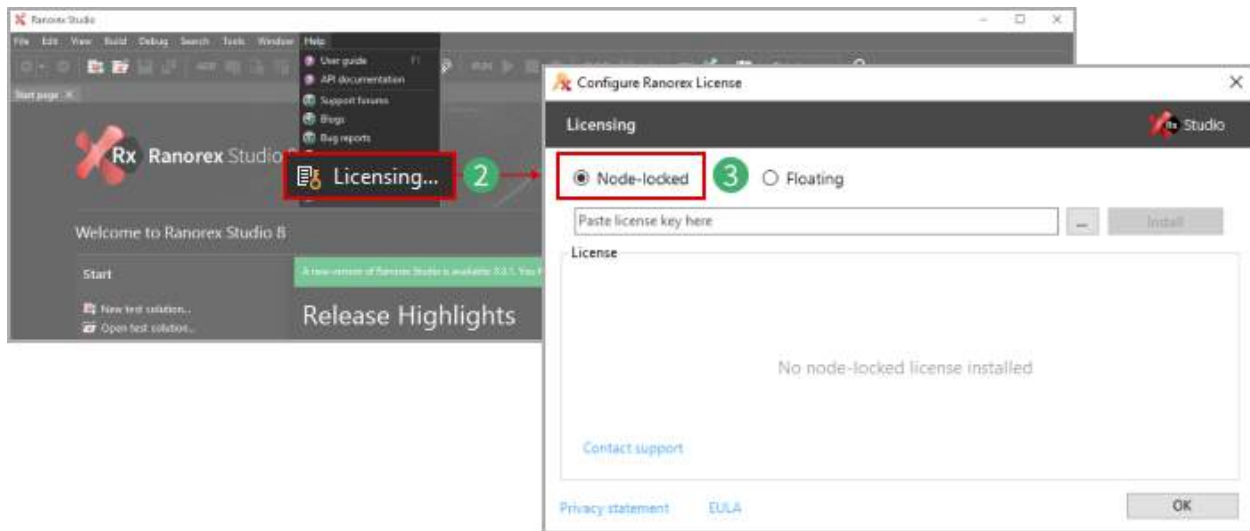
## Install a node-locked license

In this chapter, we'll show you how to install a node-locked license step by step. We'll cover both online and offline installation.

### Open license configuration

First, open the license configuration dialog.

- 1 **Start** Ranorex Studio as an administrator.
- 2 Go to **Help > Licensing...**. The configuration dialog opens.
- 3 **Click Node-locked**.



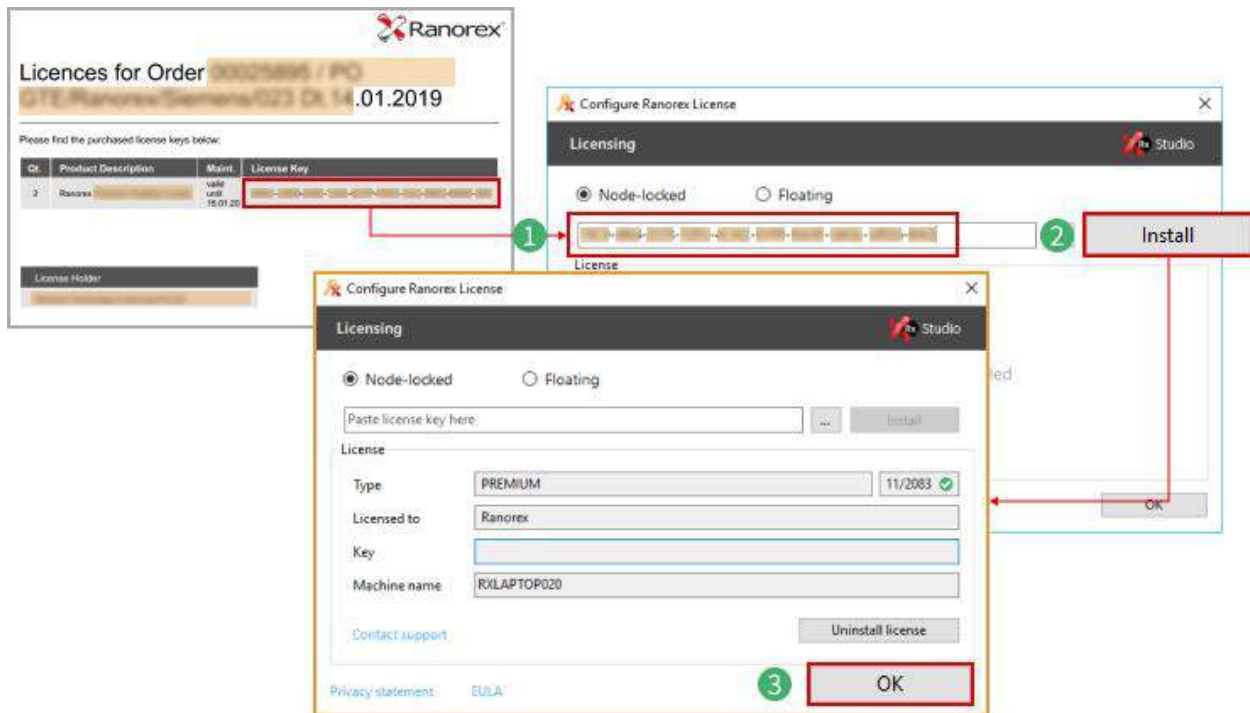
## Install with internet connection

### Requirements

- Active internet connection (to connect to the Ranorex License Authentication Server)
- Make sure the license you want to install isn't already installed on a different machine. If you want to transfer a license, → [reach this chapter](#).

To install the license:

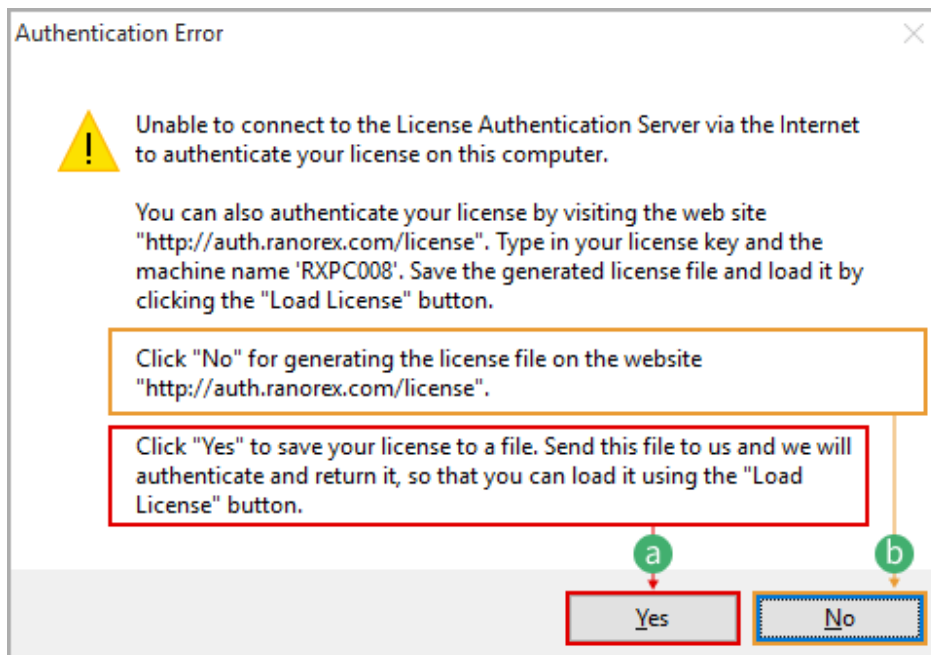
- 1 In the license configuration dialog, **paste** your license key that you received by email.
- 2 **Click Install**.
- 3 **Click OK**.



Ranorex Studio automatically registers the license with the Ranorex License Authentication Server. This completes the installation.

### Authentication error

If the machine you want to install the license on can't connect to the Ranorex License Authentication Server, the following error message will appear:



Do one of the following:

- a **Click Yes.**
  - 1 **Save** the license file.
  - 2 **Send** the file to [support@ranorex.com](mailto:support@ranorex.com) with the email subject **License file authentication**.
  - 3 We will authenticate the file and send it back to you.
  - 4 **Proceed** with **step 6** under **Install without internet connection**.
- b **Click No** and **proceed** with **step 1** under **Install without internet connection**.

## Install without internet connection

While we recommend installation with an active internet connection, you can also install a node-locked license on a machine that can't connect to the internet and, therefore, to the Ranorex License Authentication Server.

### Requirements

- A different machine WITH an internet connection
- A way to transfer a file from the machine with internet to the one without, e.g. a USB stick.
- Make sure the license you want to install isn't already installed on a different machine. If you want to transfer a license, read this chapter.

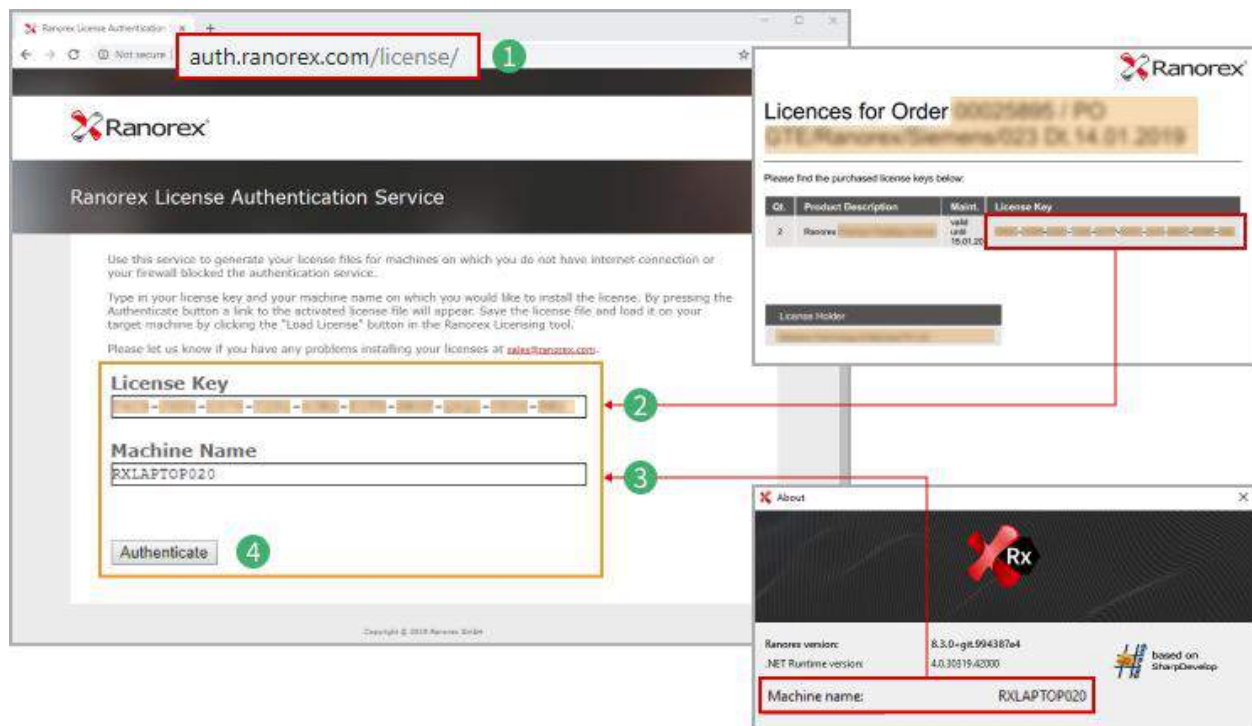
To install the license:

- 1 On the online machine, go to <http://auth.ranorex.com/license/>.
- 2 Under License Key, enter the license key you received by email.
- 3 Under Machine Name, enter the name of the machine you want to install the license on, i.e. the offline machine.

#### **Hint**

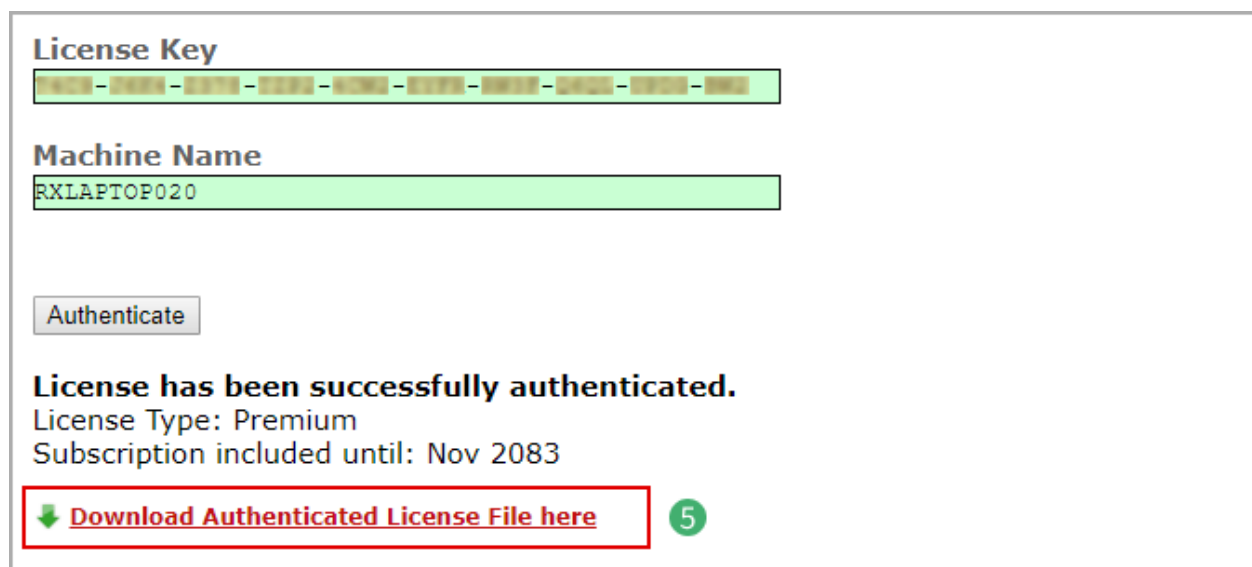
To find out the machine name, go to **Help > About** in Ranorex Studio.

- 4 **Click Authenticate.**



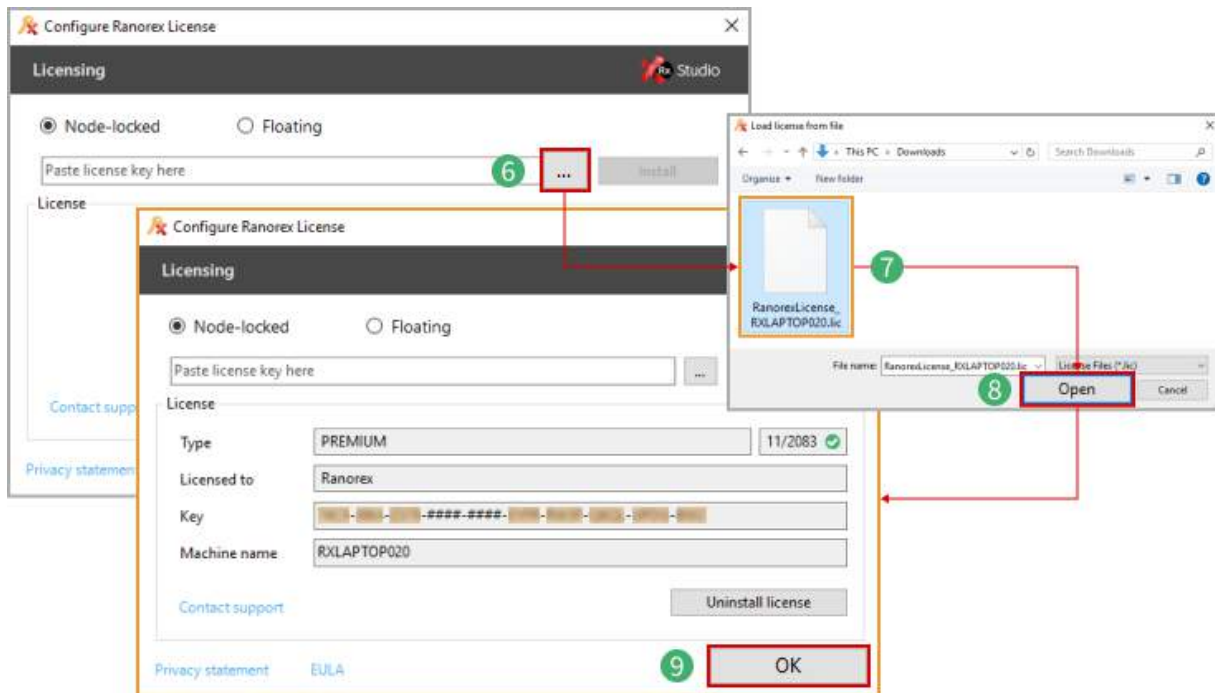
The Ranorex License Authentication Server creates an authenticated license file based on your data. You need to download this file and transfer it to the offline machine.

- 5 **Click Download Authenticated License File** and transfer the downloaded file to the offline machine, e.g. using a USB stick.





- 6 **Open** the license configuration dialog, **select Node-locked**, and **click** the breadcrumbs.
- 7 **Browse** to the authenticated license file.
- 8 **Click Open.**
- 9 **Click OK.**



## Install a floating license

In this chapter, you'll find out how to install floating licenses in the Ranorex License Manager on server machines with or without an internet connection. You'll also find out how to install the Ranorex License Manager, which manages floating licenses, about the different types of floating licenses, and how to connect client machines to the server machine so they can lease floating licenses.

### Ranorex License Manager

Floating licenses are installed and managed using the Ranorex License Manager. The machine the Ranorex License Manager is installed on acts as a license server. It automatically distributes and revokes floating license to and from client machines. This is why you need to install the License Manager on a machine that can be reached through the network by all machines you want to use a floating license on.

In the Licensing chapter, we call the server machine “the server” and the client machines “the client(s)”.

## Install the Ranorex License Manager

Normally, the Ranorex License Manager is installed automatically when you install Ranorex Studio on a machine. However, if you don’t want to install Ranorex Studio on the server, you can also install only the License Manager.

When you buy a floating license, you’ll receive an email with instructions. It also contains a download link for the License Manager.

To install only the License Manager:

- 1 From the link in the email for your floating license, **download** the License Manager to the server.
- 2 **Install** the License Manager following the instructions in the installation wizard.



### Note

C++ libraries will be installed if not already present on the server and the server may have to be restarted during the installation.



### Note

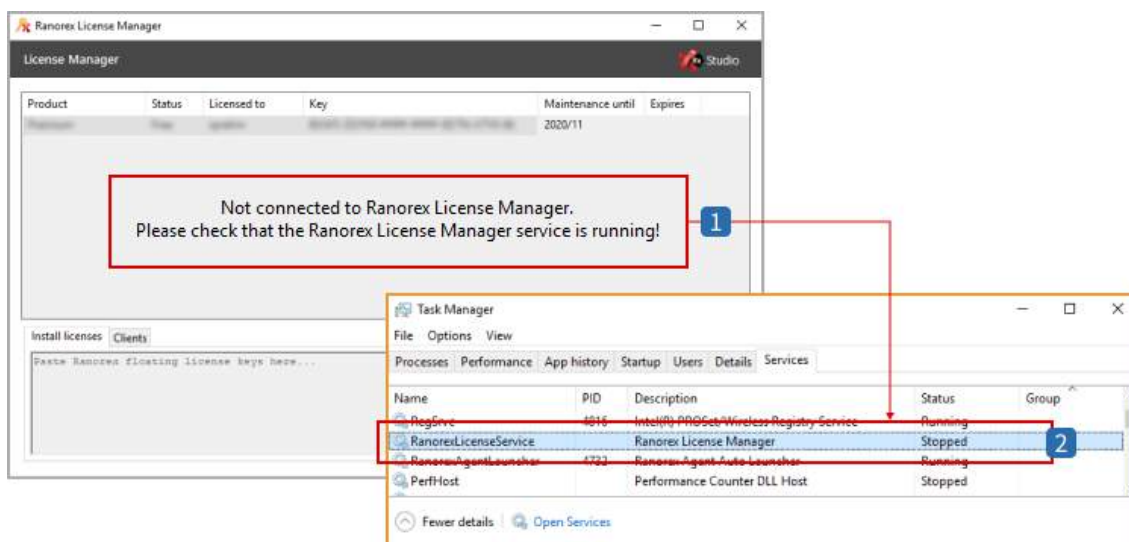
- Always install the latest version of the License Manager
- The License Manager works with networks based on IPv4, IPv6, and a mix between them.
- Port 7266 TCP/UDP must be open.

After the installation, the License Manager starts:



## Troubleshooting

In rare cases, the Windows service (Windows Task Manager > Services) for the License Manager may not be started. In this case, the License Manager will display the following error message and won't work.



1

Error message in the Ranorex License Manager

2

Windows Task Manager showing the stopped RanorexLicenseService

## Cause

The service may trigger a false positive in some anti-virus programs, causing them to deactivate the service.

## Solution

Restart the service in the Windows Task Manager. If it still doesn't work, reinstall the License Manager. If the problem persists, please contact support.

