

Altova StyleVision Server 2025



User & Reference Manual

Altova StyleVision Server 2025 User & Reference Manual

All rights reserved. No parts of this work may be reproduced in any form or by any means - graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems - without the written permission of the publisher.

Products that are referred to in this document may be either trademarks and/or registered trademarks of the respective owners. The publisher and the author make no claim to these trademarks.

While every precaution has been taken in the preparation of this document, the publisher and the author assume no responsibility for errors or omissions, or for damages resulting from the use of information contained in this document or from the use of programs and source code that may accompany it. In no event shall the publisher and the author be liable for any loss of profit or any other commercial damage caused or alleged to have been caused directly or indirectly by this document.

Published: 2025

© 2019-2025 Altova GmbH

Table of Contents

1	Introduction	7
2	Functionality	8
2.1	In the FlowForce Workflow.....	9
2.2	As a Standalone Server.....	10
3	Installation and Licensing	11
3.1	Setup on Windows.....	12
3.1.1	Install on Windows.....	12
3.1.2	Install on Windows Server Core.....	13
3.1.3	Install LicenseServer (Windows).....	15
3.1.4	Start LicenseServer, StyleVision Server (Windows).....	16
3.1.5	Register StyleVision Server (Windows).....	17
3.1.6	Assign License (Windows).....	17
3.1.7	Additional Setup Notes (Windows).....	18
3.2	Setup on Linux.....	20
3.2.1	Install on Linux.....	20
3.2.2	Install LicenseServer (Linux).....	22
3.2.3	Start LicenseServer, StyleVision Server (Linux).....	23
3.2.4	Register StyleVision Server (Linux).....	23
3.2.5	Assign License (Linux).....	24
3.2.6	Notes about Environment (Linux).....	24
3.3	Setup on macOS.....	27
3.3.1	Install on macOS.....	27
3.3.2	Install LicenseServer (macOS).....	28
3.3.3	Start LicenseServer, StyleVision Server (macOS).....	29
3.3.4	Register StyleVision Server (macOS).....	29
3.3.5	Assign License (macOS).....	30

3.3.6	Notes about Environment (macOS).....	31
3.4	Upgrade StyleVision Server.....	33
3.5	Migrate StyleVision Server to a New Machine.....	34
3.6	FOP Requirements.....	35
3.7	Security Considerations.....	36
4	StyleVision Server Command Line	37
4.1	accepteula (Linux only).....	39
4.2	assignlicense.....	40
4.3	exportresourcestrings.....	42
4.4	generate.....	44
4.5	help	49
4.6	licenseserver.....	50
4.7	pdfdata.....	52
4.8	setdeflang.....	54
4.9	setfopath.....	55
4.10	verifylicense.....	57
4.11	version.....	58
5	StyleVision Server API	59
5.1	About the .NET Interface.....	60
5.2	About the COM Interface.....	61
5.3	About the Java Interface.....	62
5.4	Code Examples.....	64
5.4.1	C#	64
5.4.2	C++	65
5.4.3	Java	67
5.4.4	VBScript.....	68
5.4.5	Visual Basic.....	70
5.5	API Reference.....	72
5.5.1	COM and .NET	72
5.5.2	Java	79

6	Schema Manager	84
6.1	Run Schema Manager.....	88
6.2	Status Categories.....	91
6.3	Patch or Install a Schema.....	93
6.4	Uninstall a Schema, Reset.....	95
6.5	Command Line Interface (CLI).....	96
6.5.1	help	96
6.5.2	info	97
6.5.3	initialize.....	97
6.5.4	install	98
6.5.5	list	98
6.5.6	reset	99
6.5.7	uninstall.....	100
6.5.8	update.....	101
6.5.9	upgrade.....	101
7	Taxonomy Manager	102
7.1	Run Taxonomy Manager.....	106
7.2	Status Categories.....	109
7.3	Patch or Install a Taxonomy.....	111
7.4	Uninstall a Taxonomy, Reset.....	113
7.5	Command Line Interface (CLI).....	114
7.5.1	help	114
7.5.2	info	115
7.5.3	initialize.....	115
7.5.4	install	116
7.5.5	list	116
7.5.6	reset	117
7.5.7	uninstall.....	118
7.5.8	update.....	119
7.5.9	upgrade.....	119

Index

120

1 Introduction

Altova StyleVision Server is an implementation of [Altova StyleVision's](#) built-in execution engine. It can be used as a [standalone server product](#)¹⁰ or as a module of Altova's [FlowForce Server](#)⁹.



StyleVision Server executes transformation packages that are stored on disk or that have been deployed to a [FlowForce Server](#)⁹:

- StyleVision Server functionality can be invoked [via the command line](#)³⁷ to run a transformation package and generate the files specified in the package.
- StyleVision Server transformations are initiated by [FlowForce Server](#) based on time triggers, file triggers, or remote triggers that are defined in FlowForce Server..

StyleVision Server is available for both 32-bit and 64-bit on Windows machines. For details about installation and licensing, see the setup sections for [Windows](#)¹², [Linux](#)²⁰, and [macOS](#)²⁷.

Note: If the fillable parts of a fillable PDF are missing when the PDF is opened on a macOS system, one likely cause is that Java 6 is not installed on the machine. If this is the case, you can install Java 6 from https://support.apple.com/kb/dl1572?locale=en_US. If a version newer than Java 6 has already been installed on your system, then the installation of the older Java 6 version will not affect the working of the newer version, which will be the default version of the system.