## Install with internet connection

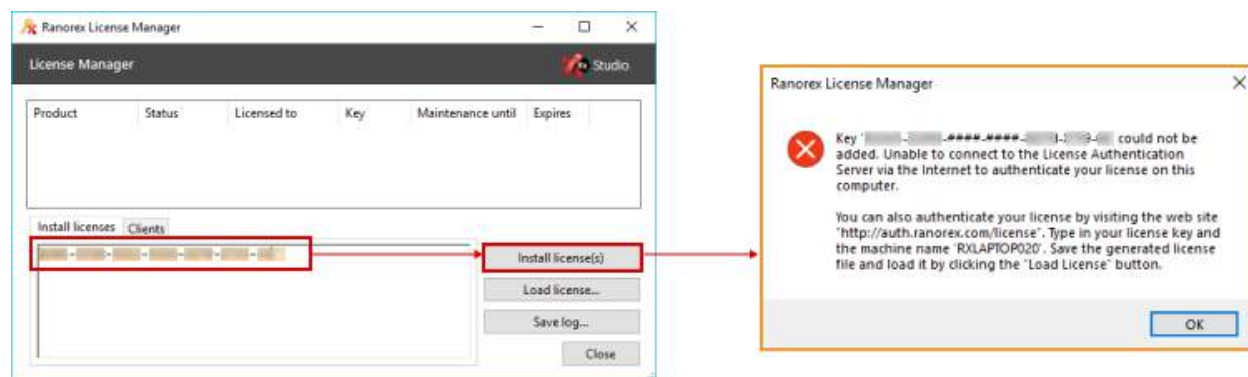
Before you can use a floating license, it needs to be installed in the Ranorex License Manager on your server. This way, it is registered and authenticated by the Ranorex License Authentication Server. (The Ranorex License Authentication Server is not the server for your client machines; it's a Ranorex company server.)

To install floating licenses:

- 1 Under Install licenses, **paste** the license key(s) you received by email.
- 2 **Click Install license(s).**
- 3 **Click OK.**
- 4 The floating license(s) appear in the list of installed licenses. This means they have been authenticated by the Ranorex License Authentication Server and installation is complete.

## Install without internet connection

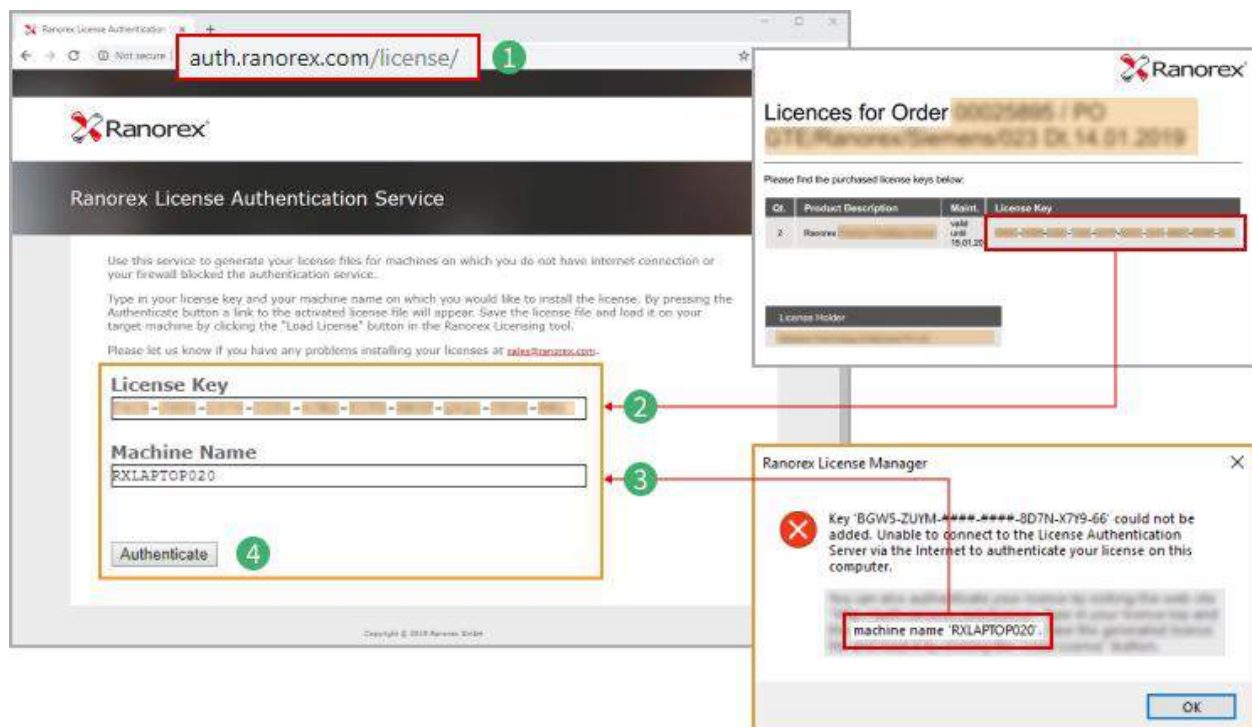
If you try to install a license and the server isn't connected to the internet, the following error message will appear:



In this case, do the following:

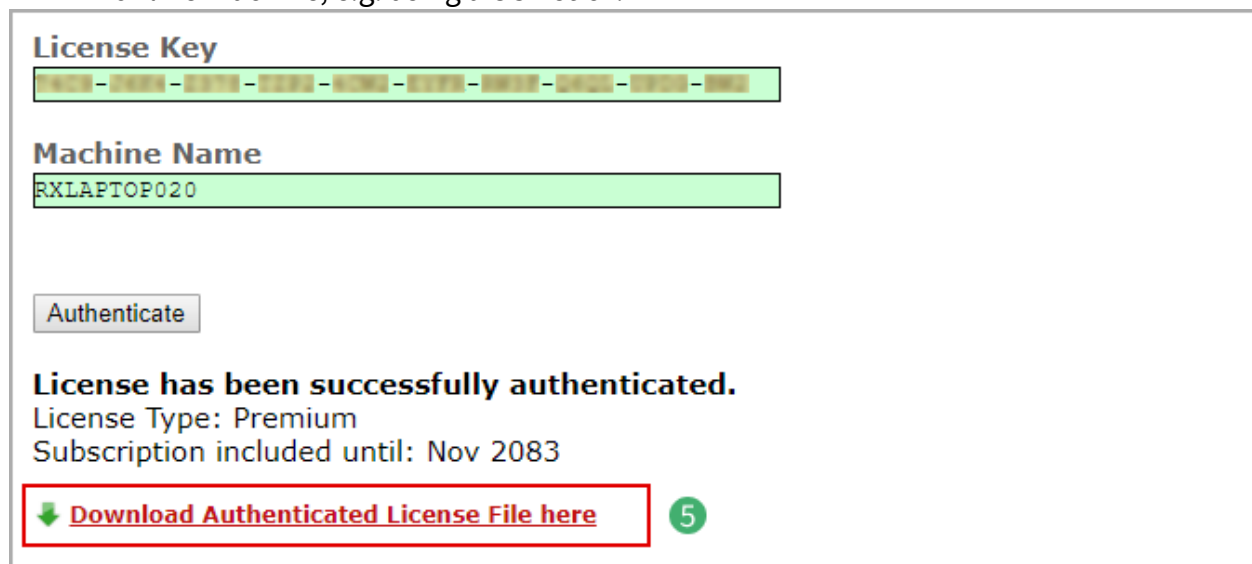
- 1 On a machine with an internet connection, **go** to <http://auth.ranorex.com/license/>.
- 2 Under License Key, **enter** your license key.
- 3 Under Machine Name, **enter** the machine name of your server.

- 4 Click **Authenticate** and **confirm** the machine name of your server.

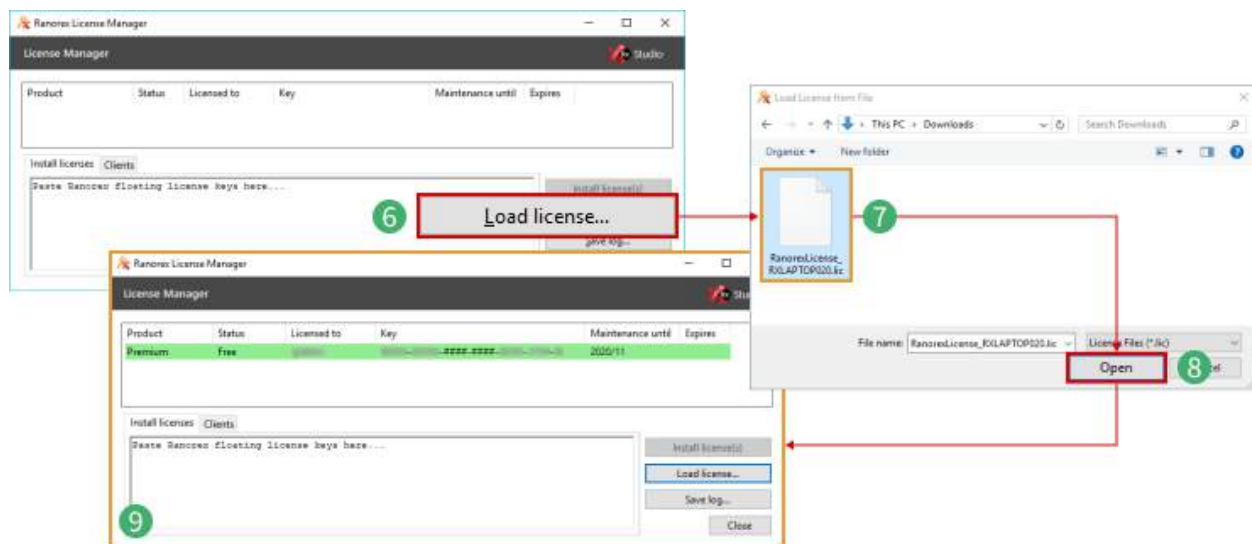


The Ranorex License Authentication Server creates an authenticated license file based on your data. You need to download this file and transfer it to the offline machine.

- 5 Click **Download Authenticated License File** and transfer the downloaded file to the offline machine, e.g. using a USB stick.



- 6 In the License Manager, **click Load license....**
- 7 **Browse** to the authenticated license file.
- 8 **Click Open.**
- 9 **Click OK.** The license appears under the installed floating licenses.



Once the license has been installed and authenticated on the server, clients can lease it. The server **doesn't have to be connected** to the internet for normal operation. (It must be connected to the same network as the clients, however.)

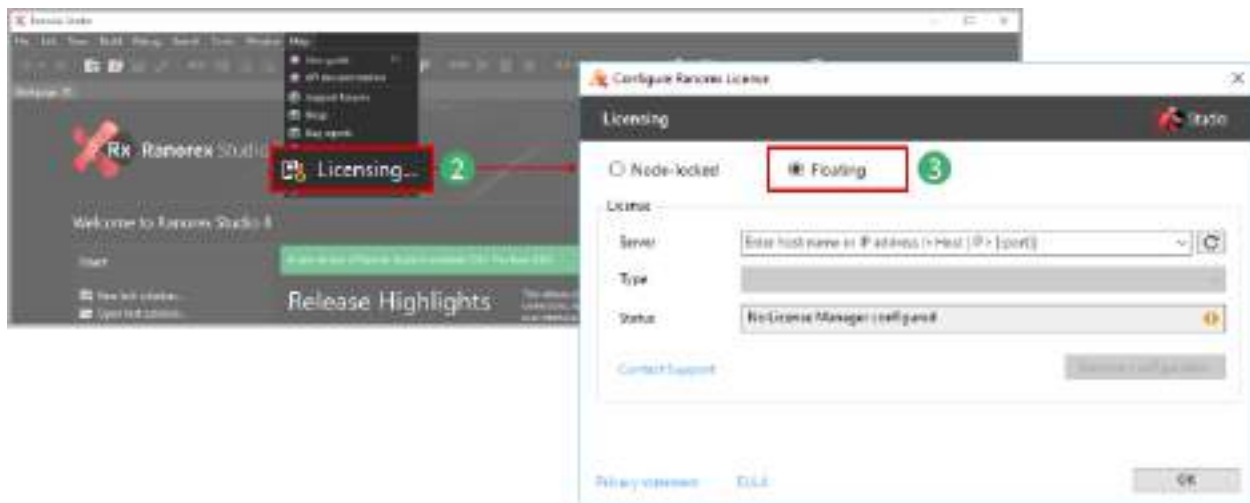
## Configure a client

Once you've installed a floating license on the server, you need to configure the clients so they can lease the license.

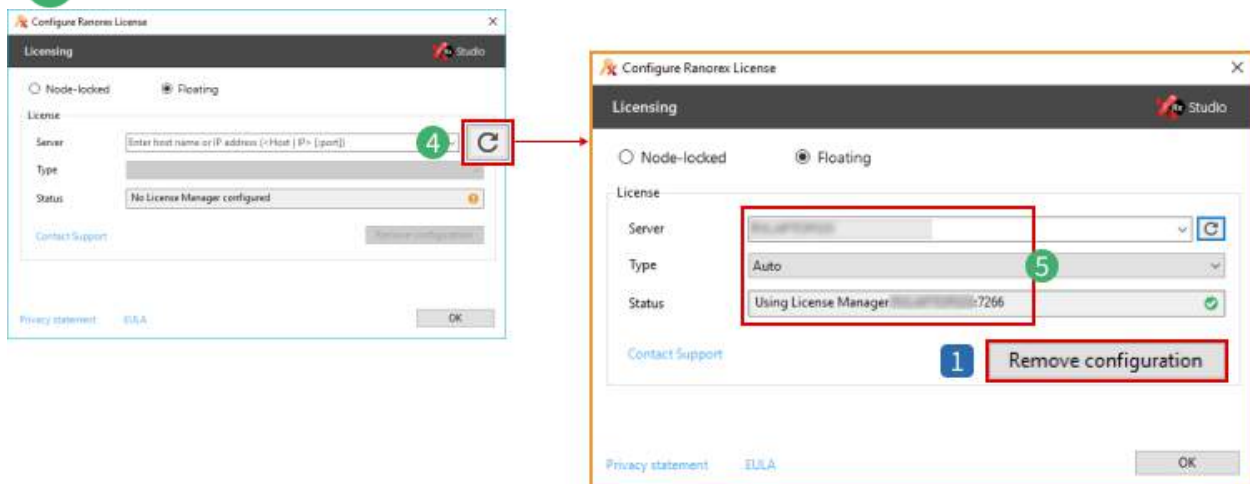
Make sure the client is connected to the same network as the server and can access it.

To configure a client:

- 1 On the client, **open** Ranorex Studio.
- 2 **Go to Help > Licensing.** The license configuration dialog opens.
- 3 **Click Floating.**



- 4 Under **Server**, enter the machine name or IP address of your server, or click the **Refresh** icon to see a list of machines with installed Ranorex License Managers in the network.
- 5 The other fields are configured automatically.



- 1 Click this button to reset all fields. Useful if you also have a node-locked license installed on the machine and want to use only it.

## License types

There are three types of floating licenses: Runtime, Professional, and Premium.

When configuring a client, you can use the Type drop-down menu to specify the type of license you want this client to use. For example, if you plan to only run tests on this client, then Runtime would suffice. However, **we recommend you leave Type on Auto.**

Type	<div>Auto</div> <div>Auto</div> <div>Runtime</div> <div>Professional</div> <div>Premium</div>
------	---

### **Auto (recommended)**

The client automatically requests the correct license for the current operation from the server.

### **Runtime**

The client can only request a Runtime Floating License. This license type is limited to running tests via an Agent, the command line, or the Ranorex Test Suite Runner.

### **Professional (discontinued)**

The client can only request a Professional Floating License. This license type is limited to Runtime Floating operations and using Ranorex Spy and tracking.

### **Premium**

The client can only request a Premium Floating License. This license type has no limitations and is required for using the full functionality of Ranorex Studio, e.g. starting it, recording tests, or editing test suites (including through the API).

## **How license leasing works**

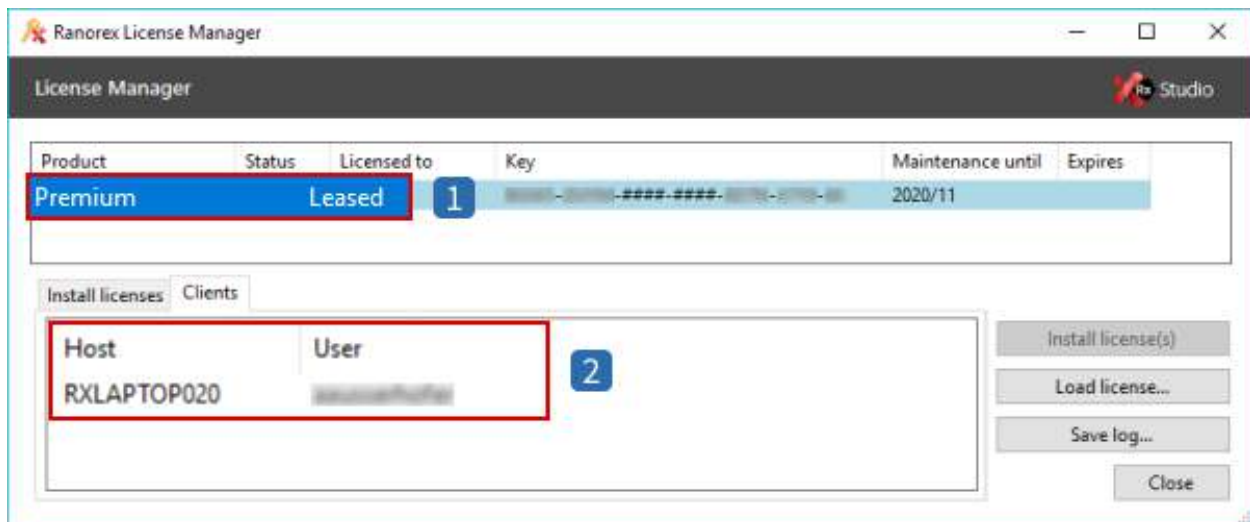
Leasing works based on what you're doing in Ranorex Studio or one of its tools. For example, simply starting Ranorex Studio triggers the lease of a Premium Floating License. Running a test on a Ranorex Agent, on the other hand, triggers the lease of a Runtime Floating License, but if that isn't available, it can also use a Premium Floating License. To see what the different license types allow, see **License types** above.

When a client has leased a license from the server, the license becomes unavailable to other clients. After the triggering operation is finished (e.g. the test run has finished, Ranorex Studio is closed), the client returns the license to the server and it's available to other clients again.



### Note

If a floating license **can't be returned** to the server because of a network error, it will be **blocked for 5 minutes**, meaning it's completely unavailable for lease during that time. We therefore strongly recommend a stable network infrastructure.



- 1 The License Manager shows that the Premium Floating License is currently leased and therefore unavailable to other clients.
- 2 Under **Clients**, you can see the machine name and user who's leasing the license.

## Uninstall a license

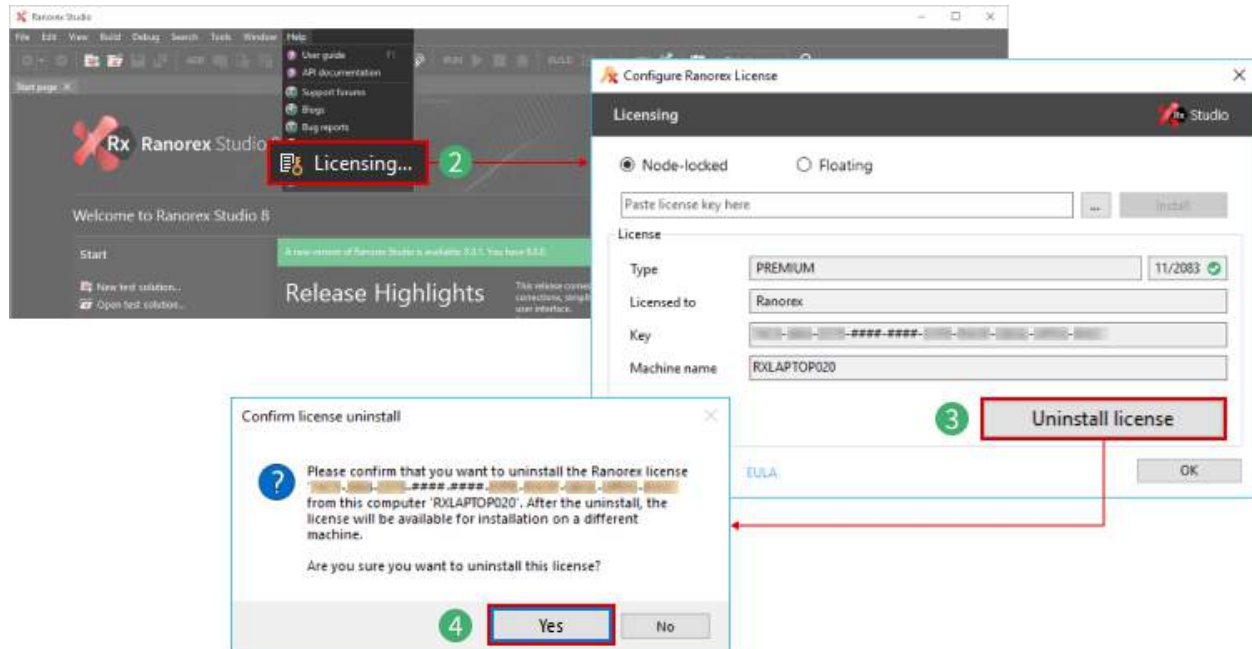
In this chapter, you'll find out how to uninstall node-locked and floating licenses.

### Uninstall a node-locked license

To uninstall a node-locked license:

- 1 **Start** Ranorex Studio as an administrator.
- 2 **Go to Help > Licensing...** The license configuration dialog opens.

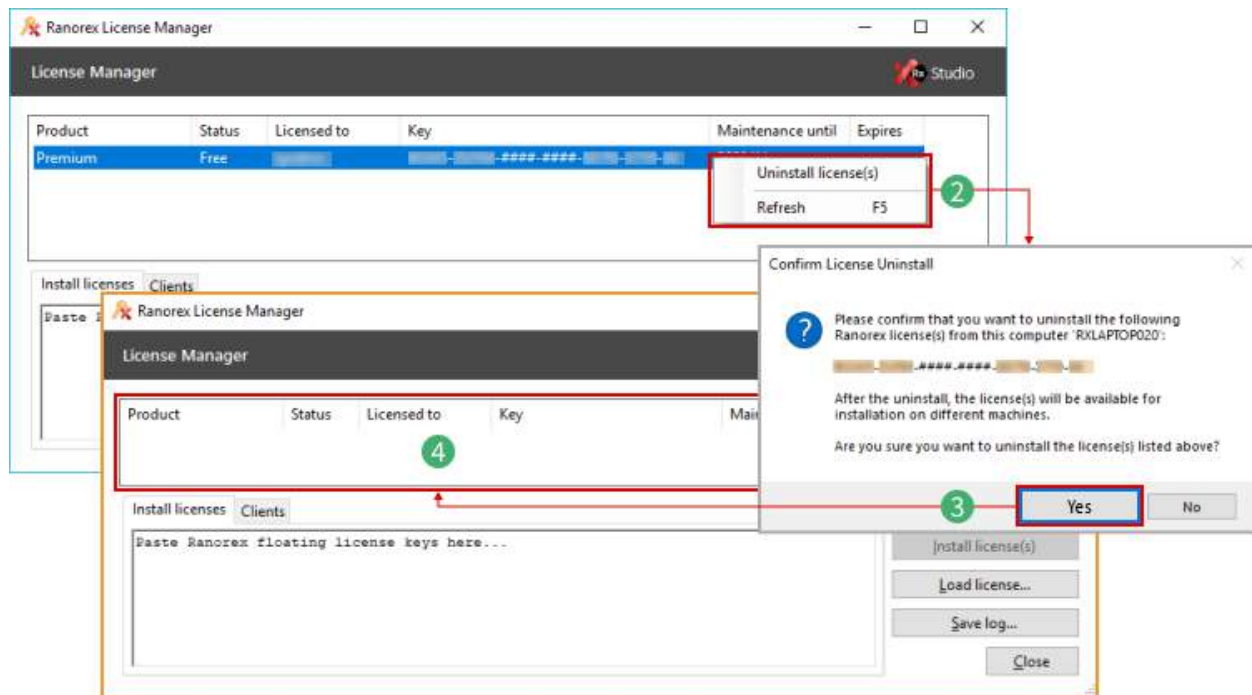
- 3 Click Uninstall license.
- 4 Click Yes.



Ranorex Studio will automatically remove the license from the Ranorex License Authentication Server and the configuration dialog.

## Uninstall a floating license

- 1 On your server, **start** the Ranorex License Manager.
- 2 In the context menu of the license(s) you want to uninstall, **click Uninstall license(s)**.
- 3 **Click Yes.**
- 4 The license is deleted from the License Manager.

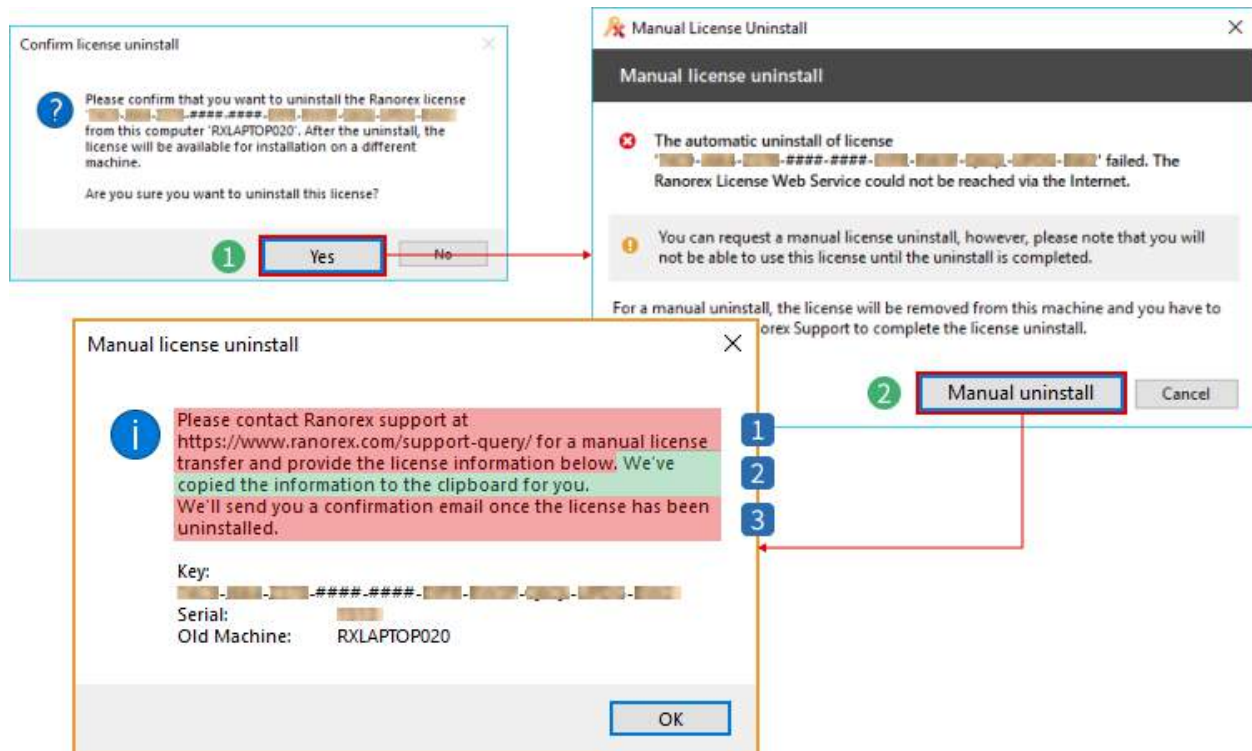


## Troubleshooting

### No connection to the Ranorex License Authentication Server

If the machine you want to uninstall a license from can't connect to the Ranorex License Authentication Server during the uninstall workflow, the following will happen:

- 1 When you click **Yes** to confirm the license uninstall, an error message appears.
- 2 **Read** the message, **click Manual uninstall**, and **follow** the instructions in the next dialog:



- 1 On a machine with an internet connection, **open** the [support query form](#) and apply for a **support ticket** with your personal support key. Request a manual license uninstall and provide the information of the dialog window.
- 2 The license information is shown in the dialog and has been copied to the clipboard for you. Simply **paste** it into the support query form, or attach a screenshot of the dialog window to it.
- 3 Before you reuse the license, **wait** for the confirmation email that the license has been successfully uninstalled and deregistered on the Ranorex License Authentication Server.

## Transfer a license

In this chapter, you'll find out how to transfer node-locked and floating licenses. Transferring a license means uninstalling a license on one machine and installing it on another.

## Overview

### Note

- You must have administrator privileges on both the source and target machines to transfer licenses.
- You can only transfer licenses yourself if outside the lock period of a license (e.g. 90 days since last install for a floating license). While inside the lock period, you must contact our support team for a license transfer.



## Transfer a node-locked license yourself

### Requirements

- You are outside the lock period of the license

### Uninstall the license from the source machine

- 1 On the source machine, uninstall the license as described in → [Uninstall a license...](#)
  - a ...under **Uninstall a node-locked license** if the source machine **can connect** to the Ranorex License Authentication Server.
  - b ...under **Troubleshooting** if the source machine **can't connect** to the Ranorex License Authentication Server.

### Install the license on the target machine

- 2 On the target machine, install the license as described in → [Install a node-locked license...](#)
  - a ...under **Install with internet connection** if the target machine **can connect** to the Ranorex License Authentication Server.
  - b ...under **Install without internet connection** if the target machine **can't connect** to the Ranorex License Authentication Server.

## Transfer a floating license yourself

### Requirements

- More than 90 days have passed since you last installed the license.

### Uninstall the license from the source machine

- 1 On the source machine, uninstall the license as described in → [Uninstall a license...](#)
  - a ...under **Uninstall a floating license** if the source machine **can connect** to the Ranorex License Authentication Server.
  - b ...under **Troubleshooting** if the source machine **can't connect** to the Ranorex License Authentication Server.

### Install the license on the target machine

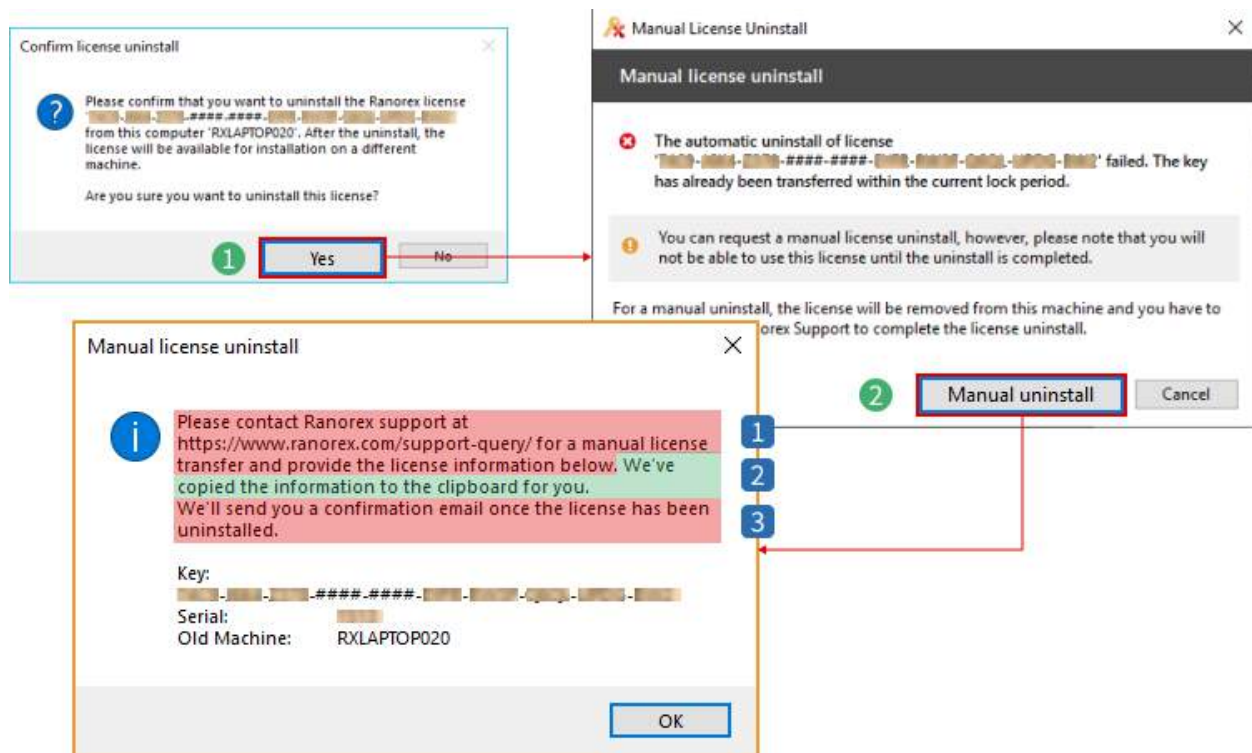
- 2 On the target machine, install the license as described in → [Install a floating license...](#)
  - a ...under **Install with internet connection** if the target machine **can connect** to the Ranorex License Authentication Server.
  - b ...under **Install without internet connection** if the target machine **can't connect** to the Ranorex License Authentication Server.

### Transfer within the lock period with help from support

You can only transfer licenses yourself if outside the lock period of a license. If you need to transfer a license within the lock period, you need to contact us with your license information. To do so, follow the workflow below.

When you try to transfer a license within the lock period, the following will happen:

- 1 When you click **Yes** to confirm the license uninstall, an error message appears.
- 2 **Read** the message, **click Manual uninstall**, and **follow** the instructions in the next dialog:



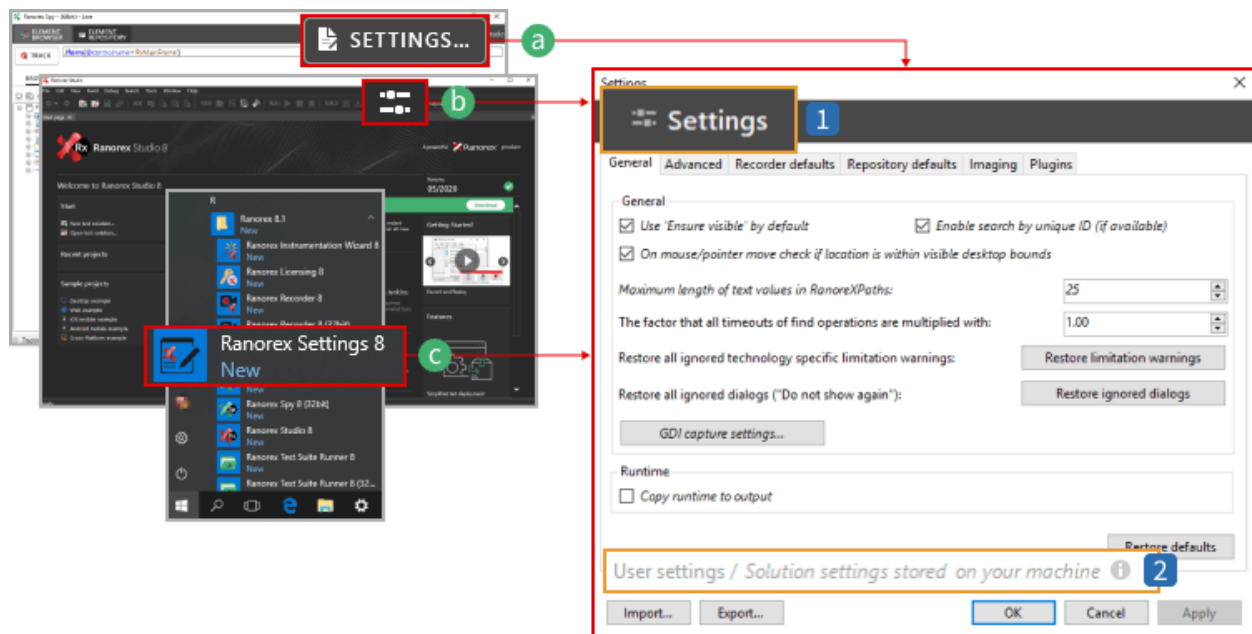
- 1 On a machine with an internet connection, **open** the [support query form](#) and apply for a **support ticket** with your personal support key. Request a manual license transfer and provide the information of the dialog window.
- 2 The license information is shown in the dialog and has been copied to the clipboard for you. Simply **paste** it into the support query form, or attach a screenshot of the dialog window to it.
- 3 Before you reuse the license, **wait** for the confirmation email that the license has been successfully uninstalled and deregistered on the Ranorex License Authentication Server.

# Settings & configuration

This chapter introduces and explains the various options for setting and configuring Ranorex Studio and its tools.

## Machine-stored settings

If you access settings from a Ranorex tool without a solution associated with it, the settings are stored on the machine in a Ranorex internal file structure.



### How to access machine-stored settings:

- a Click the **SETTINGS** button from **stand-alone Spy**
- b Click the **settings** button from **Ranorex Studio start page** (with no solution opened)
- c Click **stand-alone Ranorex Settings** program from Windows start menu

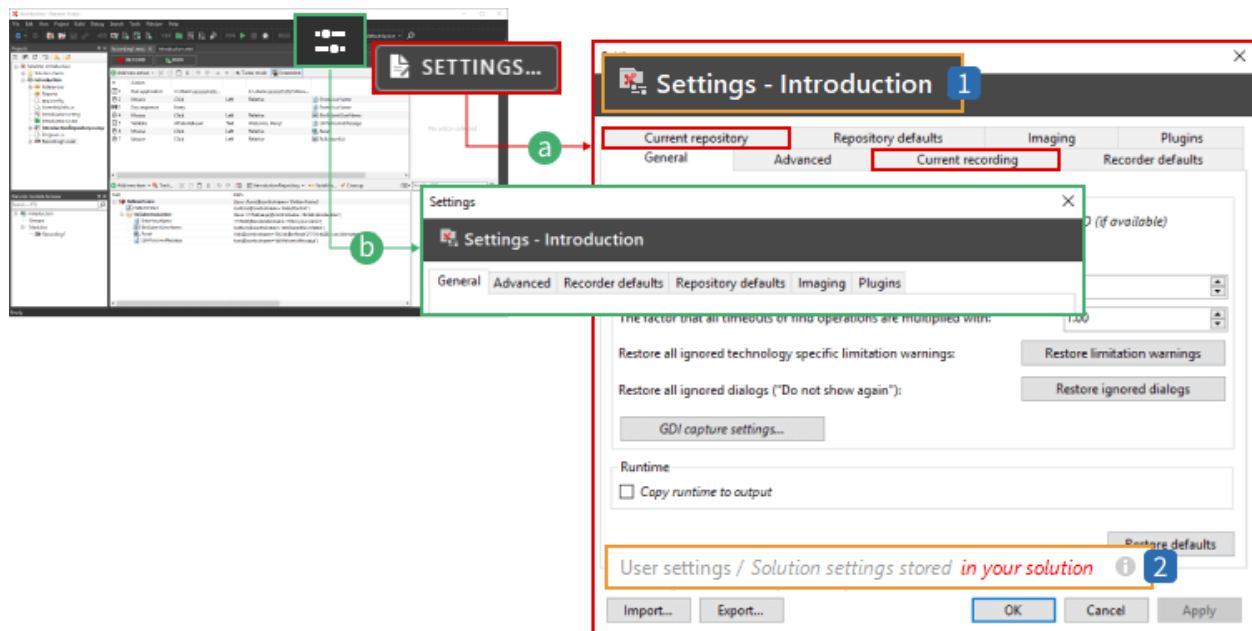
### Settings dialog:

- 1 See the settings title with **no solution** associated
- 2 See the information stating that the settings are **stored on the machine** and not with a solution

## Solution-stored settings



Whenever you access settings from a Ranorex tool with an associated solution, the settings are stored in this solution.



### How to access solution-stored settings:

- a Click the **SETTINGS** button in Ranorex Studio while a solution is opened
  - This option opens an **extended** settings dialog with ...
  - ...**Current recording** settings dialog and ...
  - ...**Current repository** settings dialog register tab
- b Click the **settings toolbar button** in Ranorex Studio to open the regular settings dialog

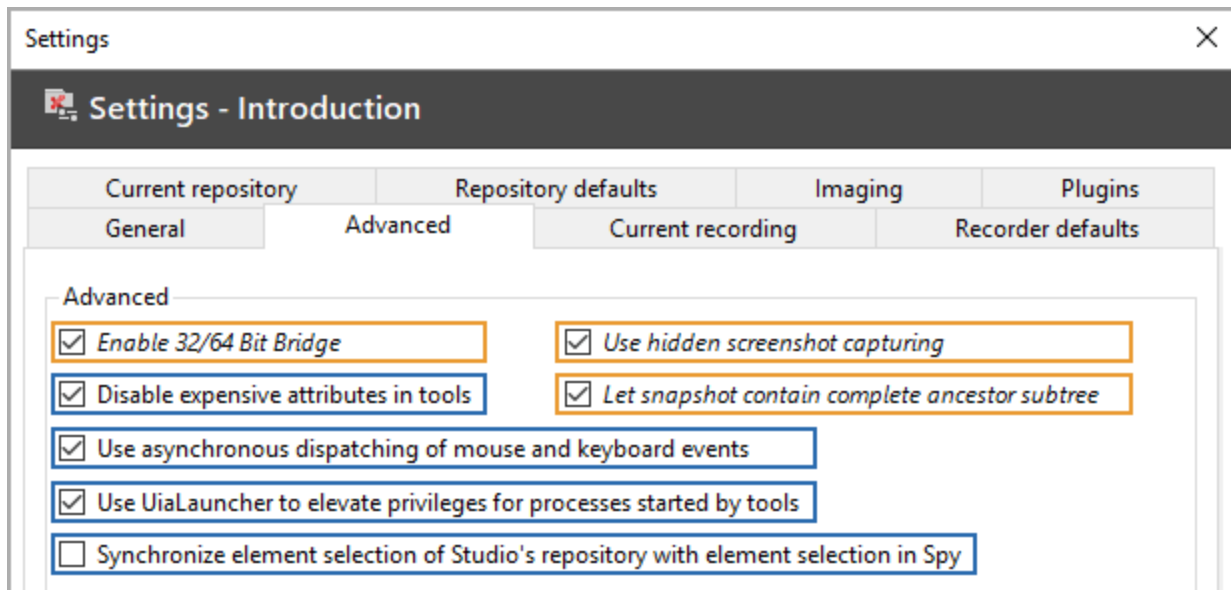
### Settings dialog:

- 1 See the **setting title with associated solution name**
- 2 See the information stating that the setting **is stored in the associated solution**

## User-settings & solution-settings

**User settings** are always stored locally on a machine and include user-specific settings related to the Ranorex Studio working environment.

**Solution settings** include all settings which relate to the test solution, such as plugins, RanoreXPath generation, Ranorex Recorder defaults and many more, which can be shared between different systems, within teams and also be put under Version control.