Last updated: 17 March 2025

2 Functionality

StyleVision Server transforms XML files into output HTML, PDF, RTF, and DOCX documents with the use of XSLT stylesheets. These XSLT stylesheets are obtained from PXF files that have been created in Altova's stylesheet designer application, [Altova StyleVision](#).

StyleVision Server can be used in two ways:

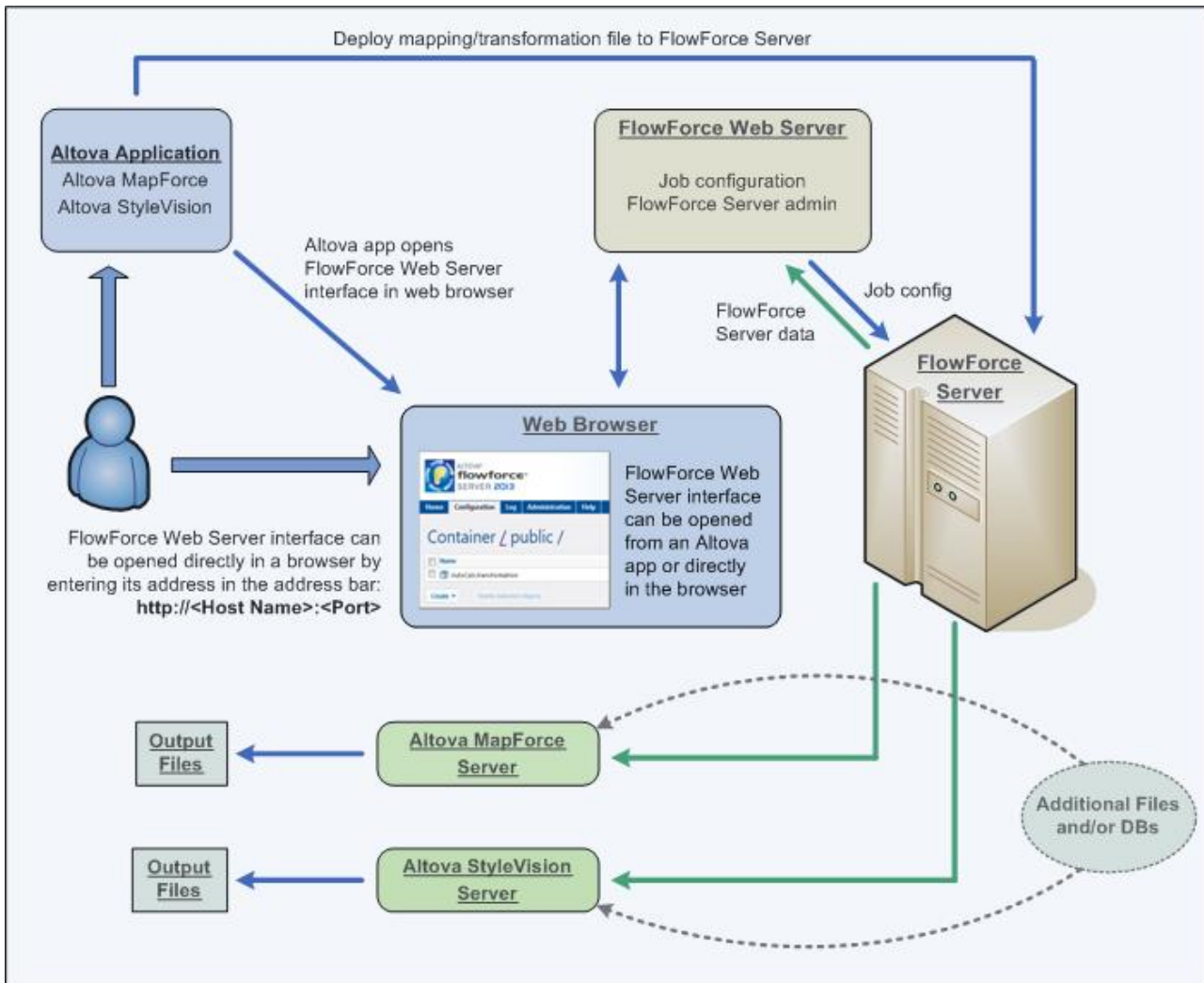
- As part of the [Altova FlowForce workflow](#)⁹. For more information about [Altova FlowForce](#), visit the [Altova website](#).
- [As a standalone server product](#)¹⁰ that is accessed via its command line interface (CLI).

An XML input file and a PXF file are passed to StyleVision Server, which produces the required output document/s.

2.1 In the FlowForce Workflow

A FlowForce job is created in [Altova FlowForce Server](#). The FlowForce job specifies: (i) the inputs and outputs of a StyleVision Server transformation; and (ii) the triggers for when the job is to be executed, such as a specific time every day. At execution time, Altova FlowForce Server passes the transformation instructions to StyleVision Server, which then carries out the transformation.

The role of StyleVision Server in the FlowForce workflow is shown in the diagram below. (The role of MapForce Server in the workflow is also displayed since FlowForce jobs can be created that send [Altova MapForce](#) mappings to the [Altova MapForce Server](#) for execution.)