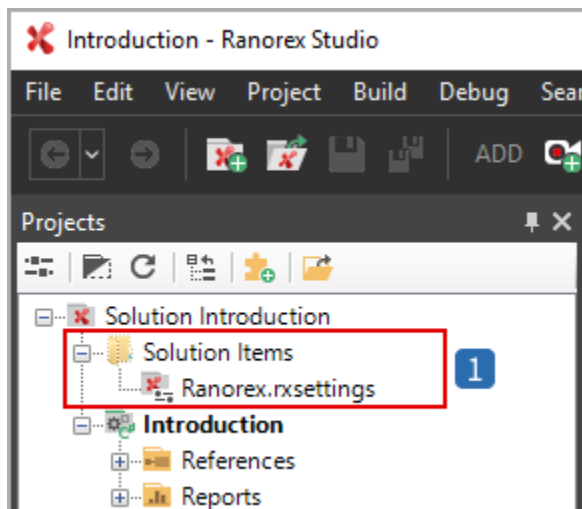


- 1 User settings
- 2 Solution settings

- 1 **User-settings** are printed in regular font
- 2 **Solution settings** are printed in italic font

## Solution settings file

The solution settings are saved in the settings file `Ranorex.rxsettings`, which is located in the 'Solution Items' folder of the corresponding solution.



- 1 Solution settings file in **project file view** of Ranorex Studio

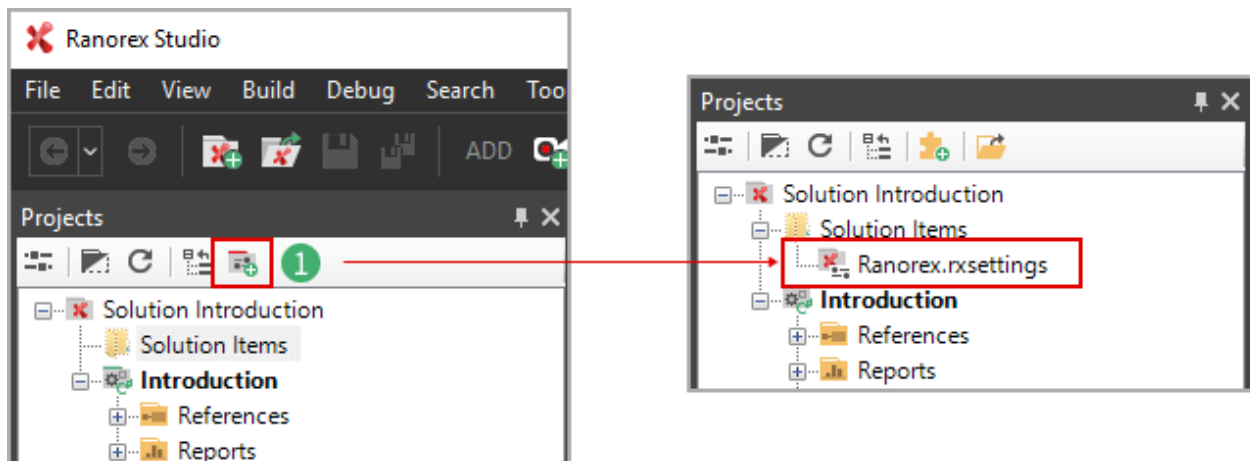


### Attention

- Do not change the name of the folder or the file
- If you remove the settings file (`Ranorex.rxsettings`), the solution settings stored on the local machine are automatically used during test execution
- If you open the settings file, it is shown in its raw format. It is not recommended to edit the file in this view

## Adding a solution settings file

If a solution does not contain a settings file, it can be added following the below-listed instructions.



1

Click the **Create solution settings** in the 'Projects view' toolbar of a solution

## Solution settings and test execution

The solution settings file `Ranorex.rxsettings` is automatically copied into the output directory of the test project. It is saved in the same folder as the test executable. This is why the solution settings will be used for test execution.

### Note

- Do not rename the solution settings file
- When deploying the test executable, make sure to include the solution settings.

If the solution settings are not stored in the same folder as the test executable, the solution settings from the local machine will be used.

## Solution settings and Ranorex Remote

When selecting a solution for remote test execution, the solution settings saved in this solution will automatically be sent to the Ranorex Agent and used during test execution.

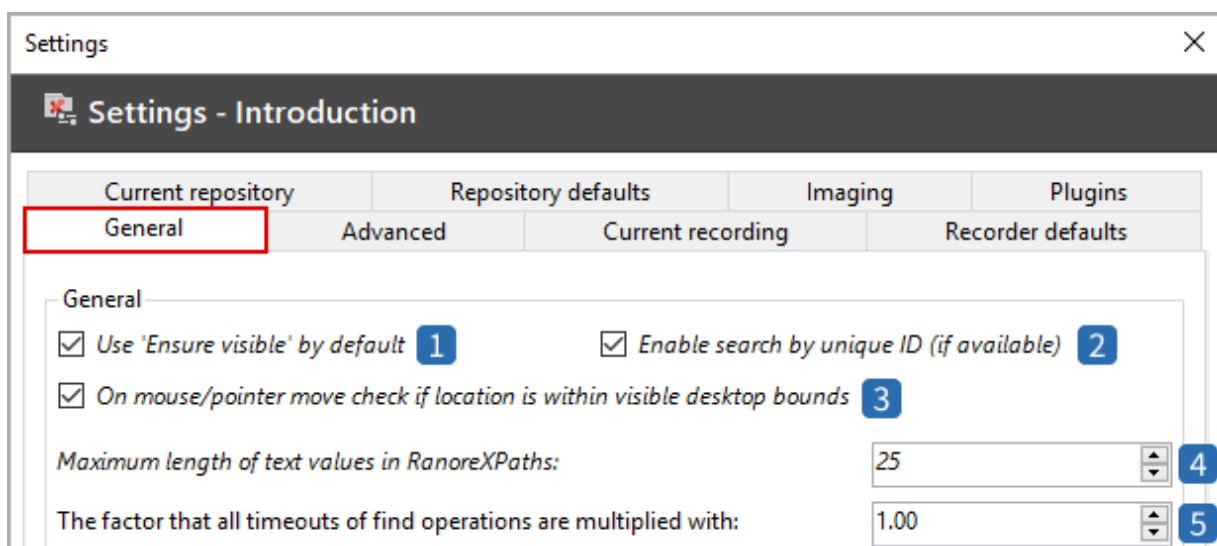
If the solution does not include a solution settings file, the solution settings stored locally on the Ranorex Agent's machine will automatically be used during test execution.

## Ranorex Recorder settings

The purpose of this chapter is to introduce and explain the various options for setting and configuring Ranorex Recorder.

### General settings

General settings are the ones which can be accessed by selecting the '**General**' register tab in the settings dialog window.



- 1 **Use 'ensure visible' by default:** Specifies whether a UI-element used as Ranorex role should be forced to become visible before automation or not. By default, this value is automatically used for each repository item
- 2 **Enable search by unique ID (if available):** Specifies whether web elements should be identified using their unique id or not
- 3 **On mouse/pointer move check if location is within visible desktop bounds:** Specifies whether an exception should be thrown if the mouse/pointer tries to move outside the visible desktop bounds or not
- 4 **Maximum length of text values in RanoreXPath:** Specifies the maximum length of attribute values used within RanoreXPath
- 5 **The factor that all timeouts of find operations are multiplied with:** Multiplies all repository timeouts with the specified value to prevent test cases from failing in case of different timing behavior. This is especially useful when executing tests on different system configurations

**Note**

The value can also be set at the time of executing a test suite using the test suite's command line arguments, or by directly using the `Ranorex.Core.Configuration.Current.Adapter.TimeoutFactor` property in code.

The screenshot shows the 'Ranorex Configuration' dialog box. It has several sections and buttons, each with a numbered callout:

- 6** points to the 'Restore limitation warnings' button in the top right.
- 7** points to the 'Restore ignored dialogs' button in the top right.
- 8** points to the 'GDI capture settings...' button on the left.
- 9** points to the 'Copy runtime to output' checkbox in the 'Runtime' section.
- 10** points to the 'Include license information' button in the 'Runtime' section.
- 11** points to the 'Show invisible characters in report' checkbox in the 'Report' section.

There is also a warning message in the 'Runtime' section: "Open a solution and ensure a premium floating license is configured in the License Manager first."

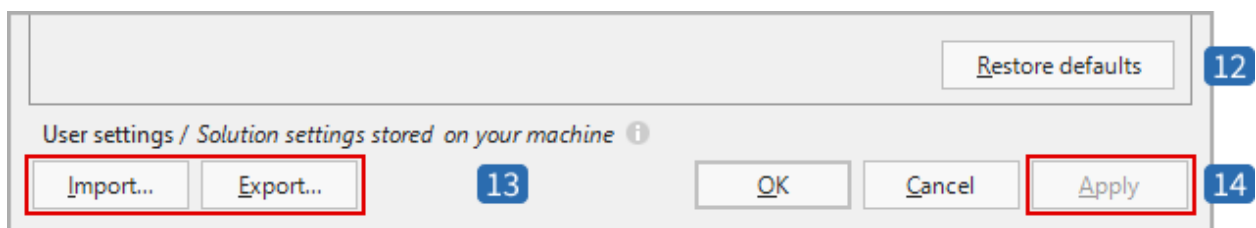
- 6 Restore limitation warnings:** Resets the ‘Do not show again’ checkboxes used to suppress the assistance dialog for technology limitations. Click to show the dialog again for non-instrumented technologies or applications
- 7 Restore ignored dialogs:** Resets all other ‘Do not show again’ checkboxes
- 8 GDI capture settings...:** Opens the dialog to change the current GDI capture list



### Further reading

The GDI capture feature of Ranorex is introduced and explained in detail in  
 > Ranorex Studio advanced > Ranorex Spy > → [GDI capture feature](#).

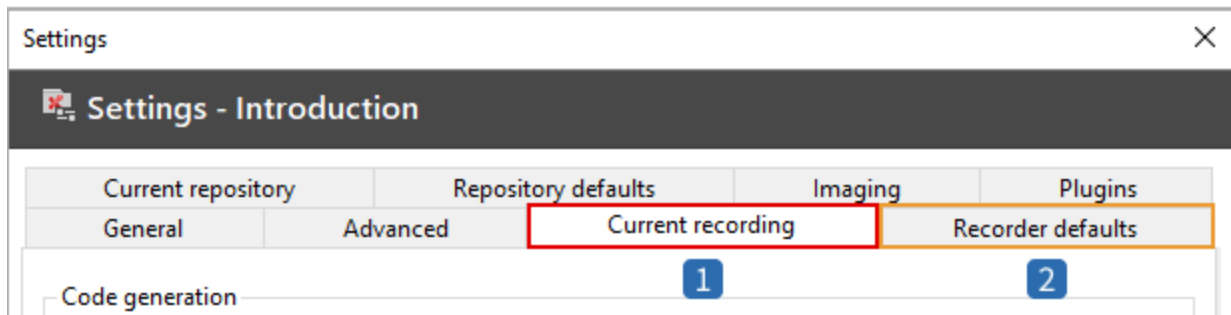
- 9 Copy runtime to output:** If checked, copies the Ranorex Studio runtime to the output folder when compiling/running a test
- 10 Include license information:** Click to include the local machine’s floating license information with the test. For this to work, you must have a solution opened and a floating license configured on the local machine.
- 11 Show invisible characters in report:** Display otherwise invisible characters in report values.



- 12 Restore defaults:** to reset all values to their default values
- 13 Import.../Export...:** to save and load user-specific configurations
- 14 Apply:** is necessary to make changes valid

## Recorder settings overview

Recorder settings can be accessed through the settings dialog. Depending on from where the dialog is opened, different options are available. If you open the settings from within the Studio Recorder view (**SETTINGS** button in action table), the configuration options for the current recording are available in addition to the recorder default settings.



- 1 Current recording:** This tab primarily contains configuration parameters for code generation specific to the current recording. All settings on this tab page are stored within the recording.
- 2 Recorder defaults:** In this tab global default values for every newly created recording are specified.

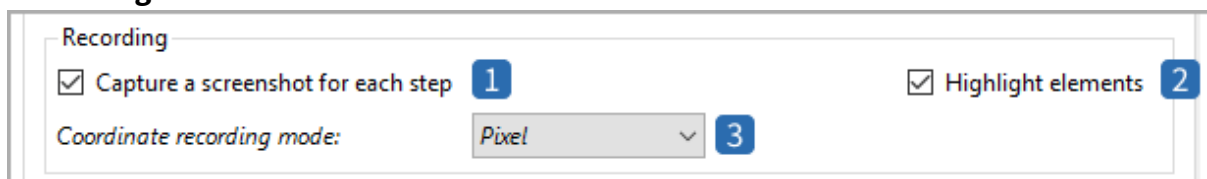
#### Hint

The settings tab page for **Current recording** can only be accessed from the **SETTINGS** button within a recording.

## Recorder default settings

In this tab, you can specify global default values for every newly created recording. The settings are divided into three sections:

### Recording



- 1 Capture a screenshot for each step:** Specifies whether a screenshot of the current action should be made during the recording.
- 2 Highlight elements:** Specifies whether element highlighting during recording should be enabled.
- 3 Coordinate recording mode:** Specifies the way coordinates will be recorded. Following options are available:

None:	Actions will always be invoked on the 'center' of the recognized elements
Pixel:	Relative coordinates within a recognized element are recorded in pixels
Proportional:	Relative coordinates within a recognized element are recorded in percent

## Timings for new actions



### Note

Please note that the following settings only affect newly created actions, not existing actions.

Timings for new actions

☒ Use timings also for recorded actions 4
☒ Use keystroke optimization 5

Key sequence split time (ms):

1000

6

Mouse move time per action (ms):

300

7

Mouse/Pointer-related action time (ms):

500

8

Keyboard-related action time (ms):

100

9

4

**Use timings also for recorded actions:** When enabled, the specified timings below will be taken instead of the recorded durations.

5

**Use keystroke optimization:** Enabled by default. This setting makes recorded key sequences easier to read. Only applies to characters of the English alphabet, i.e. A–Z. Can also be enabled/disabled in the keystroke editor of an action.

### Disabled

{LShiftKey down} h {LShiftKey up}

{CAPITAL} hello

helll {Back} o

### Enabled

H

HELLO

hello



hello {Space} there

hello there

- 6 Key sequence split time (ms):** Use this setting to specify the maximum time between key presses in a key sequence during recording. If this time is exceeded, the sequence is split into multiple key sequences.
- 7 Mouse move time per action (ms):** Specifies the time in milliseconds used to move the mouse to the UI object related to the action.
- 8 Mouse/Pointer-related action time (ms):** Specifies the overall time in milliseconds used for a manually created mouse action. The value set for this setting is only taken into account for manually created actions, when the time of an action cannot be determined during recording or when 'Use timings also for recorded actions' is activated.
- 9 Keyboard-related action time (ms):** Specifies the default overall time in milliseconds used for a manually created keyboard action. The value set for this setting is only taken into account for manually created actions, when the time of an action cannot be determined during recording or when 'Use timings also for recorded actions' is activated.

### Code generation

Code generation

Recording namespace:

MyTestProject

Recording class name:

New\_Recording

☒ Use item logging by default

☒ Generate replay reports

☒ Show warnings for possible invalid property values

- 10 Recording namespace:** Specifies the namespace used for the generated code.
- 11 Recording class name:** Specifies the class name used for the generated code.
- 12 Use item logging by default:** Activate this setting to turn on a default logging message for each action item.
- 13 Generate replay reports:** This setting is used to turn the generation of reports during replay on or off.
- 14 Show warnings for possible invalid property values:** Option to turn invalid property warnings on/off.

### Current recording setting

This tab primarily contains configuration parameters for code generation specific to the current recording. All settings on this tab page are stored within the corresponding recording.

Code generation

Description: (XmlDoc summary) The Recording1 recording.

Recording namespace: Introduction

Recording class name: Recording1

Replay speed factor: 1.00

Replay iteration count: 1

Recorder UI mode: Desktop & Web

- 1 Description:** Can be used to briefly describe the actions the recording performs.
- 2 Recording namespace:** Specifies the namespace used for the generated code.
- 3 Recording class name:** Specifies the class name used for the generated code.
- 4 Replay speed factor (%):** The speed factor is used to increase or decrease the replay overall speed by a specific factor value.
- 5 Replay iteration count:** The repeat count is used to specify the number of iterations.
- 6 Recorder UI mode:** By specifying the recorder UI mode the set of actions available in the actions table will be adapted (“Global”: all actions, “Desktop & Web”: no mobile actions, “Mobile”: no mouse and keyboard actions).

☐ Enable turbo mode for replay and generated code

☒ Generate replay reports

☒ Use item logging by default

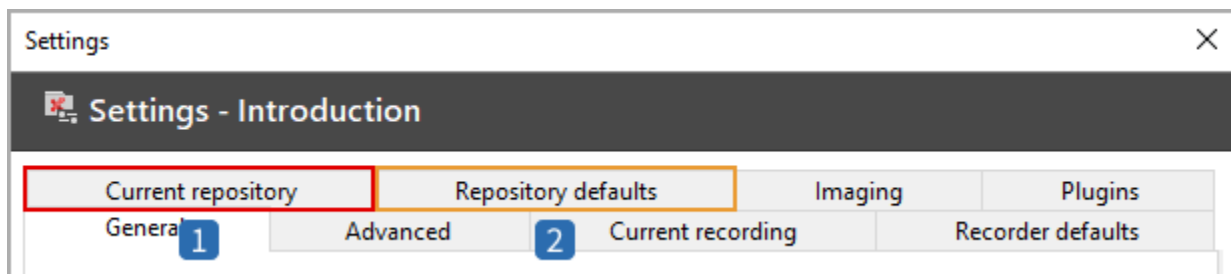
- 7 Enable turbo mode for replay and generated code:** Is used to specify whether recorded delays between actions should be part of the generated code.
- 8 Generate replay reports:** This setting is used to turn the generation of reports during replay on and off.
- 9 Use item logging by default:** Activate this setting to turn on a default logging message for each action item.

## Repository settings

This chapter explains the various settings to configure repositories.

### Repository settings overview

Repository settings can be accessed through the settings dialog. Depending on from where the dialog is opened, different options are available. If you open the settings from within the Studio Recorder view (**SETTINGS** button in action table), the configuration options for the current repository are available in addition to the repository default settings.



- 1 Current repository:** This tab primarily contains configuration parameters specific to the current repository. All settings on this tab page are stored within the corresponding repository.
- 2 Repository defaults:** In this tab global default values for every newly created repository are specified.

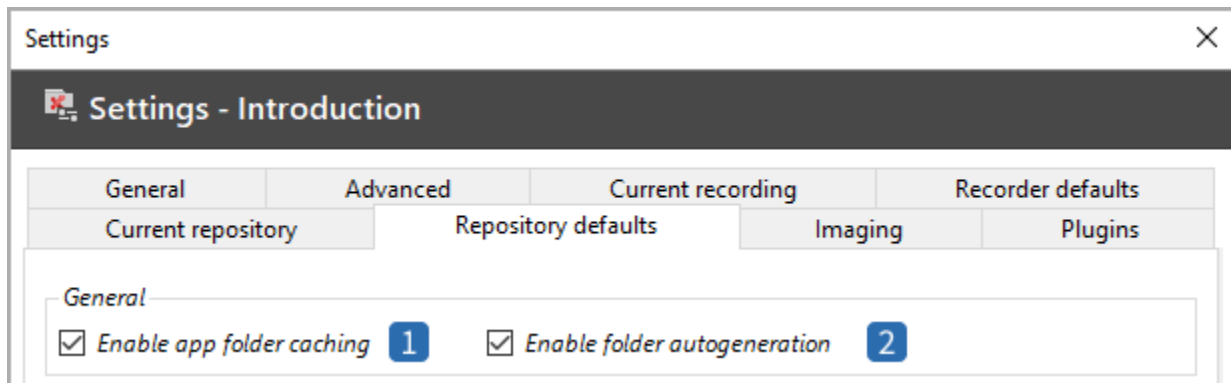
#### Hint

The settings tab page for **Current repository** can only be accessed from the **SETTINGS** button within a recording.

### Repository default settings

The repository default settings are structured in three main sections:

## General:



**1 Enable app folder caching:** This setting can be unchecked to turn-off folder caching for all items by default.

**2 Enable folder autogeneration:** Uncheck this option to prevent the repository from creating rooted folders automatically.

## Timeouts for new entries:

The timeouts used for newly created repository folders and items can be changed in this group box.