Additionally to being invoked by a FlowForce job, StyleVision Server can also be invoked via the command line. Usage is described in the section [StyleVision Server Command Line](#)³⁷.

2.2 As a Standalone Server

StyleVision Server can be installed as a standalone product on Windows, Linux, and macOS systems. In this version its functionality is invoked only via the command line. Usage is described in the section [StyleVision Server Command Line](#)³⁷.

3 Installation and Licensing

This section describes installation, licensing and other setup procedures. It is organized into the following sections:

- [Setup on Windows](#) ¹²
- [Setup on Linux](#) ²⁰
- [Setup on macOS](#) ²⁷
- [Upgrade StyleVision Server](#) ³³
- [Migrate StyleVision Server to a New Machine](#) ³⁴

3.1 Setup on Windows

This section describes the [installation](#)¹² and licensing of StyleVision Server on Windows systems. The setup comprises the following steps:

1. [Install StyleVision Server](#)¹²
2. [Install LicenseServer](#)¹⁵
3. [Start LicenseServer and StyleVision Server](#)¹⁶
4. [Register StyleVision Server with LicenseServer](#)¹⁷
5. [Assign a license to StyleVision Server](#)¹⁷

The setup steps described above do not need to occur in exactly the same order in which they are listed. However, you do need to install before you start. And you do need to register StyleVision Server with LicenseServer before you can assign a license to StyleVision Server from LicenseServer.

System requirements (Windows)

Note the following system requirements:

- Windows 10, Windows 11
- Windows Server 2016 or newer

Prerequisites

Note the following prerequisites:

- Perform installation as a user with administrative privileges.
- From version 2021 onwards, a 32-bit version of StyleVision Server cannot be installed over a 64-bit version, or a 64-bit version over a 32-bit version. You must either (i) remove the older version before installing the newer version or (ii) upgrade to a newer version that is the same bit version as your older installation.

3.1.1 Install on Windows

Installing StyleVision Server

StyleVision Server can be installed on Windows systems as follows:

- As a separate standalone server product. To install StyleVision Server, download and run the StyleVision Server installer. Follow the on-screen instructions.
- To install StyleVision Server as part of the [FlowForce Server](#) package, download and run the FlowForce Server installer. Follow the on-screen instructions and make sure you check the option for installing StyleVision Server.

The installers of both StyleVision Server and [FlowForce Server](#) are available at the Altova Download Center (<https://www.altova.com/download.html>). You can select your installation language from the box in the lower left area of the wizard. Note that this selection also sets the default language of StyleVision Server. You can change the language later from the command line.

After installation, the StyleVision Server executable will be located by default at the following path:

```
<ProgramFilesFolder>\Altova\StyleVisionServer2025\bin\StyleVisionServer.exe
```

All the necessary registrations to use StyleVision Server via a COM interface, as a Java interface, and in the .NET environment will be done by the installer.

Uninstall StyleVision Server

Uninstall StyleVision Server as follows:

1. Right-click the Windows **Start** button and select **Settings**.
2. Open the Control Panel (start typing "Control Panel" and click the suggested entry).
3. Under *Programs*, click **Uninstall a program**.
4. In Control Panel, select StyleVision Server and click **Uninstall**.

Evaluation license

During the installation process, you will be given the option of requesting a 30-day evaluation license for StyleVision Server. After submitting the request, an evaluation license will be sent to the email address you registered.

3.1.2 Install on Windows Server Core

Windows Server Core is a minimal Windows installation that does not use a number of GUI features. You can install StyleVision Server on a Windows Server Core machine as follows:

1. Download the StyleVision Server installer executable from the Altova website. This file is named `styleVisionServer<version>.exe`. Make sure to choose the executable matching your server platform (32-bit or 64-bit).
2. On a standard Windows machine (not the Windows Server Core machine), run the command `styleVisionServer<version>.exe /u`. This unpacks the `.msi` file to the same folder as the installer executable.
3. Copy the unpacked `.msi` file to the Windows Server Core machine.
4. If you are updating an earlier version of StyleVision Server, shut down StyleVision Server before carrying out the next step.
5. Use the `.msi` file for the installation by running the command `msiexec /i styleVisionServer.msi`. This starts the installation on Windows Server Core.

Important: Keep the MSI file!

Note the following points:

- Keep the extracted `.msi` file in a safe place. You will need it later to uninstall, repair, or modify your installation.
- If you want to rename the MSI file, do this before you install StyleVision Server.
- The MSI filename is stored in the registry. You can update its name there if the filename has changed.

Register StyleVision Server with LicenseServer

If you are installing StyleVision Server for the first time or are upgrading to a **major version**, you will need to register StyleVision Server with an Altova LicenseServer on your network. If you are upgrading to a non-major version of StyleVision Server, then the previous LicenseServer registration will be known to the installation and there is no need to register StyleVision Server with LicenseServer. However, if you want to change the LicenseServer that is used by StyleVision Server at any time, then you will need to register StyleVision Server with the new LicenseServer.

To register StyleVision Server with an Altova LicenseServer during installation, run the installation command with the `REGISTER_WITH_LICENSE_SERVER` property, as listed below, providing the name or address of the LicenseServer machine as the value of the property, for example:

```
msiexec /i StyleVisionServer.msi REGISTER_WITH_LICENSE_SERVER="localhost"
```

To register StyleVision Server with an Altova LicenseServer after installation, run the following command:

```
msiexec /r StyleVisionServer.msi REGISTER_WITH_LICENSE_SERVER="<MyLS-IPAddress>"
```

Useful commands

Given below are a set of commands that are useful in the installation context.