**3 Search timeout for application folders (ms):** Timeout in ms defined for searching for an application folder during runtime

**4 Search timeout for rooted folders (ms):** Timeout in ms defined for searching for a rooted folder during runtime

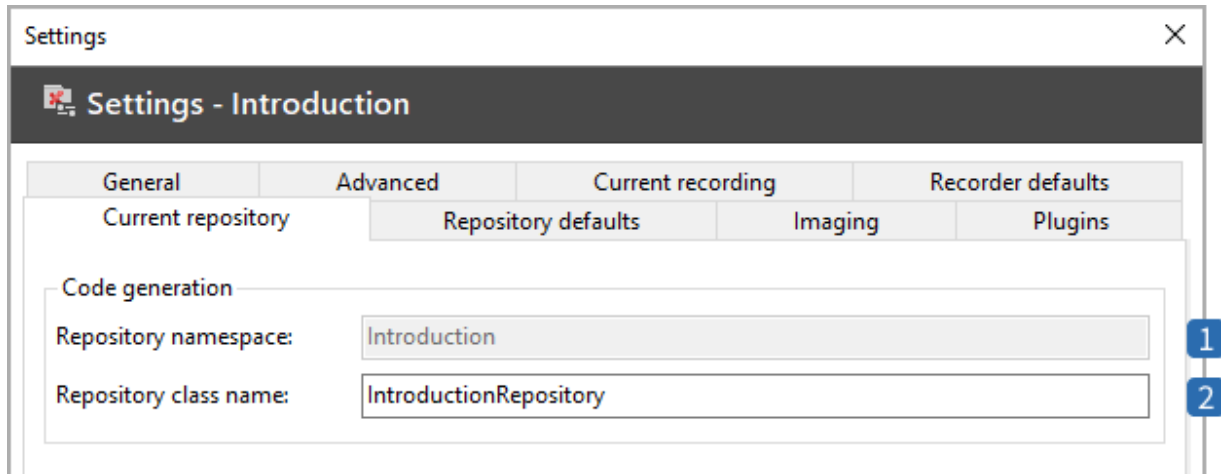
**5 Search timeout for items (ms):** Timeout in ms defined for searching for any other repository item during runtime

## Code generation:



- 6 **Repository namespace:** Non-editable namespace for repositories
- 7 **Repository class name:** Option to specify the repository class name for newly created repositories

## Current repository settings



- 1 **Repository namespace:** Non-editable namespace for current repository
- 2 **Repository class name:** Option to specify the class name for the current repository

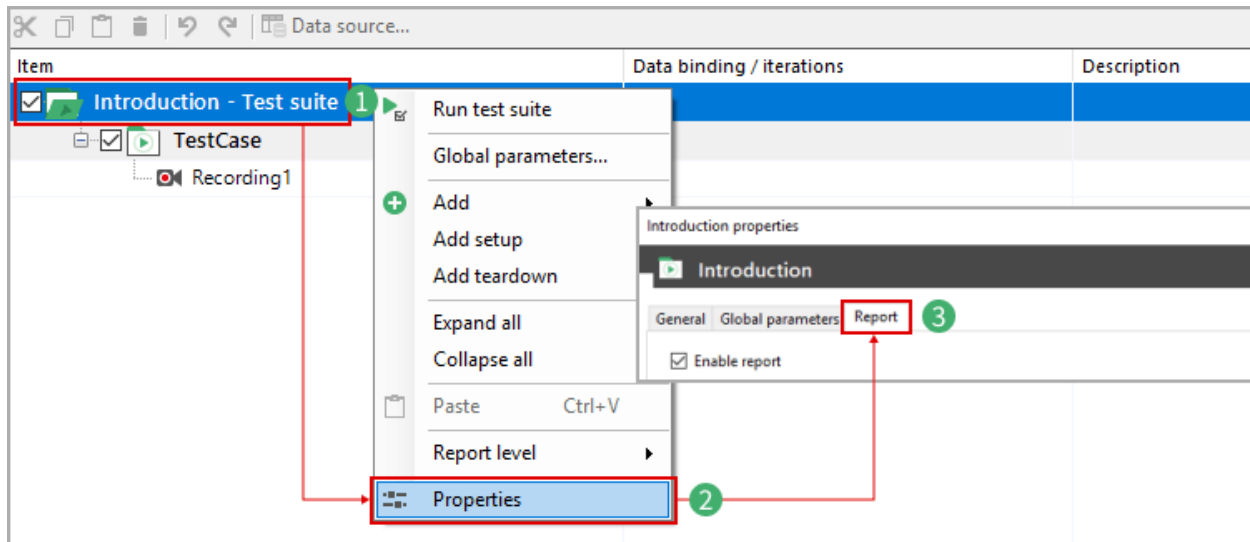
## Report settings

This chapter introduces the various options for setting and configuring reports.

### Accessing report settings

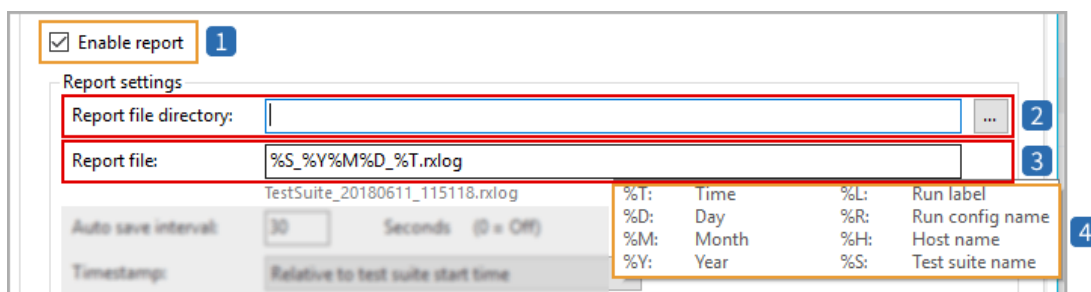
Report settings are accessed through the **Property** pane of the corresponding test suite.

- 1 Select the **test suite** and open the context menu
- 2 Click **Properties** in the context menu
- 3 In the settings window, click the **Report** register tab



## Report file & directory settings

Report file & directory settings are changed in the top of the report settings dialog.



- 1 Option to **enable/disable** the generation of reports
  - If de-selected, no report (files) will be created
- 2 Option to select an individual **report file directory** for storing Ranorex report files
  - If a folder is selected here, this folder is replacing the former standard report directory
- 3 Option to design/create the **automated standard report file name**
- 4 Set of **file name parts** for automated report file names (set available as tool-tip)



## Further reading

The concept of report file names and report directories is introduced and explained in detail in > Ranorex Studio fundamentals > Reporting > → [Ranorex standard reporting](#).

## Timestamp settings

Setting the 'Auto-save-interval' and the type of report timestamps is done in the report settings.

The screenshot shows a settings dialog with two main sections. The first section, labeled 'Auto save interval:', contains a text input field with the value '30', followed by the text 'Seconds (0 = Off)'. To the right of this section is a blue square with the number '5'. The second section, labeled 'Timestamp:', contains a dropdown menu with four options: 'Relative to test suite start time' (selected), 'Relative to test suite start time', 'Relative to test module start time', and 'Wall clock (machine time)'. To the right of this section is a blue square with the number '6'. Below the 'Timestamp' section, there are two unchecked checkboxes: 'Create a compressed copy of the report' and 'Create a JUnit compatible copy of the report'.

5

Option to set the **auto save interval**. Timing units are seconds. If set to '0', auto-save is turned off

6

Definition of **timestamp** in reports

## Enabling compressed reports

The creation of compressed report copies can be set in the report settings dialog.

The screenshot shows a settings dialog with a 'Timestamp:' dropdown menu set to 'Relative to test suite start time'. Below this, there are two unchecked checkboxes: 'Create a compressed copy of the report' and 'Create a JUnit compatible copy of the report'. The first checkbox is highlighted with a red rectangle.

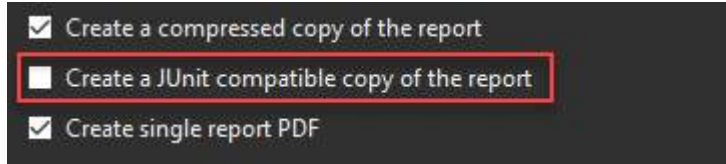


## Further reading

The creation of compressed reports is introduced and explained in detail in > Hands-on application topics > Best practices > → [Creating a compressed Ranorex report](#).

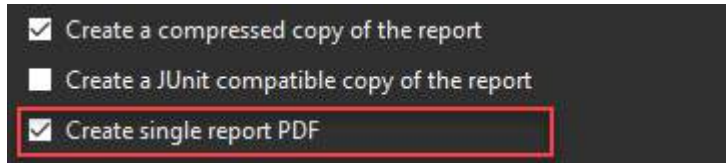
## Enabling JUnit-compatible reports

In Ranorex 7.0 and above, JUnit-compatible reports for use in CI tools can automatically be created.



## Enabling PDF Reports Merger

Ranorex Studio combines all the created reports by Ranorex test run into a single PDF file. The PDF includes the following report types: \*.rxlog, \*.rxlog.data, \*.xsl, \*.css, \*.png, images, and links to videos.



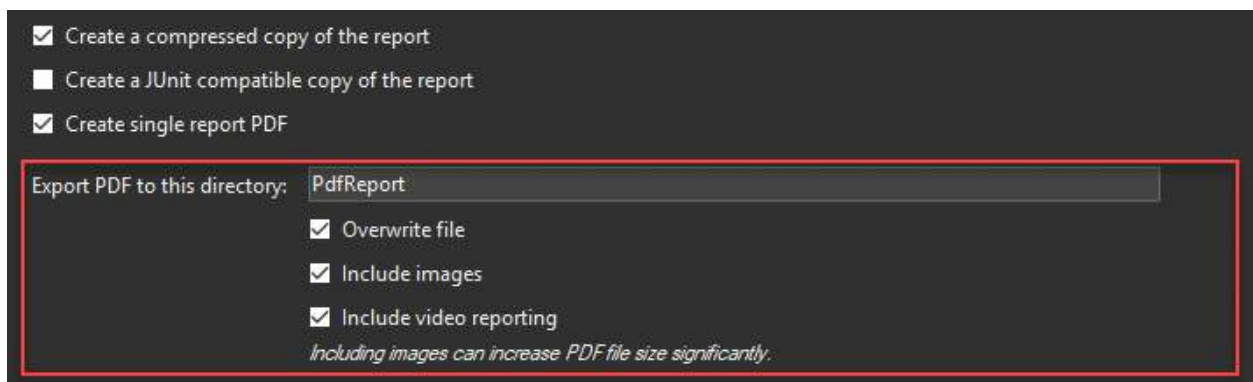
Enabling **Create single report PDF** checkbox display the following options to export a PDF:

- Export PDF to this directory

After setting up the PDF export directory, enable the following options according to your needs:

- Overwrite file
- Include images
- Include video reporting

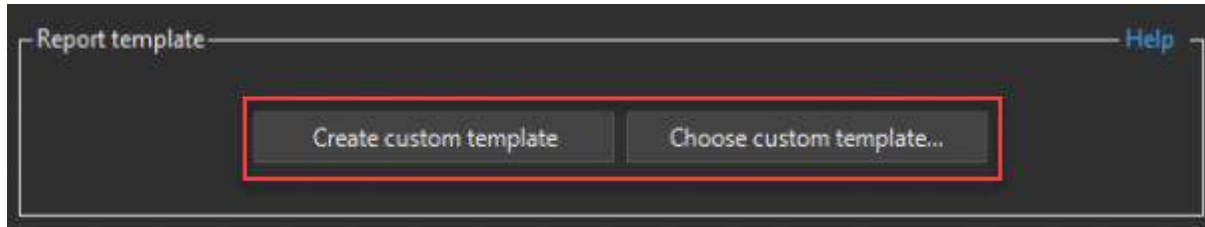
The above options are not enabled by default.





## Setting report templates

The various options for defining, applying and changing report templates can also be set in the report settings dialog.

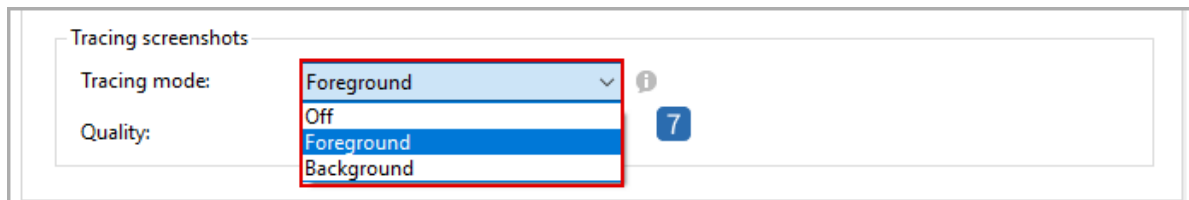


### Further reading

The concept of report customization by means of report templates is introduced and explained in > Ranorex Studio fundamentals > Reporting > [Report customization](#).

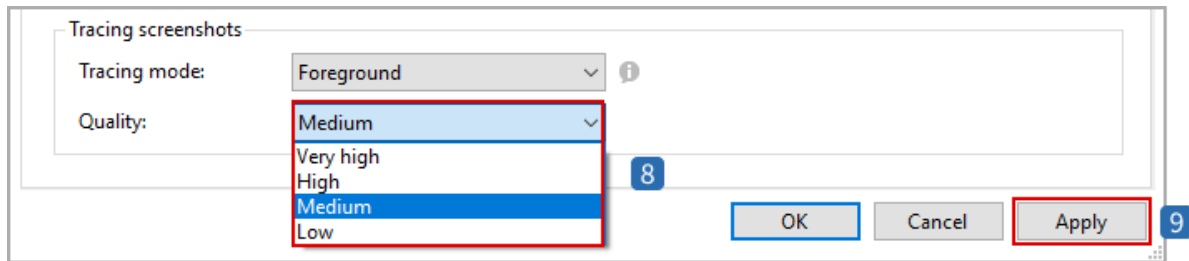
## Tracing screenshot settings

The settings for tracing screenshots in reports are to be defined in the lower part of the settings dialog.



**7** The settings for the **tracing mode** are:

- **Foreground:** Tracing screenshot is captured **before** the corresponding action is executed (more accurate, but slower performance)
- **Background:** Tracing screenshot is captured **while** action is being executed (less accurate, better performance)
- **Off:** No tracing screenshots are captured

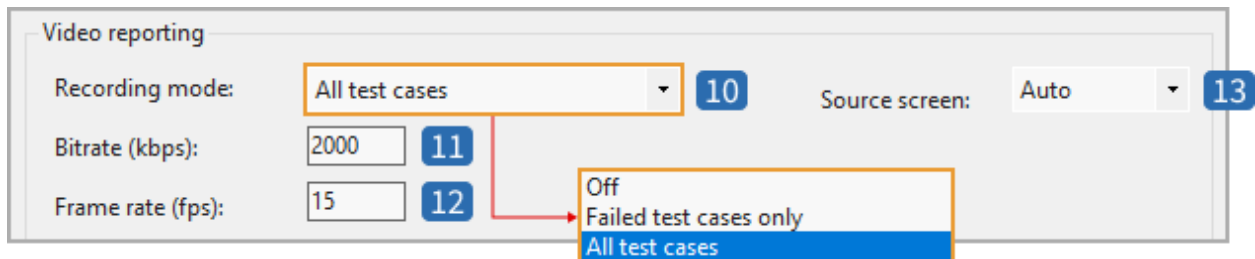


**8** Quality & size of tracing screenshot files

**9** Button **Apply** must be clicked in order to apply changed settings

## Video reporting

Video reporting lets you record a test run as a collection of videos. This also works on Ranorex Agents.



**10** Set the recording mode:

- Off: Video reporting is disabled. No videos will be recorded.
- Failed test cases only: Records the entire test run, but saves videos for failed test cases only.
- All test cases: Records and saves videos for all test cases.

### **i** Note

Ranorex Studio records a separate video for each test case. For the purpose of video reporting, test case means:

- Test cases
- Setup/teardown regions outside of test cases

**11** The bitrate of the videos in kilobytes per second. Lower values make video files smaller, but also lower image quality.

- 12 The frame rate of the videos in frames per second. Lower values make video files smaller, but fewer frames also mean less information.
- 13 Select the screen you want Ranorex Studio to record. The screen numbers correspond to the display settings in Windows. **Auto** uses the screen the mouse is on at the start of the test run.

#### Note

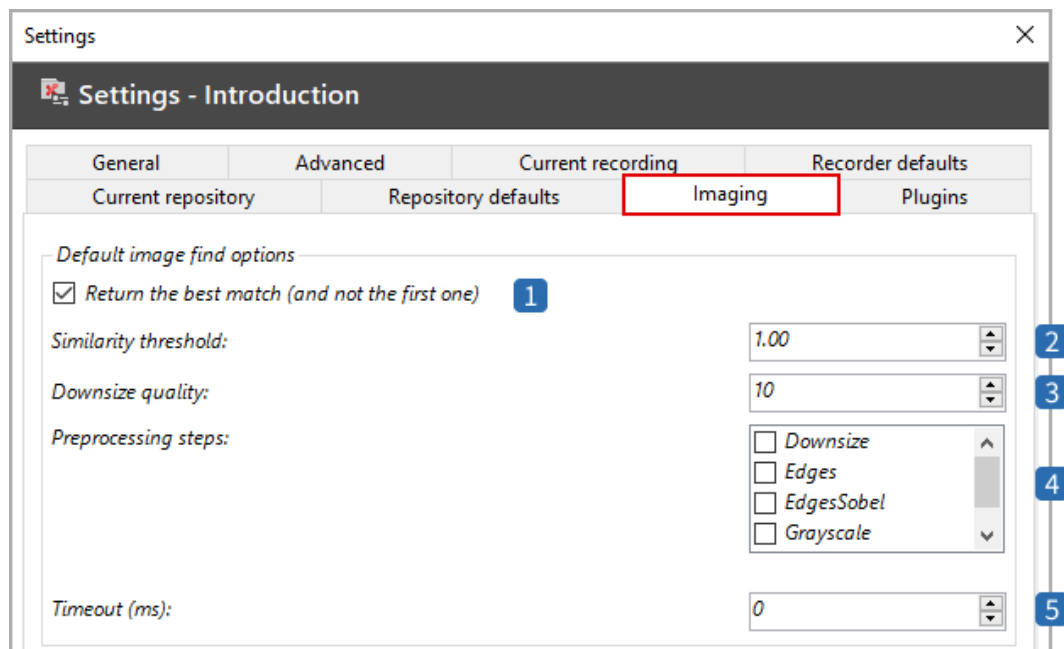
If you only have three screens and select screen 4, Ranorex Studio will automatically record the primary screen as defined in the display settings in Windows.

## Image-based automation settings

Image-based automation settings are to specify the behavior of Ranorex in detecting and identifying UI-elements in an image-based way. The settings can either be made generally (by means of the settings dialog) or on an action-based way (by means of the action properties).

### Image-based settings

To set up the default values for image-based recording, open up the '**Settings**' dialog and continue with activating the '**Imaging**' tab.



- 1 **Return the best match (and not the first one):** If 'Best Match' is set to true, the result position with the highest similarity is used for validation. If is set to false, the first available result position will be used for the validation. The first is more accurate and the second is faster.
- 2 **Similarity threshold (0.0-1.0):** Specifies the **minimum** similarity (0.0-1.0), that the image region to search for needs to have in common with the image in order to be considered a match
- 3 **Downsize quality**
- 4 **Preprocessing steps:** Defines preprocessing steps that can be performed on an image before search. Available options are 'Downsize', 'Edges', 'EdgesSobel', 'Grayscale' and 'Threshold'



### Further reading

The similarity threshold and the preprocessing steps are part of the image-based properties which are introduced and explained in detail in > Ranorex Studio advanced > Image-based automation > [Image-based properties](#).

## 5 Timeout (ms)

### Default validation options

These options control the default reporting values for image-based validations (ContainsImage and CompareImage).

You can set them for individual validations in the respective validation action's properties.



- 6 **Report difference image:** If and when an image showing the differences between the expected and the actual image will be included in the report. Default setting = **Never**.
- 7 **Report screenshot:** If and when a screenshot of the validated UI element will be included in the report. Default setting = **Never**.
- 8 **Report similarity:** If and when the similarity of the expected and the actual image will be included in the report. Default setting = **Never**.
- 9 **Report expected and actual images:** If and when the expected and the actual image will be included in the report. Default setting = **On Fail**.

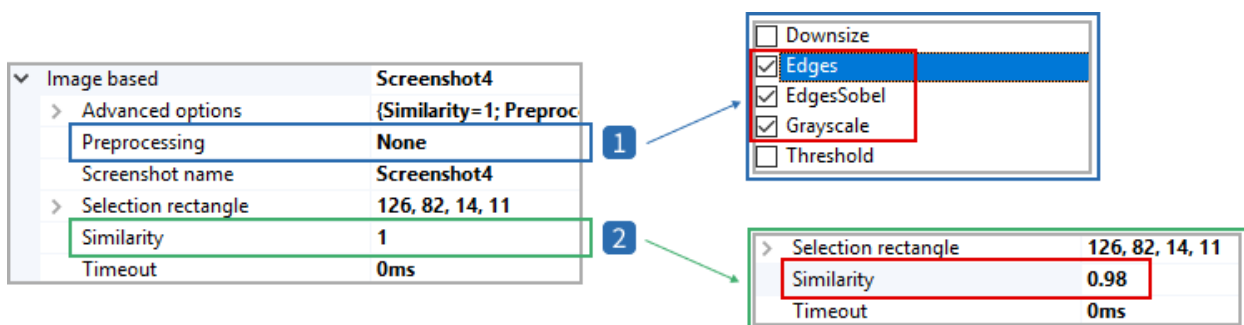
## Image-based properties

Image-based properties are available on an action-based access by selecting an action and pressing the **F4** key. Remember that image-based properties only affect the corresponding action and not the image-based behavior of Ranorex generally.



- 1 Select an action and press **F4** to open the action properties
- 2 See the **image-based** properties in the action properties pane

You find the same settings and configurations as in the image-based settings dialog.