To test the return value of the installation, run a script similar to that below. The return code will be in the `%errorlevel%` environment variable. A return code of 0 indicates success.

```
start /wait msiexec /i StyleVisionServer.msi /q
echo %errorlevel%
```

For a silent installation with a return code and a log of the installation process:

```
start /wait msiexec /i StyleVisionServer.msi /q /L*v! <pathToInstallLogFile>
```

To modify the installation:

```
msiexec /m StyleVisionServer.msi
```

To repair the installation:

```
msiexec /r StyleVisionServer.msi
```

To uninstall StyleVision Server:

```
msiexec /x StyleVisionServer.msi
```

To uninstall StyleVision Server silently and report the detailed outcome in a log file:

```
start /wait msiexec /x StyleVisionServer.msi /q /L*v! <pathToUninstallLogFile>
```

To install StyleVision Server using another language (available language codes are: German=`de`; Spanish=`es`; French=`fr`):

```
msiexec /i StyleVisionServer.msi INSTALLER_LANGUAGE=<languageCode>
```

Note: On Windows Server Core, the charts and barcode functionality of StyleVision Server will not be available.

Note: To install taxonomies, use the Taxonomy Package Manager via the command line. See the StyleVision Server manual for information about how to do this.

3.1.3 Install LicenseServer (Windows)

In order for StyleVision Server to work, it must be licensed via an [Altova LicenseServer](#) on your network. When you install StyleVision Server or FlowForce Server on Windows systems, you can install LicenseServer together with StyleVision Server or FlowForce Server. If a LicenseServer is already installed on your network, you do not need to install another one—unless a newer version of LicenseServer is required. (See *next point*, [LicenseServer versions](#).)

During the installation process of StyleVision Server or FlowForce Server, check or uncheck the option for installing LicenseServer as appropriate.

Note the following points:

- If you have not installed LicenseServer yet, leave the default settings as is. The wizard will install the latest version on the computer where you are running the wizard.
- If you have not installed LicenseServer yet and want to install Altova LicenseServer on another computer and use it from there, then clear the check box *Install Altova LicenseServer on this machine* and choose **Register Later**. In this case, you will need to install LicenseServer separately on the other machine and register StyleVision Server afterwards with the LicenseServer on that machine.
- If LicenseServer has already been installed on your computer but is a lower version than the one that would be installed by the installation wizard, then leave the wizard's default setting (for upgrading to the newer version) as is. In this case, the installation wizard will automatically upgrade your LicenseServer version. The existing registration and licensing information will be carried over to the new version of LicenseServer.
- If LicenseServer has already been installed on your computer or network and has the same version as the one indicated by the wizard, do the following:
 - Clear the check box *Install Altova LicenseServer on this machine*.
 - Under *Register this product with*, choose the LicenseServer with which you want to register StyleVision Server. Alternatively, choose **Register Later**. Note that you can always select **Register Later** if you want to ignore the LicenseServer associations and carry on with the installation of StyleVision Server.

For information, see how to [register](#)¹⁷ and [license](#)¹⁷ StyleVision Server with [Altova LicenseServer](#). Also see the [LicenseServer documentation](#) for more detailed information.

LicenseServer versions

- Altova products must be licensed either (i) with a version of LicenseServer that corresponds to the installed StyleVision Server version or (ii) with a later version of LicenseServer.
- The LicenseServer version that corresponds to the current version of StyleVision Server is [3.17](#).
- On Windows, you can install the corresponding version of LicenseServer as part of the StyleVision Server installation or install LicenseServer separately. On Linux and macOS, you must install LicenseServer separately.
- Before a newer version of LicenseServer is installed, any older one must be de-installed.
- At the time of LicenseServer de-installation, all registration and licensing information held in the older version of LicenseServer will be saved to a database on your server machine. This data will be imported automatically into the newer version when the newer version is installed.
- LicenseServer versions are backwards compatible. They will work with older versions of StyleVision Server.

- The latest version of LicenseServer available on the Altova website. This version will work with any current or older version of StyleVision Server.
- The version number of the currently installed LicenseServer is given at the bottom of the [LicenseServer configuration page](#) (all tabs).

3.1.4 Start LicenseServer, StyleVision Server (Windows)

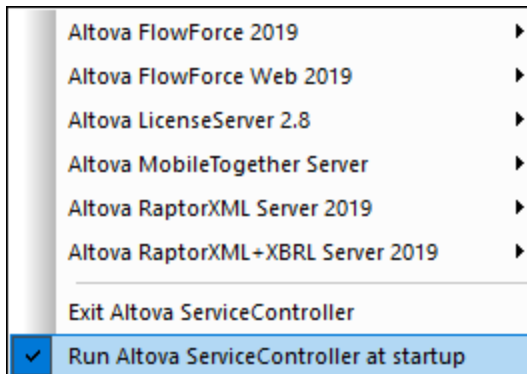
Altova LicenseServer (LicenseServer for short) and StyleVision Server are both started via Altova ServiceController.