- 3 See the **preprocessing** settings
- 4 See the **similarity factor** setting



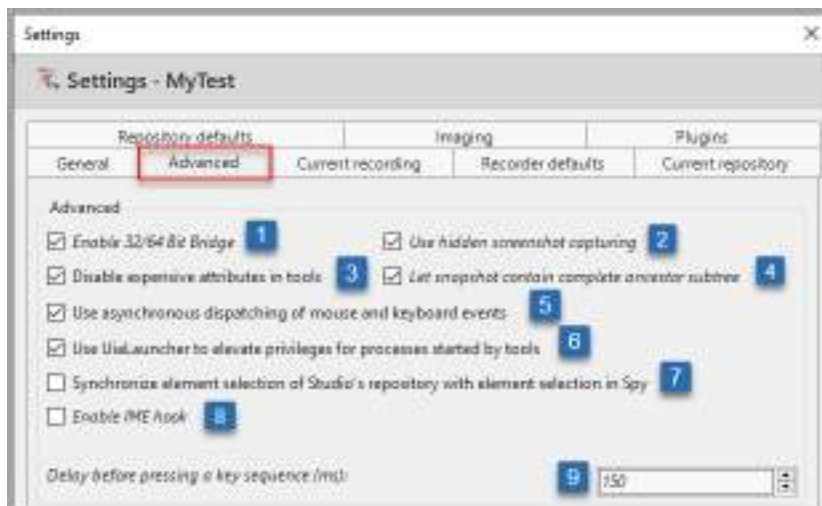
## Reference

Image-based automation is an advanced concept which is introduced and explained in detail beginning in > Ranorex Studio advanced > Image-based automation > → [Introduction](#).

## Advanced settings & configurations

Most of the settings shown in the '**Advanced**' tab are used to configure Ranorex object recognition and RanoreXPath generation. Please be careful when changing these settings.

### Advanced



- 1 **Enable 32/64 Bit Bridge:** Use the checkbox 'Enable 32/64 Bit Bridge' to turn off the bit bridge required to handle 32/64 bit based applications on 64 bit operating systems automatically.
- 2 **Use hidden screenshot capturing (if possible):** Tries to capture screenshots used by recordings, repositories or Ranorex snapshot files, even from application windows which are not in the foreground. If disabled, Ranorex tries to make the application window visible before capturing a screenshot.

- 3 Disable expensive attribute in tools:** This setting instructs plug-ins not to evaluate computationally intensive attributes for Ranorex Spy, Recorder, and Ranorex Studio. If checked, attributes like `Row . Index` do not have a value for certain technologies when shown in Ranorex Spy. This improves performance in some cases.
- 4 Let snapshot contain complete ancestor subtree:** When checked, the subtree containing the whole application will be stored when saving a snapshot file. If not checked, only the direct subtree to the selected item will be stored.
- 5 Use asynchronous dispatching of mouse and keyboard events:** This setting is used to turn on or off asynchronous dispatching of mouse and keyboard events during recording. With enabled asynchronous dispatching mouse and keyboard events will not be forwarded to an application before Ranorex has finished processing the event.
- 6 Use UiaLauncher to elevate privileges for processes started by tools:** Specifies whether test execution is started with elevated privileges or not.
- 7 Synchronize element selection of Studio's repository with element selection in Spy:** Specifies whether a UI-element will be automatically selected in Ranorex Spy when the representing repository item will be selected or not.
- 8 Enable IME Hook:** This option enables input support for complex characters, e.g. for Chinese and Japanese. **Note:** Does not work for Windows Store apps, WPF applications, and in Microsoft Edge, Internet Explorer, or Mozilla Firefox.
- 9 Delay before pressing a key sequence (ms):** Specifies the time to wait in milliseconds before performing a key sequence simulation.

## RanoreXPath settings

These settings allow you to change the behavior of automatic RanoreXPath generation.

### Speed/robustness slider

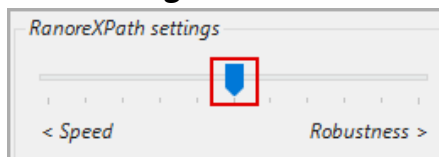
This slider controls how Ranorex Studio balances speed (flexibility; faster execution times) and robustness (more resistance to UI changes, slower execution times) when generating RanoreXPaths.

#### Note

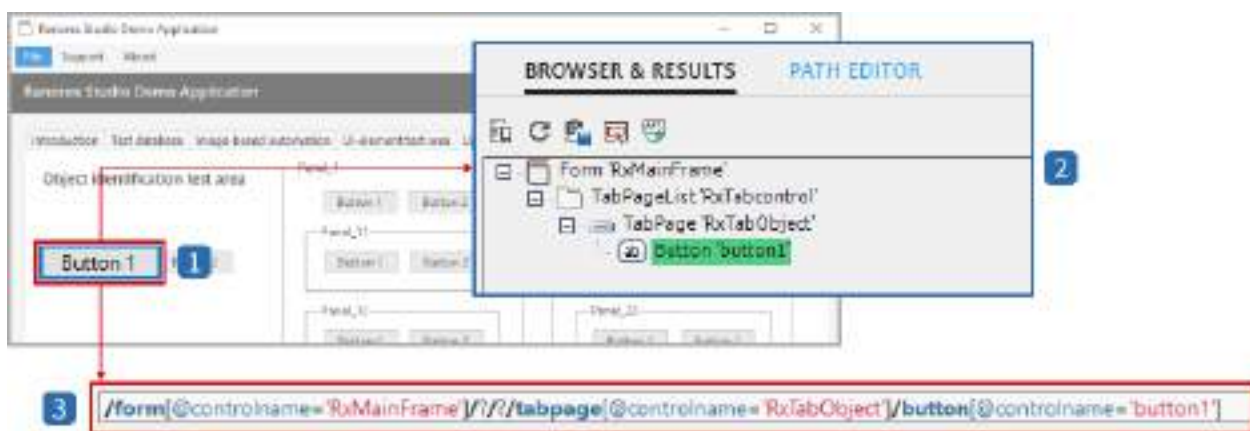
Setting the slider to between 30 % and 50 % achieves the best results for generating RanoreXPaths that are well balanced between speed and robustness.

Let's take a look at how setting the slider to different positions influences RanoreXPath generation.

### 50 % setting – balanced

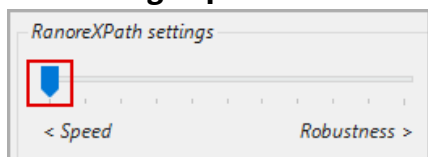


At the default setting of 50 %, Ranorex Studio generates a RanoreXPath for the button in the Ranorex Studio Demo Application that is equally balanced in terms of execution speed and resistance to UI changes.



- 1 Button in the Ranorex Studio Demo Application
- 2 The button in the element tree in Ranorex Spy. It is located on the fourth level of the tree.
- 3 The generated RanoreXPath for the button. It contains two ? wildcards between the root node and the parent of the UI element, which give the path a reasonable amount of flexibility.

### 0 % setting – speed



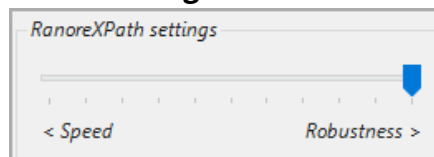
At 0%, Ranorex Studio generates a path that is focused on achieving the highest execution speed. This results in a very fast, definitive path that has the least resistance against UI changes.





- 1 Button in the Ranorex Studio Demo Application
- 2 The button in the element tree in Ranorex Spy. It is located on the fourth level of the tree.
- 3 The generated RanoreXPath for the button. It does not contain any wildcards. This makes the path fast, but means that changes in the UI or its structure will likely prevent it from successfully identifying the targeted UI element.

### 100 % setting - robustness

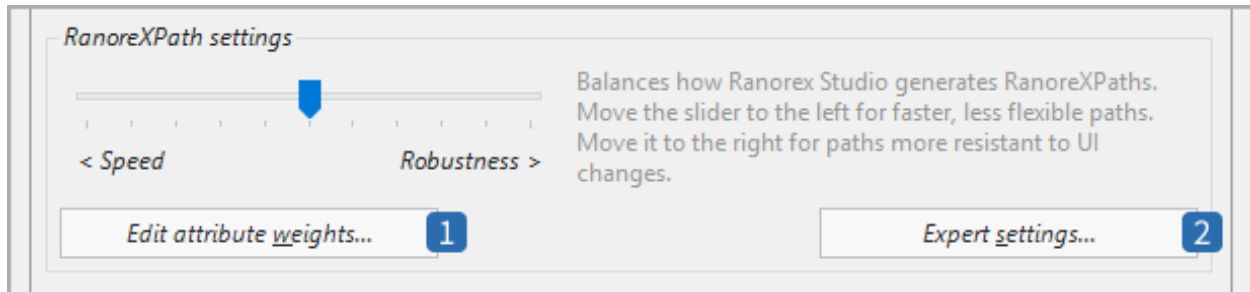


At 100%, Ranorex Studio generates a path that is focused on achieving the highest robustness. This results in a slower, flexible path that has the most resistance against UI changes.



- 1 Button in the Ranorex Studio Demo Application
- 2 The button in the element tree in Ranorex Spy. It is located on the fourth level of the tree.
- 3 The generated RanoreXPath for the button. It consists only of the root node, a // wildcard, and the final node. This means that any number of other nodes can be between the root and the final node. This makes the path slower, but very resistant to UI changes.

### Attribute weights and Expert settings



- 1 Edit attribute weights... lets you edit how automatic RanoreXPath generation weighs UI element attributes, i.e. which of them it prefers using over others.



### Reference

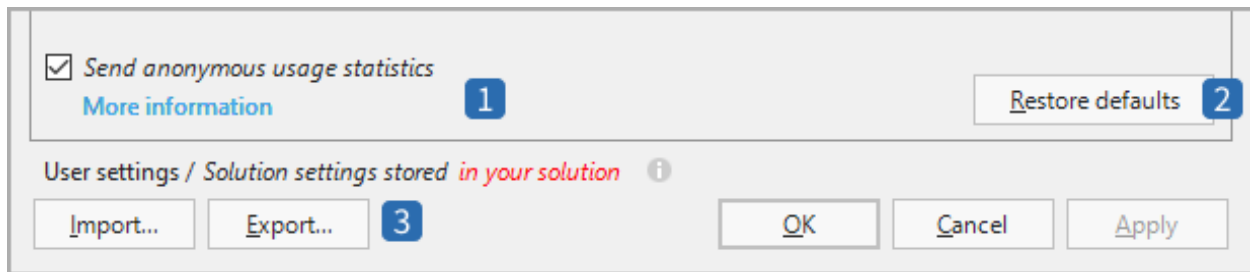
Attribute weights and other weight rules are explained in:

Ranorex Studio expert > → [Mapping dynamic UI elements](#)

- 2 Expert settings contains critical settings that change core aspects of how Ranorex Studio's algorithm generates RanoreXPaths.

Under these settings, you can enable the Robust RanoreXPath generation mode. For doing so, follow the instructions on the [Self-healing](#) page.

## Statistics, defaults, import/export



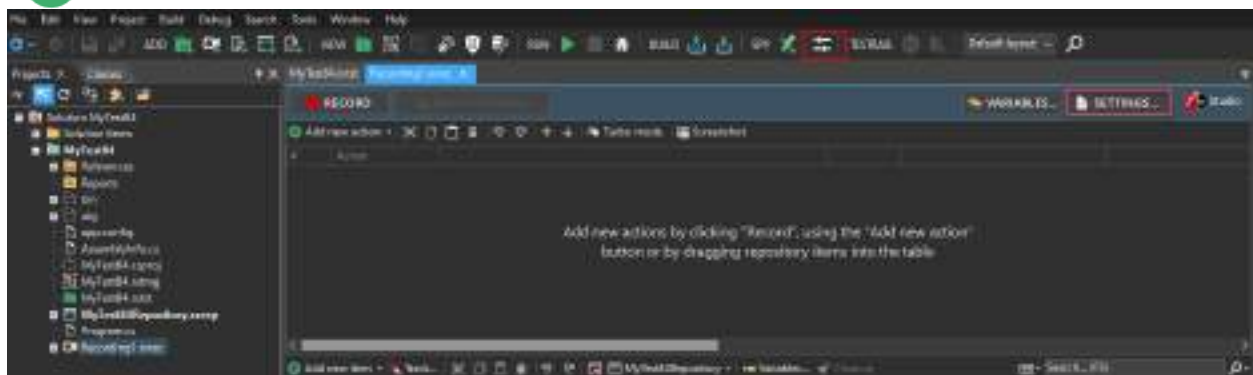
- 1 Checkbox for agreeing to send us anonymous usage statistics to improve Ranorex Studio. Default is checked. Click the button for more information about what data we use and how we use it.
- 2 Resets all advanced settings to their defaults.
- 3 Allows you to import settings from other solutions or export them for use in other solutions.

## Self-healing

This feature helps to reduce the “Failed to find item” error when running your automated test. Self-healing overcomes this issue by adding an additional robust layer of object recognition to avoid exceptions and reduce manual maintenance

### Enable Self-healing

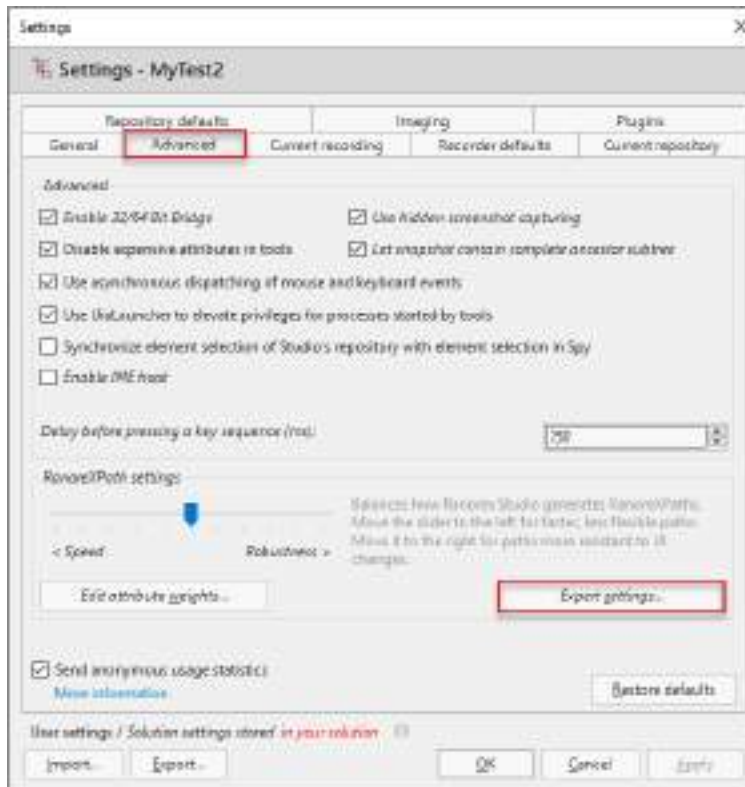
- 1 In the top toolbar, **click SETTINGS...**



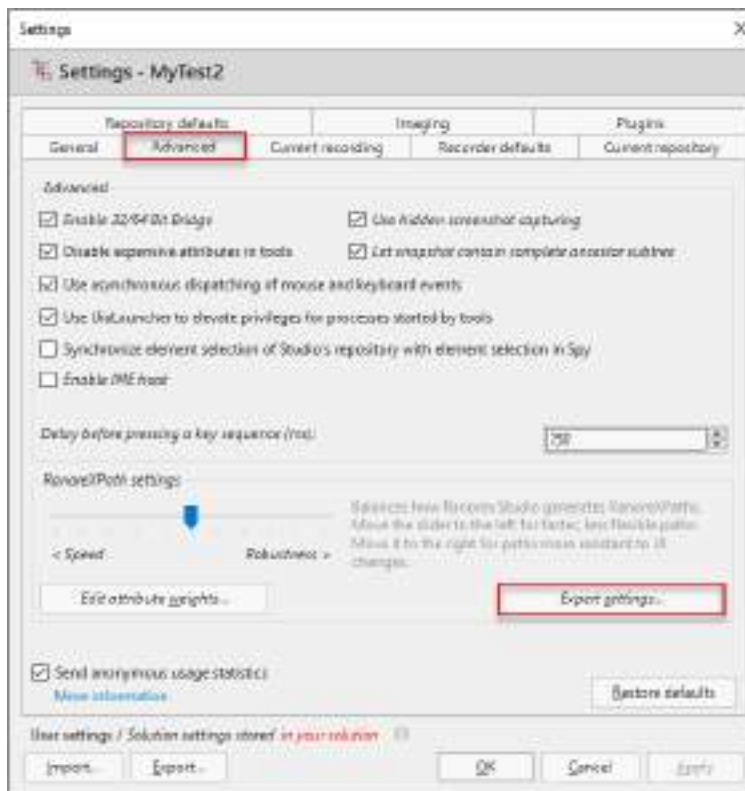
- 2 When the Settings window opens, **select the Advanced tab**, and **click Expert settings...**

**Note**

Review [Advanced settings and configurations](#) for more information about RanoreXPath settings.



- 3 In the Expert path generation settings window, **select Robust RanoreXPath (new)** from the **RanoreXPath generation mode** drop-down list.



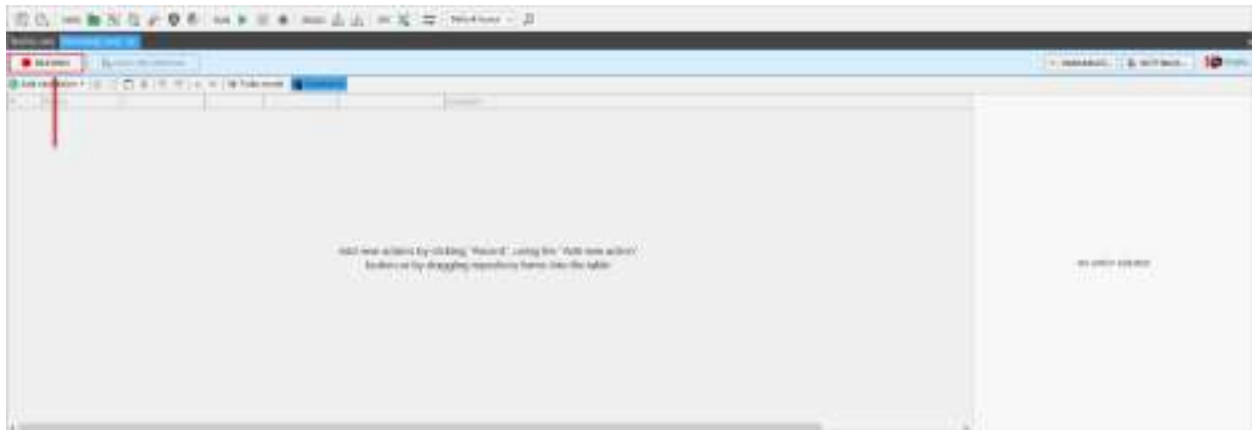
4 Click Apply and OK.

## Record your test

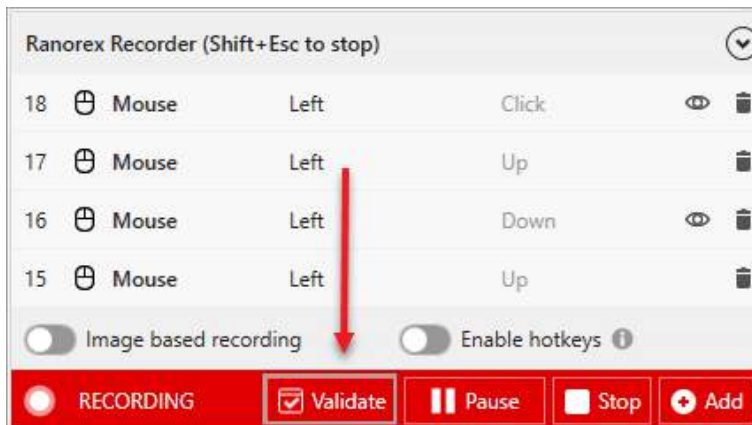
Before the self-healing option, when a test failed to find an object, an error displayed in the test log and reported a failed status in the test. The issue had to be solved, to re-run and complete these tests.

The self-healing option finds a similar or the closest object to the missing one in the system while running a test. All the objects are checked based on the attributes that RanoreXPath uses, for more information review [Mapping dynamic UI elements](#).

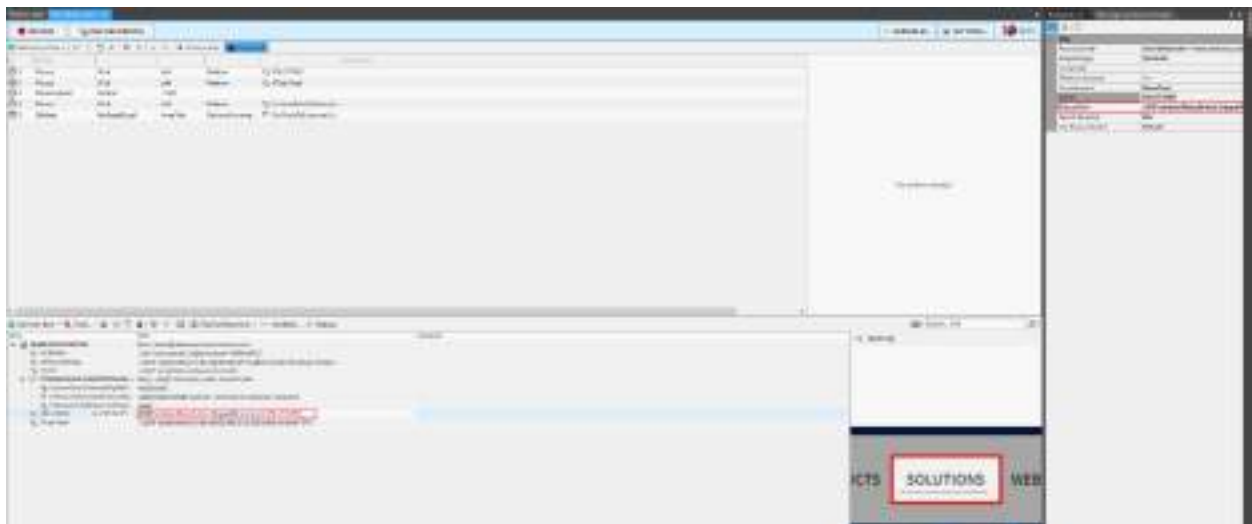
1 After you enabled the self-healing option, start recording your actions by **clicking RECORD**.



- 2 As soon as you complete your recording, **click Validate** located in the Ranorex Recorder window.

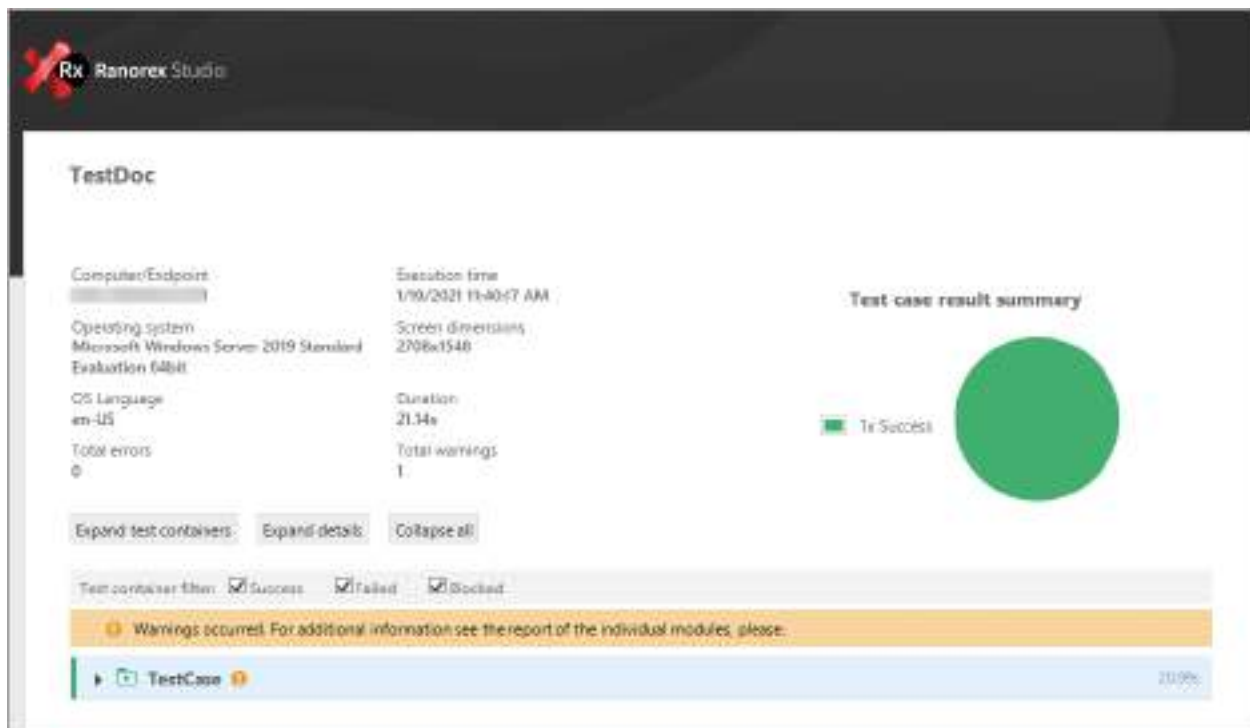


- 3 Once the validation is completed, **click Stop** to finish your recording. As a result, the recording opens in Ranorex Studio.
- 4 In your items section, expand the ApplicationUnderTest. When you select a sub-item, note the **RobustPath** under the **Properties** section located on the right side window



5

Click **RUN** to start running your test. The self-healing option allows the test to find an appropriate object and continue to run without interruptions or errors.



## Plugin-specific settings

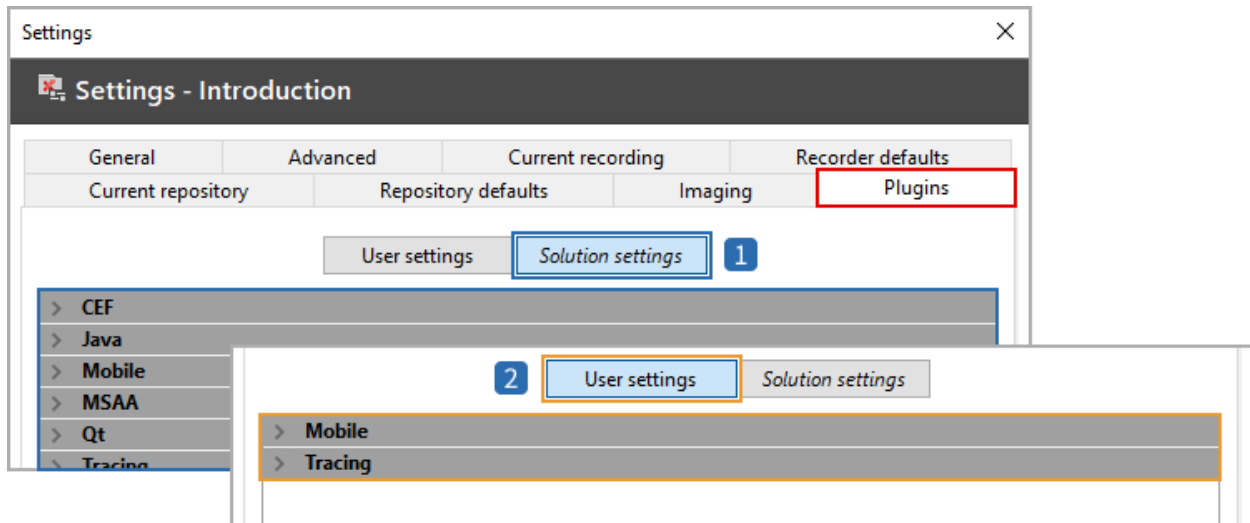
Plugin-specific settings can be used to alter the behavior of individual Ranorex plugins, for example, to achieve backward compatibility with older versions. Plugin-specific settings are grouped in **User settings** and **Solution settings**.



### Attention

Most of these settings affect Ranorex Studio's object recognition capabilities. Changes can easily **lead to test failures**. Only change these settings if you've been instructed to do so by Ranorex support or official documents.

Settings that do not affect object recognition or are generally safe to use are marked with an asterisk \*. It's ok to change them.



The plugin-specific settings are accessible through the settings dialog and the tab page **Plugins**. The plugin-specific settings are grouped as follows:

- 1 **Solution settings**
- 2 **User settings**

### **CEF (solution setting)**

#### **Process name black list**

This list contains process names (without the file extension) which should be ignored by the CEF plugin. If a process should be ignored by the CEF plugin, add it to the list in a new line.

### **Delphi (solution setting)**

#### **Enabled**

Determines if the Delphi plugin is enabled.

### **Java (solution setting)**

#### **Enable filtering on the Fx scenegraph**

Filters dummy and intermediate containers out of the Fx scenegraph.



### **Show SWT custom data properties**

Includes SWT custom Widget Data values as dynamic properties.

### **Use Java SWT legacy automation mode**

Enable Java SWT legacy automation mode using MSAA and Win32

### **Whitelisted class names**

Adds additional whitelisted class names (comma separated) for java object recognition.

### **Mobile (solution setting)**

#### **Android OS Automation**

Enables automation on Android OS screens. Disable to increase performance.

#### **\* Auto-reconnect attempts**

When the connection to a device is lost Ranorex tries to reconnect automatically. With this parameter, you can specify how many reconnect attempts should be made until the device will be set to the 'Error' state. If you don't want to reconnect automatically, set this parameter to 0.

#### **\* Connect timeout**

After this time span, the connection attempt will be aborted if the device is not reachable.

#### **\* Deploy timeout**

Sets the timeout for deploying a mobile app. The required time depends on your application size and the deploy method. Deploying an app takes longer when deploying via a network.

### **Devices**

This is used internally by the Ranorex system and is not user editable.

#### **\* General timeout**

The timeout for short running background processes. This timeout is used for very short running processes (less than 10 seconds in general). If you get a lot of timeout exceptions,

that does not occur during special operations like instrumentation or deploying, try to increase this value.

### **\* Group devices by**

Determines how devices are grouped in a dialog.

### **\* Instrument timeout**

Sets the timeout for instrumentation operations. A good initial value is 2 minutes. For very large applications the time required for instrumentation can take up to 10 minutes depending on the machine power.

### **iOS: Maximum number of invisible children**

- **-1** will fetch all cells of UITableView and UICollectionView views. Use this setting only, when all views have a relatively small number of cells and static/finite data sources.
- **0** will disable fetching of invisible cells. Generally, you should use this setting, when data in your tables is loaded dynamically e.g. from successive web requests.
- **# (e.g. 50)** will fetch all visible cells and the first 50 cells surrounding the visible cells area.

### **Java Runtime installation path**

The path of the currently used Java Runtime Environment (JRE) installation required for Android automation.

### **\* Network discovery timeout**

Specifies for how long Ranorex sends UDP broadcasts to find devices that can be used for automation. Normally, a device should respond within a few seconds.

### **Prompt on IP – address change**

If set to true, Ranorex will try to check if the IP-address of a device has changed since the last usage and will show a message if so. By setting this parameter to false you can disable this check.

### **\* Remote call timeout**

The time Ranorex waits for a remote procedure call response until it assumes a connection loss if no response was received.

### **\* Screenshots on AndroidOS**

Captures screenshots on AndroidOS screens. Disable to improve test execution performance.

### **\* Sort devices by**

Determines how devices are sorted in a dialog.

### **\* USB discovery timeout**

The timeout for discovering USB connected devices.

## **Mouse (solution setting)**

### **Mouse move interpolation mode**

Interpolation modes for the mouse cursor moving between start and end points (Mouse.Move(.)).

## **MSAA (solution setting)**

### **Evaluate computationally expensive attributes**

Setting this value to true will instruct the plugin to evaluate attributes that are expensive to compute and may result in longer delays when getting such attributes. Note that spying an element with such expensive attributes may then take considerable time.

### **Filter elements**

Setting this value to false will make all MSAA elements available without filtering, including elements being unavailable, invisible, or equivalent to elements of other flavors.

### **Filter Windows Forms elements**

Setting this value to false will make MSAA elements available that are direct children of Windows Forms elements and have the same role and screen rectangle as the parent element.

### **Filtering compatibility level**

If you need legacy RanoreXPaths (created with an older Ranorex version) to work with the current Ranorex version, set this property to the appropriate value matching the Ranorex version used to create the legacy RanoreXPaths.

## **Refine FromPoint results**

Setting this value to true results in an additional search operation for a better fitting element for every MSAA FromPoint operation (e.g. used by Element.FindFromPoint). This may produce better results if the MSAA FromPoint implementation of a control is broken.

## **\*Performance tracing (solution setting)**

This setting is explained in Ranorex Studio advanced > → [Performance tracing](#)

## **Qt (solution setting)**

### **Enable filtering on the QtQuick scenegraph**

Enables filtering of the QtQuick scenegraph by skipping or removing unnecessary layout, loader and style items.

### **Use QT legacy automation mode**

Enable Qt legacy automation mode using MSAA and Qt Accessibility.

## **UIA (solution setting)**

### **Enable debugging mode for Windows apps**

If set to true, the debugging mode for Windows apps will be enabled, causing them to not be suspended until the login session is closed (user logout). When set to false, the plugin will instead try to resume suspended apps when it needs to access them (possible race condition may cause freeze until the app is manually resumed).

### **Enable filtering of Windows app frames**

If set to true, the frames of Windows apps introduced with Windows 10 are filtered and all relevant elements of those frames are displayed as the child of the app element. This allows for transparent automation of Windows apps in windowed and full-screen mode.

### **Enumerate lists using the ItemContainerPattern**

Setting this value to true instructs the plugin to use the ItemContainerPattern to iterate items of virtual lists that implement this UI Automation pattern. Depending on the implementation of the control, iterating children using this pattern should also return items that are currently scrolled out of view, but might also be slower than the usual way to get child elements. Note

that switching this setting can render existing RanoreXPaths invalid for controls implementing the ItemContainerPattern.

### **Evaluate computationally expensive attributes**

Setting this value to true will instruct the plugin to evaluate attributes that are expensive to compute and may result in longer delays when getting such attributes. Note that spying an element with such expensive attributes may then take considerable time.

### **Force virtual items to be realized**

Forces virtual items to be realized when trying to get their child elements. Setting this value to true will allow searching lists with virtual items, but realizing may have undesired effects on the list depending on the list implementation, e.g. that the list is scrolled to make the realized item visible.

### **Provide elements for non-WPF windows natively implementing UIA**

Setting this value to true instructs the plugin to provide elements also for non-WPF windows which natively implement the UIA interface.

### **Subscribe to the UIA FocusChangedEvent**

If set to True, the plugin subscribes to the UI FocusChangedEvent, which may cause applications (like MS PowerPoint) to create more UIA elements and thus may improve object recognition, but may also negatively affect performance.

### **Web (solution setting)**

#### **Enable automation of embedded IE web documents**

Enable automation of Internet Explorer web documents embedded in other applications.

#### **WebDriver (solution setting)**

#### **Attach to open sessions**

Attach to existing WebDriver sessions to analyze and debug tests. Not recommended for high-latency environments. If the setting is disabled, tests/debug runs have to include an 'Open Browser' Action.

## **Browser Tab Automation**

Enable automation of multiple tabs in a browser. Not recommended for high-latency environments.

### **DOM fetch mode**

Can be set to 'Always' or 'WhenChanged'. Always: DOM will be updated at specific intervals. WhenChanged: Before DOM is fetched a call to check for changes is done. Only after DOM changes are detected, the whole DOM is pulled from the remote. This setting preserves bandwidth.

### **DOM update interval**

The timespan of the interval to update the DOM.

### **\* Screenshot update interval**

The timespan of the interval to update the screenshots. Will cause errors if too small.

### **Update screenshots after dom change**

Update screenshots after DOM changing operations (e.g. mouse clicks). Not recommended for high-latency environments.

### **\* WebDriver Command Timeout**

Maximum wait duration for a web driver command to return a result. If the call takes longer a timeout exception will be thrown and may set the endpoint into the error state. This parameter does only take effect after a program restart.

## **Win32 (solution setting)**

### **\*Black-listed process names**

Process names you enter here are blacklisted for object recognition, i.e. Ranorex Studio and its components will not be able to identify UI elements in these processes.

Processes entered here are ignored if there are entries in the whitelist or in the setting "White-listed process names" (see below).

## **Enable accessibility (MSAA) actions and attributes**

Specifies whether Win32 elements provide accessibility (MSAA) actions and attributes (as a Dynamic capability).

## **Enable basic Delphi support**

Enables support for basic Delphi controls such as textboxes, buttons, etc. Set to False for backward compatibility.

## **Use legacy Form role**

Enables 2.X legacy mode where many elements improperly had the Form role. Set to True for backward compatibility with 2.X paths.

## **\*White-listed process names**

Process names you enter here are whitelisted for object recognition, i.e. Ranorex Studio and its components will be able to identify UI elements **only** for these processes, all others will be ignored.

This setting shares entries with the → [whitelisting](#) feature and is a simpler version of it. Overrides entries in the setting Black-listed process names.

## **WPF (solution setting)**

### **Allow selected instance properties**

Extend the dynamic attribute list with entries for plain **.NET** properties. Each line specifies a full-type name and the attribute name, separated by a pipe, like **qW**

### **Disable WPF plug-in for processes**

Do not use the native WPF plug-in for any process specified in the list; keep using UIAutomation for WPF as in Ranorex version up to 5.2.

### **Enable WpfDebug capability**

Enable the WpfDebug capability for all WPF elements. This capability provides attributes and dynamic actions that are useful for analyzing an issue in the element tree.

## Ignore Attributes starting with

Reduce clutter in the list of dynamic attributes.

## Realize Items in Virtualizing Containers

Many WPF containers only show children which are also visible on the screen; to show all child-items, set this option to true. By default, this is false, as the performance impact for large grids can be very high.

## Show All Elements

Show the complete element tree, incorporating all visual and logical WPF elements. This option disables any other filtering and is useful for analyzing the structure of WPF applications when elements cannot be accessed.

## WPF Legacy/UIA Interaction

- Set to 'UiaOnly' to completely deactivate this plug-in, making all other settings obsolete.
- **WpfOnly**: Show only the native WPF plug-in tree, suppress UIA.
- **WpfPreferred**: Show both WPF and UIA tree, and return WPF elements for tracking.
- **UiaPreferred**: Show both WPF and UIA tree, and return UIA elements for tracking.
- **UiaOnly**: Do not use the native WPF plug-in at all, and keep using UIAutomation for WPF as in Ranorex versions up to 5.2.
- **WpfImprovedOnly**: Show only the improved native WPF plug-in tree, suppress UIA.



### Further reading

An improved WPF plugin (Ranorex 7.0+) is introduced and explained in > Interfaces & connectivity > Plugins > → [Improved WPF plugin](#).

## WPF Tree (solution setting)

### Always Show Visual Children

Always show the visual children for specific types, even if those types do also have one or more logical children. Enables accessing elements that are not part of the logical WPF element tree, like data-binding generated elements for well-known types. Example: "CurrentType|ParentType|ApparentParentType" Each entry is a pipe (the | character) separated tuple of type-names for (current, parent, apparent-parent) elements. An empty



field denotes match-all, e.g. “FrameworkElement||” matches for all parent and apparent-parent elements.

### **Never Show Visual Children**

Do not show the visual children for matching elements, even if those types do not have any logical children. Reduces unnecessary element tree branching. Example:

“CurrentType|ParentType|ApparentParentType” Each entry is a pipe (the | character) separated tuple of type-names for (current, parent, apparent-parent) elements. An empty field denotes match-all, e.g. “FrameworkElement||” matches for all parent and apparent-parent elements.

### **Skip Elements**

Neither show nor traverse elements for specific types. Reduces the size of the element tree by hiding types not used in UI testing. Example: “CurrentType|ParentType|ApparentParentType” Each entry is a pipe (the | character) separated tuple of type-names for (current, parent, apparent-parent) elements. An empty field denotes match-all, e.g. “FrameworkElement||” matches for all parent and apparent-parent elements.

### **Skip Elements but Descend to Children**

Reduce the hierarchy levels by always hiding redundant elements, and instead of continuing the tree with its children. By default, this skips over some common nested containers constructs. Example: “CurrentType|ParentType|ApparentParentType” Each entry is a pipe (the | character) separated tuple of type-names for (current, parent, apparent-parent) elements. An empty field denotes match-all, e.g. “FrameworkElement||” matches for all parent and apparent-parent elements.

### **Skip Elements but Descend to Single Child**

Reduce the hierarchy levels by hiding redundant containers, if they have none or only one child element. By default, this hides some layout-containers. Containers will be shown if they contain at least two children. Example: “CurrentType|ParentType|ApparentParentType” Each entry is a pipe (the | character) separated tuple of type-names for (current, parent, apparent-parent) elements. An empty field denotes match-all, e.g. “FrameworkElement||” matches for all parent and apparent-parent elements.

# System requirements

This chapter lists the hardware and software requirements for all versions of Ranorex Studio.