Altova ServiceController

Altova ServiceController (ServiceController for short) is an application for conveniently starting, stopping and configuring Altova services **on Windows systems**. ServiceController is installed with Altova LicenseServer and with Altova server products that are installed as services (DiffDog Server, FlowForce Server, Mobile Together Server, and RaptorXML(+XBRL) Server). ServiceController can be accessed via the system tray (*screenshot below*).

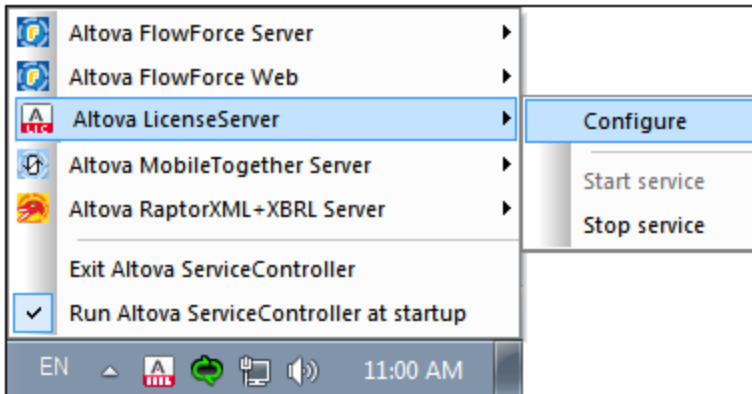


To specify that ServiceController starts automatically on logging in to the system, click the **ServiceController** icon in the system tray to display the **ServiceController** menu (*screenshot below*), and then toggle on the command **Run Altova ServiceController at Startup**. (This command is toggled on by default.) To exit ServiceController, click the **ServiceController** icon in the system tray and, in the menu that appears (see *screenshot below*), click **Exit Altova ServiceController**.



Start LicenseServer

To start LicenseServer, click the **ServiceController** icon in the system tray, hover over **Altova LicenseServer** in the menu that pops up (see *screenshot below*), and then select **Start Service** from the LicenseServer submenu. If LicenseServer is already running, then the *Start Service* option will be disabled. You can also stop the service via ServiceController.



3.1.5 Register StyleVision Server (Windows)

To be able to license StyleVision Server from Altova LicenseServer, StyleVision Server must be registered with LicenseServer. To register StyleVision Server from the command line interface, use the `licenseserver` command and supply the address of the LicenseServer machine (see *below*).

```
StyleVisionServer licenseserver [options] ServerName-Or-IP-Address
```

For example, if `localhost` is the name of the server on which LicenseServer is installed, use the following command:

```
StyleVisionServer licenseserver localhost
```

If StyleVision Server was installed as part of a [FlowForce Server](#) installation, registering FlowForce Server with LicenseServer will automatically also register StyleVision Server. Essentially: (i) Start Altova FlowForce Web as a service via ServiceController (see *previous point*); (ii) Enter your password to access the Setup page; (iii) Select the LicenseServer name or address and click **Register with LicenseServer**. For more information, see [Register FlowForce Server](#).

After successful registration, go to the [Client Management tab of LicenseServer's configuration page](#) to assign a license to StyleVision Server.

For more information about registering Altova products with LicenseServer, see the [LicenseServer user manual](#).

3.1.6 Assign License (Windows)

After successfully registering StyleVision Server, it will be listed in the Client Management tab of the configuration page of LicenseServer. Go there and [assign a license](#) to StyleVision Server.

The licensing of Altova server products is based on the number of processor cores available on the product machine. For example, a dual-core processor has two cores, a quad-core processor four cores, a hexa-core processor six cores, and so on. The number of cores licensed for a product must be greater than or equal to

the number of cores available on that server machine, whether the server is a physical or virtual machine. For example, if a server has eight cores (an octa-core processor), you must purchase at least one 8-core license. You can also combine licenses to achieve the core count. So, two 4-core licenses can also be used for an octa-core server instead of one 8-core license.

If you are using a computer server with a large number of CPU cores but only have a low volume to process, you may also create a virtual machine that is allocated a smaller number of cores and purchase a license for that number. Such a deployment, of course, would have less processing speed than if all available cores on the server were utilized.

Note: Each Altova server product license can be used for only one client machine at a time, even if the license has unused licensing capacity. (A client machine is the machine on which the Altova server product is installed.) For example, if a 10-core license is used for a client machine that has 6 CPU cores, then the remaining 4 cores of licensing capacity cannot be used simultaneously for another client machine.

Single-thread execution

If an Altova server product allows single-thread execution, an option for *Single-thread execution* will be available. In these cases, if an Altova server-product license for only one core is available in the license pool, a machine with multiple cores can be assigned this one-core license. In such a case, the machine will run that product on a single core. Processing will therefore be slower, because multi-threading (which is possible on multiple cores) will not be available. The product will be executed in single thread mode on that machine.

To assign a single-core license to a multiple-core machine in LicenseServer, select the *Limit to single thread execution* check box for that product.

Estimate of core requirements

There are various external factors that influence the data volumes and processing times your server can handle (for example: the hardware, the current load on the CPU, and memory allocation of other applications running on the server). In order to measure performance as accurately as possible, test the applications in your environment with data volumes and in conditions that approximate as closely as possible to real business situations.

3.1.7 Additional Setup Notes (Windows)

In order to run the Windows examples that are packaged with StyleVision Server in the `etc\examples` sub-folder of the application folder, the StyleVision Server DLL must be correctly registered with the system. A registration error typically occurs if you have, over time, installed different bit-versions of StyleVision Server (32-bit and 64-bit) on a single machine.

To correctly register the StyleVision Server DLL (either the 32-bit or 64-bit version) on Windows machines, do the following:

1. Open a command prompt in administrator mode
2. Switch to the folder in which the DLL is located. The command to do this would be: `cd C:\Program Files\Altova\StyleVisionServer2025\bin`
3. Run the following command to register the DLL (either 32-bit or 64-bit): `regsvr32 styleVisionServer.dll`
4. Ensure that you get a popup saying that the registration succeeded
5. Open Visual Studio
6. Load the project using `styleVisionServerAPI_Sample.sln`

7. Confirm that your `Program.cs` file contains valid pathways
8. Run the project by using **Ctrl+F5**

Note: The path to the application folder on Windows systems is typically: `C:\Program Files\Altova\StyleVisionServer2025`.

3.2 Setup on Linux

This section describes the [installation](#)²⁰ and licensing of StyleVision Server on Linux systems (Debian, Ubuntu, CentOS, RedHat). The setup comprises the following steps:

1. [Install StyleVision Server](#)²⁰
2. [Install LicenseServer](#)²²
3. [Start LicenseServer](#)²³
4. [Register StyleVision Server with LicenseServer](#)²³
5. [Assign a license to StyleVision Server](#)²⁴

The setup steps described above do not need to occur in exactly the same order in which they are listed. However, you do need to install before you start. And you do need to register StyleVision Server with LicenseServer before you can assign a license to StyleVision Server from LicenseServer.

System requirements (Linux)

- Red Hat Enterprise Linux 7 or newer
- CentOS 7, CentOS Stream 8
- Debian 10 or newer
- Ubuntu 20.04, 22.04, 24.04
- AlmaLinux 9.0
- Rocky Linux 9.0

Prerequisites

- Perform installation either as **root** user or as a user with **sudo** privileges.
- The previous version of StyleVision Server must be uninstalled before a new one is installed.
- If you plan to use Altova's Charts functionality, then at least one font must be installed on your system to ensure that charts will be rendered correctly. To list installed fonts, use, for example, the `fc-list` command of the [Fontconfig library](#).
- The following libraries are required as a prerequisite to install and run the application. If the packages below are not already available on your Linux machine, run the `yum` command (or `apt-get` if applicable) to install them.

CentOS, RedHat	Debian	Ubuntu
krb5-libs	libgssapi-krb5-2	libgssapi-krb5-2

3.2.1 Install on Linux

StyleVision Server is available for installation on Linux systems. Do the installation either as `root` user or a user with `sudo` privileges.

Integration of FlowForce Server and other Altova server products

If you are installing StyleVision Server together with FlowForce Server, it is recommended that you install FlowForce Server first. If you install StyleVision Server before FlowForce Server, then, after having installed both StyleVision Server and FlowForce Server, run the following command:

```
cp /opt/Altova/StyleVisionServer2025/etc/*.tool /opt/Altova/FlowForceServer2025/tools
```

This command copies the `.tool` file from `/etc` directory of StyleVision Server to the FlowForce Server `/tools` directory. The `.tool` file is required by FlowForce Server. It contains the path to the StyleVision Server executable. You do not need to run this command if you install FlowForce Server before installing StyleVision Server.

Uninstall StyleVision Server

Before you install StyleVision Server, you should uninstall any older version.

To check which Altova server products are installed:

```
[Debian, Ubuntu]:  dpkg --list | grep Altova
[CentOS, RedHat]:  rpm -qa | grep server
```

To uninstall an old version of StyleVision Server:

```
[Debian, Ubuntu]:  sudo dpkg --remove stylevisionserver
[CentOS, RedHat]:  sudo rpm -e stylevisionserver
```

On Debian and Ubuntu systems, it might happen that StyleVision Server still appears in the list of installed products after it has been uninstalled. In this case, run the `purge` command to clear StyleVision Server from the list. You can also use the `purge` command *instead* of the `remove` command listed above.

```
[Debian, Ubuntu]:  sudo dpkg --purge stylevisionserver
```

Download the StyleVision Server Linux package

StyleVision Server installation packages for the following Linux systems are available at the [Altova website](#).

Distribution	Package extension
Debian	.deb
Ubuntu	.deb
CentOS	.rpm
RedHat	.rpm

After downloading the Linux package, copy it to any directory on the Linux system. Since you will need to license StyleVision Server with an [Altova LicenseServer](#), you may want to download LicenseServer from the [Altova website](#) at the same time as you download StyleVision Server.

Install StyleVision Server

In a terminal window, switch to the directory where you copied the Linux package. For example, if you copied it to a user directory called `MyAltova` that is located in the `/home/User` directory, switch to this directory as follows:

```
cd /home/User/MyAltova
```

Install StyleVision Server using the relevant command:

```
[Debian]:  sudo dpkg --install stylevisionserver-2025-debian.deb
[Ubuntu]:  sudo dpkg --install stylevisionserver-2025-ubuntu.deb
[CentOS]:  sudo rpm -ivh stylevisionserver-2025-1.x86_64.rpm
[RedHat]:  sudo rpm -ivh stylevisionserver-2025-1.x86_64.rpm
```

You may need to adjust the name of the package above to match the current release or service pack version.