## Hardware requirements

The hardware requirements **apply to all versions of Ranorex Studio.**

### Minimum:

- **Processor:** 2 GHz dual core
- **Memory:** 1 GB
- **Disk space:** 1GB for Ranorex Studio plus required libraries

### Recommended:

- **Processor:** 2 GHz dual-core
- **Memory:** 4 GB
- **Disk space:** 1GB for Ranorex Studio plus required libraries

## Ranorex Studio 10.2

Supported operating systems (32- and 64-bit editions, except for Itanium-based systems):

- Windows 7 SP1  
(all updates installed)
- Windows 8.1
- Windows 10
- Windows 10 21H1
- Windows 11
- Windows Server 2008 R2 SP1
- Windows Server 2012
- Windows Server 2012 R2
- Windows Server 2016
- Windows Server 2019

## Software requirements

The following software requirements are needed to design tests in Ranorex Studio and use the full suite of tools (Spy, Recorder, License Manager). They are **automatically installed** when running the Ranorex Studio setup:

- Microsoft Windows Installer 3.1 or higher
- Microsoft Visual C++ 2017 x86
- Microsoft Visual C++ 2017 x64 (required for 64 bit Windows versions only)

Microsoft .NET Framework 4.8

## **Ranorex Studio 10.1.1 - 10.1.7**

Supported operating systems (32- and 64-bit editions, except for Itanium-based systems):

- Windows 7 SP1 (all updates installed)
- Windows 8.1
- Windows 10
- Windows 10 21H1
- Windows Server 2008 R2 SP1
- Windows Server 2012
- Windows Server 2012 R2
- Windows Server 2016
- Windows Server 2019

## **Software requirements**

The following software requirements are needed to design tests in Ranorex Studio and use the full suite of tools (Spy, Recorder, License Manager). They are **automatically installed** when running the Ranorex Studio setup:

- Microsoft Windows Installer 3.1 or higher
- Microsoft Visual C++ 2017 x86
- Microsoft Visual C++ 2017 x64 (*required for 64 bit Windows versions only*)
- Microsoft .NET Framework 4.8

## **Ranorex Studio 10.1**

Supported operating systems (32- and 64-bit editions, except for Itanium-based systems):

- Windows 7 SP1 (all updates installed)
- Windows 8.1
- Windows 10
- Windows 10 21H1
- Windows Server 2008 R2 SP1
- Windows Server 2012
- Windows Server 2012 R2
- Windows Server 2016
- Windows Server 2019

## **Software requirements**

The following software requirements are needed to design tests in Ranorex Studio and use the full suite of tools (Spy, Recorder, License Manager). They are **automatically installed** when running the Ranorex Studio setup:

- Microsoft Windows Installer 3.1 or higher

- Microsoft Visual C++ 2017 x86
- Microsoft Visual C++ 2017 x64 (*required for 64 bit Windows versions only*)
- Microsoft .NET Framework 4.8
- Internet Explorer 11 or higher (with the latest patches and updates)

## Ranorex Studio 10.0

Supported operating systems (32- and 64-bit editions, except for Itanium-based systems):

- |  |   |
|--|---|
| <ul style="list-style-type: none"> <li>• Windows 7 SP1 (all updates installed)</li> <li>• Windows 8.1</li> <li>• Windows 10</li> </ul> | <ul style="list-style-type: none"> <li>• Windows Server 2008 R2 SP1</li> <li>• Windows Server 2012</li> <li>• Windows Server 2012 R2</li> <li>• Windows Server 2016</li> <li>• Windows Server 2019</li> </ul> |
|--|---|

## Software requirements

The following software requirements are needed to design tests in Ranorex Studio and use the full suite of tools (Spy, Recorder, License Manager). They are **automatically installed** when running the Ranorex Studio setup:

- Microsoft Windows Installer 3.1 or higher
- Microsoft Visual C++ 2017 x86
- Microsoft Visual C++ 2017 x64 (*required for 64 bit Windows versions only*)
- Microsoft .NET Framework 4.6.2 or higher
- Internet Explorer 11 or higher (with the latest patches and updates)

## Ranorex Studio 9.5

Supported operating systems (32- and 64-bit editions, except for Itanium-based systems):

- |  |   |
|--|---|
| <ul style="list-style-type: none"> <li>• Windows 7 SP1 (all updates installed)</li> <li>• Windows 8.1</li> <li>• Windows 10</li> </ul> | <ul style="list-style-type: none"> <li>• Windows Server 2008 R2 SP1</li> <li>• Windows Server 2012</li> <li>• Windows Server 2012 R2</li> <li>• Windows Server 2016</li> <li>• Windows Server 2019</li> </ul> |
|--|---|

## Software requirements

The following software requirements are needed to design tests in Ranorex Studio and use the full suite of tools (Spy, Recorder, License Manager). They are **automatically installed** when running the Ranorex Studio setup:

- Microsoft Windows Installer 3.1 or higher
- Microsoft Visual C++ 2017 x86
- Microsoft Visual C++ 2017 x64 (*required for 64 bit Windows versions only*)
- Microsoft .NET Framework 4.6.2 or higher
- Internet Explorer 11 or higher (with the latest patches and updates)

## Ranorex Studio 9.4

Supported operating systems (32- and 64-bit editions, except for Itanium-based systems):

- Windows 7 SP1 (all updates installed)
- Windows 8.1
- Windows 10
- Windows Server 2008 R2 SP1
- Windows Server 2012
- Windows Server 2012 R2
- Windows Server 2016
- Windows Server 2019

## Software requirements

The following software requirements are needed to design tests in Ranorex Studio and use the full suite of tools (Spy, Recorder, License Manager). They are **automatically installed** when running the Ranorex Studio setup:

- Microsoft Windows Installer 3.1 or higher
- Microsoft Visual C++ 2017 x86
- Microsoft Visual C++ 2017 x64 (*required for 64 bit Windows versions only*)
- Microsoft .NET Framework 4.6.2 or higher
- Internet Explorer 11 or higher (with the latest patches and updates)

## Ranorex Studio 9.3

Supported operating systems (32- and 64-bit editions, except for Itanium-based systems):

- Windows 7 SP1  
(all updates installed)
- Windows 8.1
- Windows Server 2008 R2 SP1
- Windows Server 2012
- Windows Server 2012 R2

- Windows 10
- Windows Server 2016
- Windows Server 2019

## Software requirements

The following software requirements are needed to design tests in Ranorex Studio and use the full suite of tools (Spy, Recorder, License Manager). They are **automatically installed** when running the Ranorex Studio setup:

- Microsoft Windows Installer 3.1 or higher
- Microsoft Visual C++ 2017 x86
- Microsoft Visual C++ 2017 x64 (*required for 64 bit Windows versions only*)
- Microsoft .NET Framework 4.6.2 or higher
- Internet Explorer 11 or higher (with the latest patches and updates)

## Ranorex Studio 9.2

Supported operating systems (32- and 64-bit editions, except for Itanium-based systems):

- Windows 7 SP1  
(all updates installed)
- Windows 8.1
- Windows 10
- Windows Server 2008 R2 SP1
- Windows Server 2012
- Windows Server 2012 R2
- Windows Server 2016
- Windows Server 2019

## Software requirements

The following software requirements are needed to design tests in Ranorex Studio and use the full suite of tools (Spy, Recorder, License Manager). They are **automatically installed** when running the Ranorex Studio setup:

- Microsoft Windows Installer 3.1 or higher
- Microsoft Visual C++ 2017 x86
- Microsoft Visual C++ 2017 x64 (*required for 64 bit Windows versions only*)
- Microsoft .NET Framework 4.6.2 or higher
- Internet Explorer 11 or higher (with the latest patches and updates)

## Ranorex Studio 9.1

Supported operating systems (32- and 64-bit editions, except for Itanium-based systems):

- |  |     |   |
|--|-----|---|
| <ul style="list-style-type: none"> <li>• Windows 7 (all updates installed)</li> <li>• Windows 8.1</li> <li>• Windows 10</li> </ul> | SP1 | <ul style="list-style-type: none"> <li>• Windows Server 2008 R2 SP1</li> <li>• Windows Server 2012</li> <li>• Windows Server 2012 R2</li> <li>• Windows Server 2016</li> <li>• Windows Server 2019</li> </ul> |
|--|-----|---|

## Software requirements

The following software requirements are needed to design tests in Ranorex Studio and use the full suite of tools (Spy, Recorder, License Manager). They are **automatically installed** when running the Ranorex Studio setup:

- Microsoft Windows Installer 3.1 or higher
- Microsoft Visual C++ 2017 x86
- Microsoft Visual C++ 2017 x64 (*required for 64 bit Windows versions only*)
- Microsoft .NET Framework 4.6.2 or higher
- Internet Explorer 11 or higher (with the latest patches and updates)



### Attention

If you get errors when building your tests, please also install the [SDK for .NET Framework 4.6.2](#).

## Ranorex Studio 9.0

Supported operating systems (32- and 64-bit editions, except for Itanium-based systems):

- |   |     |   |
|---|-----|---|
| <ul style="list-style-type: none"> <li>• Windows7 (all updates installed)</li> <li>• Windows 8.1</li> <li>• Windows 10</li> </ul> | SP1 | <ul style="list-style-type: none"> <li>• Windows Server 2008 R2 SP1</li> <li>• Windows Server 2012</li> <li>• Windows Server 2012 R2</li> <li>• Windows Server 2016</li> <li>• Windows Server 2019</li> </ul> |
|---|-----|---|

## Software requirements

The following software requirements are needed to design tests in Ranorex Studio and use the full suite of tools (Spy, Recorder, License Manager). They are **automatically installed** when running the Ranorex Studio setup:

- Microsoft Windows Installer 3.1 or higher
- Microsoft Visual C++ 2017 x86
- Microsoft Visual C++ 2017 x64 (*required for 64 bit Windows versions only*)
- Microsoft .NET Framework 4.6.2 or higher
- Internet Explorer 11 or higher (with the latest patches and updates)

## Ranorex Studio 8.3

Supported operating systems (32- and 64-bit editions, except for Itanium-based systems):

- |  |                          |
|--|--------------------------|
| • Windows 7 SP1<br>(all updates installed) | • Windows Server 2008    |
| • Windows 8                                | • Windows Server 2008 R2 |
| • Windows 8.1                              | • Windows Server 2012    |
| • Windows 10                               | • Windows Server 2012 R2 |
|  | • Windows Server 2016    |

## Software requirements

The following software requirements are needed to design tests in Ranorex Studio and use the full suite of tools (Spy, Recorder, License Manager). They are **automatically installed** when running the Ranorex Studio setup:

- Microsoft Windows Installer 3.1 or higher
- Microsoft Visual C++ 2017 x86
- Microsoft Visual C++ 2017 x64 (*required for 64 bit Windows versions only*)
- Microsoft .NET Framework 4.5.2 or higher
- Internet Explorer 11 or higher (with the latest patches and updates)

## Ranorex Studio 8.2

Supported operating systems (32- and 64-bit editions, except for Itanium-based systems):

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

## Software requirements

The following software requirements are needed to run all Ranorex components (Runtime, Studio, Spy, Recorder, License Manager, and Agent). They are **automatically installed** as part of the Ranorex runtime when running the Ranorex Studio setup:



- Microsoft Windows Installer 3.1 or higher
- Microsoft Visual C++ 2008 x86
- Microsoft Visual C++ 2008 x64 (*required for 64 bit Windows versions only*)
- Microsoft Visual C++ 2010 x86
- Microsoft Visual C++ 2010 x64 (*required for 64 bit Windows versions only*)
- Microsoft Visual C++ 2015 x86
- Microsoft Visual C++ 2015 x64 (*required for 64 bit Windows versions only*)
- Microsoft .NET Framework 4.5.2 or higher
- Internet Explorer 11 or higher (with the latest patches and updates)

## Ranorex Studio 8.0, 8.1

Supported operating systems (32- and 64-bit editions, except for Itanium-based systems):

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

## Software requirements

The following software requirements are needed to run all Ranorex components (Runtime, Studio, Spy, Recorder, License Manager, and Agent). They are **automatically installed** when running the Ranorex Studio setup:

- Microsoft Windows Installer 3.1 or higher
- Microsoft Visual C++ 2008 x86
- Microsoft Visual C++ 2008 x64 (*required for 64 bit Windows versions only*)
- Microsoft Visual C++ 2010 x86
- Microsoft Visual C++ 2010 x64 (*required for 64 bit Windows versions only*)
- Microsoft Visual C++ 2015 x86
- Microsoft Visual C++ 2015 x64 (*required for 64 bit Windows versions only*)
- Microsoft .NET Framework 4.5.2 or higher
- Internet Explorer 8 or higher (with the latest patches and updates)

## Ranorex Studio 7.1.2, 7.2

### Supported operating systems

Supported operating systems (32- and 64-bit editions, except for Itanium-based systems):

- |              |                       |
|--------------|-----------------------|
| • Windows XP | • Windows Server 2003 |
|--------------|-----------------------|

- Windows Vista
- Windows 7
- Windows 8
- Windows 8.1
- Windows 10
- Windows Server 2003 R2
- Windows Server 2008
- Windows Server 2008 R2
- Windows Server 2012
- Windows Server 2012 R2
- Windows Server 2016

## Software requirements

The following software requirements are needed to run all Ranorex components (Runtime, Studio, Spy, Recorder, License Manager, and Agent). They are **automatically installed** when running the Ranorex Studio setup:

- Microsoft Windows Installer 3.1 or higher
- Microsoft Visual C++ 2008 x86
- Microsoft Visual C++ 2008 x64 (*required for 64 bit Windows versions only*)
- Microsoft Visual C++ 2010 x86
- Microsoft Visual C++ 2010 x64 (*required for 64 bit Windows versions only*)
- Microsoft Visual C++ 2015 x86
- Microsoft Visual C++ 2015 x64 (*required for 64 bit Windows versions only*)
- Microsoft .NET Framework 4.0 or higher
- Internet Explorer 8 or higher (with the latest patches and updates)

## Ranorex Studio 7.1, 7.1.1

### Supported operating systems

Supported operating systems (32- and 64-bit editions, except for Itanium-based systems):

- Windows XP
- Windows Vista
- Windows 7
- Windows 8
- Windows 8.1
- Windows 10
- Windows Server 2003
- Windows Server 2003 R2
- Windows Server 2008
- Windows Server 2008 R2
- Windows Server 2012
- Windows Server 2012 R2
- Windows Server 2016

## Software Requirements

The following software requirements are needed to run all Ranorex components (Runtime, Studio, Spy, Recorder, License Manager, and Agent). They are **automatically installed** when running the Ranorex Studio setup:

- Microsoft Windows Installer 3.1 or higher
- Microsoft Visual C++ 2015 x86
- Microsoft Visual C++ 2015 x64 (*required for 64 bit Windows versions only*)
- Microsoft .NET Framework 4.0 or higher
- Internet Explorer 8 or higher (with the latest patches and updates)

## Ranorex Studio 6.1-7.0

### Supported operating systems

Supported operating systems (32- and 64-bit editions, except for Itanium-based systems):

- |                 |                          |
|-----------------|--------------------------|
| • Windows XP    | • Windows Server 2003    |
| • Windows Vista | • Windows Server 2003 R2 |
| • Windows 7     | • Windows Server 2008    |
| • Windows 8     | • Windows Server 2008 R2 |
| • Windows 8.1   | • Windows Server 2012    |
| • Windows 10    | • Windows Server 2012 R2 |
|                 | • Windows Server 2016    |

### Software requirements

The following software requirements are needed to run all Ranorex components (Runtime, Studio, Spy, Recorder, License Manager, and Agent). They are **automatically installed** when running the Ranorex Studio setup:

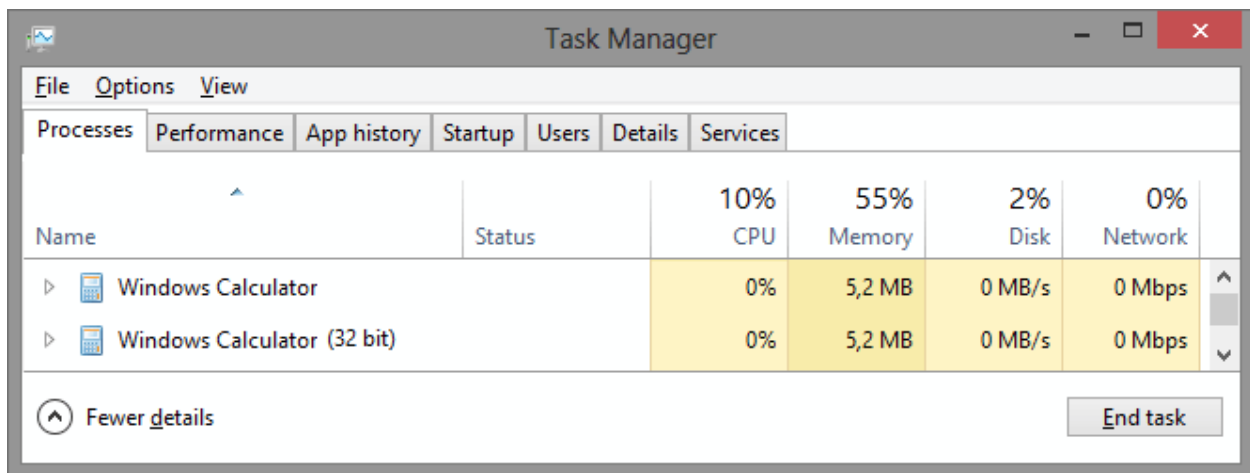
- Microsoft Windows Installer 3.1 or higher
- Microsoft Visual C++ 2008 x86
- Microsoft Visual C++ 2008 x64 (*required for 64 bit Windows versions only*)
- Microsoft Visual C++ 2010 x86
- Microsoft Visual C++ 2010 x64 (*required for 64 bit Windows versions only*)
- Microsoft Visual C++ 2015 x86
- Microsoft Visual C++ 2015 x64 (*required for 64 bit Windows versions only*)
- Microsoft .NET Framework 4.0 or higher
- Internet Explorer 8 or higher (with the latest patches and updates)

## 64-bit platforms

Ranorex Studio and all its tools (Spy, Recorder, Test Suite Runner) are **available in 32- and 64-bit versions**. Both versions of the tools are available from the start menu, but if you want to **manually start the 64-bit Ranorex Studio**, you need to do so from **`\Ranorex\Studio\Bin\x64\RanorexStudio.exe`**.

Ranorex Studio handles testing of 32/64 bit-based applications on 64-bit operating systems automatically. It is possible, but recommended only for advanced users, to turn off the bit bridge feature within the Ranorex Settings dialog. If you disable the bit bridge feature (not recommended!) or your Ranorex Studio version does not support the bit bridge (versions prior to 2.3), follow the guidelines in this chapter to make 32/64-bit automation interoperable.

On 64-bit versions of Windows, processes may run using 64-bit or 32-bit (also called 'x86') architecture. Applications that are started as 32-bit processes are marked with '\*32' or '(32 bit)' in the Windows Task Manager, all others use the 64-bit architecture.



In general, you should use the Ranorex Studio version that matches the bit architecture of the automated application. I.e. if you automate a 32 bit application, use Ranorex Studio (32bit) or the respective 32-bit tools, otherwise use the 64-bit versions of Ranorex Studio and its tools.

When you compile your own Ranorex executables, be sure to specify the right target architecture in your project settings. In Ranorex Studio this setting is on the 'Compiling' tab in the project properties. .NET applications are by default started as 64 bit processes on 64 bit operating systems unless the target CPU is explicitly set to the 32 bit (x86) architecture. Consequently, set the 'Target CPU' property to 'Any processor' for automating 64 bit applications and to '32-bit Intel-compatible processor' for 32 bit applications.

Advanced

☐ Register for COM Interop



Generate serialization assembly:

Auto

Target CPU:

32-bit Intel-compatible processor

File alignment:

Any processor (prefer 32-bit)

DLL base address:

Any processor (prefer 64-bit)

Base intermediate output path:

32-bit Intel-compatible processor

64-bit Intel-compatible processor

# Silent installation

Silent installation is an automated Ranorex method for installing Ranorex on different physical machines in a network. This allows system administrators to set up a Ranorex environment quickly and efficiently.

## Installation Packages

Ranorex uses the Microsoft/Windows Installer (MSI) for its setup. Consequently, all standard [MSI command line arguments](#) can be used. The command line arguments work both with the self-extracting zip file (**Ranorex-x.x.x.exe**) and the **setup.exe** and **Ranorex-x.x.x.x.msi** files contained in the **Ranorex-x.x.x.zip** file.

System administrators may use the Ranorex MSI package for installation. You can download the **Ranorex-x.x.x.zip** containing the MSI package from our homepage; the download link is the same as for the self-extracting zip file **Ranorex-x.x.x.exe**. Just replace the file ending **exe** with **zip**.

When installing Ranorex using the MSI package you have to make sure that all Ranorex prerequisites are installed before executing the MSI package (please see the **README.txt** included in the ZIP archive or the [→ Ranorex system requirements](#)). If you are unsure, please use the **setup.exe** or self-extracting zip file **Ranorex-x.x.x.exe** to start the installation, which will also install all required prerequisites.

## Installation Command Line Arguments

To install Ranorex silently, pass the **/quiet** (no UI) or **/passive** (progress bar only) command line argument:

**msiexec /i Ranorex-x.x.x.x.msi /quiet**

or: **setup.exe /passive**

or: **Ranorex-x.x.x.exe /quiet**

The silent installation will not work for any prerequisites (a message will pop-up asking whether you want to install the required components) due to legal limitations. If you want to make sure that all [→ required prerequisites](#) are installed silently as well, you need to install them separately, e.g. using a batch file.

If you do not want to install all Ranorex features, you can (de)select these features using the command line options **ADDLOCAL** and **REMOVE**.

For example, the following command line will install all Ranorex components except for Ranorex Studio:

**msiexec /i Ranorex-x.x.x.x.msi ADDLOCAL="ALL" REMOVE="RanorexStudioFeature"**  
or: **Ranorex-x.x.x.exe ADDLOCAL="ALL" REMOVE="RanorexStudioFeature"**

Possible feature names used as parameter to **ADDLOCAL** or **REMOVE** (separated by commas) are:

- 'MainFeature' (Core components)
- 'RanorexSamples'
- 'RanorexStudioFeature'
- 'RanorexFirefoxExtension'
- 'RanorexIEAddon'
- 'RanorexDocumentation'

For more installation options see the help for the MsiExec program by typing (/v is needed for the \*.exe files):

**msiexec /help**  
or: **setup /v /help**  
or: **Ranorex-x.x.x.exe /v /help**

## Install Ranorex License

To finish your silent installation you have to install a valid license.

If you would like to install a **floating license** you have to copy the file **Ranorex3\_Server.lic** (for Ranorex3.x installations) or **Ranorex2\_Server.lic** (for Ranorex 2.x installations) into the folder **%ALLUSERSPROFILE%** by using the **XCOPY** command in a batch file. The file will be created when you first install a License Server license on a Ranorex Client. Just search for that file on a client machine running a floating license.

If you would like to install a **node locked license** you have to generate a license file. The web address used to authenticate the license key is part of the licensing email delivered after purchasing licenses. Simply open a browser, navigate to the authentication page and enter your license key and the machine's host name into the respective fields. After clicking the **Authenticate** button you will be able to download the license file. Rename the downloaded file to **Ranorex3.lic** (for Ranorex3.x installations) or **Ranorex2.lic** (for Ranorex 2.x installations) and copy it to the folder **%ALLUSERSPROFILE%** by using the **XCOPY** command in a batch file.