The StyleVision Server package will be installed in the following folder:

```
/opt/Altova/StyleVisionServer2025
```

3.2.2 Install LicenseServer (Linux)

In order for StyleVision Server to work, it must be licensed via an [Altova LicenseServer](#) on your network. Download LicenseServer from the [Altova website](#) and copy the package to any directory. Install it just like you installed StyleVision Server (see [previous topic](#) ²⁰).

```
[Debian]:  sudo dpkg --install licenseserver-3.17-debian.deb
[Ubuntu]:  sudo dpkg --install licenseserver-3.17-ubuntu.deb
[CentOS]:  sudo rpm -ivh licenseserver-3.17-1.x86_64.rpm
[RedHat]:  sudo rpm -ivh licenseserver-3.17-1.x86_64.rpm
```

The LicenseServer package will be installed at the following path:

```
/opt/Altova/LicenseServer
```

For information, see how to [register](#) ²³ and [license](#) ²⁴ StyleVision Server with [Altova LicenseServer](#). Also see the [LicenseServer documentation](#) for more detailed information.

LicenseServer versions

- Altova products must be licensed either (i) with a version of LicenseServer that corresponds to the installed StyleVision Server version or (ii) with a later version of LicenseServer.
- The LicenseServer version that corresponds to the current version of StyleVision Server is **3.17**.
- On Windows, you can install the corresponding version of LicenseServer as part of the StyleVision Server installation or install LicenseServer separately. On Linux and macOS, you must install LicenseServer separately.
- Before a newer version of LicenseServer is installed, any older one must be de-installed.
- At the time of LicenseServer de-installation, all registration and licensing information held in the older

version of LicenseServer will be saved to a database on your server machine. This data will be imported automatically into the newer version when the newer version is installed.

- LicenseServer versions are backwards compatible. They will work with older versions of StyleVision Server.
- The latest version of LicenseServer available on the Altova website. This version will work with any current or older version of StyleVision Server.
- The version number of the currently installed LicenseServer is given at the bottom of the [LicenseServer configuration page](#) (all tabs).

3.2.3 Start LicenseServer, StyleVision Server (Linux)

Start Altova LicenseServer and StyleVision Server either as `root` user or a user with `sudo` privileges.

Start LicenseServer

To correctly register and license StyleVision Server with LicenseServer, LicenseServer must be running as a daemon on the network. Start LicenseServer as a daemon with the following command:

```
sudo systemctl start licenseserver
```

If at any time you need to stop LicenseServer, replace `start` with `stop` in the command above. For example:

```
sudo systemctl stop licenseserver
```

3.2.4 Register StyleVision Server (Linux)

To be able to license StyleVision Server from Altova LicenseServer, StyleVision Server must be registered with LicenseServer.

To register StyleVision Server, go to its CLI and use the `licenseserver` command:

```
sudo /opt/Altova/StyleVisionServer2025/bin/stylevisionserver licenseserver [options]
ServerName-Or-IP-Address
```

For example, if `localhost` is the name of the server on which LicenseServer is installed:

```
sudo /opt/Altova/StyleVisionServer2025/bin/stylevisionserver licenseserver localhost
```

In the command above, `localhost` is the name of the server on which LicenseServer is installed. Notice also that the location of the StyleVision Server executable is:

```
/opt/Altova/StyleVisionServer2025/bin/
```

After successful registration, go to the [Client Management tab of LicenseServer's configuration page](#) to assign a license to StyleVision Server.

For more information about registering Altova products with LicenseServer, see the [LicenseServer user manual](#).

3.2.5 Assign License (Linux)

After successfully registering StyleVision Server, it will be listed in the Client Management tab of the configuration page of LicenseServer. Go there and [assign a license](#) to StyleVision Server.

The licensing of Altova server products is based on the number of processor cores available on the product machine. For example, a dual-core processor has two cores, a quad-core processor four cores, a hexa-core processor six cores, and so on. The number of cores licensed for a product must be greater than or equal to the number of cores available on that server machine, whether the server is a physical or virtual machine. For example, if a server has eight cores (an octa-core processor), you must purchase at least one 8-core license. You can also combine licenses to achieve the core count. So, two 4-core licenses can also be used for an octa-core server instead of one 8-core license.

If you are using a computer server with a large number of CPU cores but only have a low volume to process, you may also create a virtual machine that is allocated a smaller number of cores and purchase a license for that number. Such a deployment, of course, would have less processing speed than if all available cores on the server were utilized.

Note: Each Altova server product license can be used for only one client machine at a time, even if the license has unused licensing capacity. (A client machine is the machine on which the Altova server product is installed.) For example, if a 10-core license is used for a client machine that has 6 CPU cores, then the remaining 4 cores of licensing capacity cannot be used simultaneously for another client machine.

Single-thread execution

If an Altova server product allows single-thread execution, an option for *Single-thread execution* will be available. In these cases, if an Altova server-product license for only one core is available in the license pool, a machine with multiple cores can be assigned this one-core license. In such a case, the machine will run that product on a single core. Processing will therefore be slower, because multi-threading (which is possible on multiple cores) will not be available. The product will be executed in single thread mode on that machine.

To assign a single-core license to a multiple-core machine in LicenseServer, select the *Limit to single thread execution* check box for that product.

Estimate of core requirements

There are various external factors that influence the data volumes and processing times your server can handle (for example: the hardware, the current load on the CPU, and memory allocation of other applications running on the server). In order to measure performance as accurately as possible, test the applications in your environment with data volumes and in conditions that approximate as closely as possible to real business situations.

3.2.6 Notes about Environment (Linux)

Folders

Given below is a list of important folders in your StyleVision Server setup.

Installation root

`/opt/Altova/StyleVisionServer2025/`

☐ License Files

```
/var/opt/Altova/StyleVisionServer
```

☐ Environment settings

```
/etc/profile.d/jdbc.sh
```

The environment settings file (typically named `jdbc.sh`) is executed at system start. The definitions in it must be specific to your particular environment. The example path above serves only as a general guide.

Note: The environment settings file sets the variables for **all users** on the system, so you must be careful when modifying settings. For example, if you modify a class path in this file, then the modifications will be applied across the system. If you wish to make changes for StyleVision Server only, you might want to consider using a unit file (explained in the section *JDBC Connections* below).

Database connections

On Linux, the following database connections are supported:

- JDBC — You can use JDBC for all supported databases except Microsoft Access
- Native connections — Currently available for SQLite and PostgreSQL databases

If you are using JDBC, note the following points:

- The Java Runtime Environment or SDK must be installed.
- The JDBC drivers for the target database must be installed.
- The following environment variables must be set correctly for your environment:
 - `CLASSPATH`: to find the jar-files that connect to the JDBC database; the jar-files can be entered either in (i) an executable script (like `jdbc.sh`) that is executed on system start or (ii) a unit file that is executed when StyleVision Server is started as a service. Using a unit file to specify the jar-files has the advantage that the files required for StyleVision Server's JDBC connections will be located without you having to modify the existing system configuration. A unit file is listed below.
 - `PATH`: to find the JRE, but might not be necessary depending on the installation
 - `JAVA_HOME`: if necessary, depending on the installation.

Listing of important files

The following shell script (or unit file) is copied to the folder `/opt/Altova/StyleVisionServer/etc` so as not to overwrite already existing configuration files. Make the necessary changes as required. Also see the section *JDBC Connections* above. The parts highlighted in blue are environment-specific and need to be adjusted to match your environment:

☐ Shell script (unit file)

```

#- jdbc - environment -
export PATH=/usr/local/jdk1.7.0_17/bin:/usr/lib64/qt-
3.3/bin:/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:/home/qa/bin
export JAVA_HOME=/usr/local/jdk1.7.0_17
export
CLASSPATH=/usr/local/jdbc/oracle/ojdbc6.jar:/usr/local/jdbc/oracle/xdm.jar:/usr/local/j
dbc/oracle/xmlparserv2.jar:/usr/local/jdbc/postgre/postgresql-9.0-
801.jdbc4.jar:/usr/local/jdbc/mssql/sqljdbc4.jar:/usr/local/jdbc/series/lib/jt400.jar:
/usr/local/jdbc/mysql/mysql-connector-java-5.1.16-
bin.jar:/usr/local/jdbc/sqlite/sqlitejdbc-

```

```
v056.jar:/usr/local/jdbc/Informix_JDBC_Driver/lib/ixjjdbc.jar:/usr/local/jdbc/sybase/jc  
onn7/jconn4.jar:/usr/local/jdbc/db2/db2jcc.jar:/usr/local/jdbc/db2/db2jcc_license_cu.ja  
r:./:
```

3.3 Setup on macOS

This section describes the [installation](#)²⁷ and licensing of StyleVision Server on macOS systems. The setup comprises the following steps:

1. [Install StyleVision Server](#)²⁷
2. [Install LicenseServer](#)²⁸
3. [Start LicenseServer](#)²⁹
4. [Register StyleVision Server with LicenseServer](#)²⁹
5. [Assign a license to StyleVision Server](#)³⁰

The setup steps described above do not need to occur in exactly the same order in which they are listed. However, you do need to install before you start. And you do need to register StyleVision Server with LicenseServer before you can assign a license to StyleVision Server from LicenseServer.

System Requirements (macOS)

Note the following system requirement:

- macOS 12 or newer

Prerequisites

Note the following prerequisites:

- Ensure that Altova LicenseServer has been installed and is running.
- Perform installation either as the `root` user or as a user with `sudo` privileges.
- The previous version of StyleVision Server must be uninstalled before a new one is installed.
- If you plan to use Altova's Charts functionality, then at least one font must be installed on your system to ensure that charts will be rendered correctly. To list installed fonts, use, for example, the `fc-list` command of the [Fontconfig library](#).
- The macOS machine must be configured so that its name resolves to an IP address. This means that you must be able to successfully ping the host name from the Terminal using the command `ping <hostname>`.

3.3.1 Install on macOS

This topic describes the installation and setup of StyleVision Server on macOS systems.

Integration with FlowForce

If you are installing StyleVision Server together with FlowForce Server, it is recommended that you install FlowForce Server first. If you install StyleVision Server before FlowForce Server, then, after having installed both, run the following command:

```
cp /usr/local/Altova/StyleVisionServer2025/etc/*.tool /usr/local/Altova/FlowForceServer2025/tools
```

This command copies the `.tool` file from `/etc` directory of StyleVision Server to the FlowForce Server `/tools` directory. The `.tool` file is required by FlowForce Server. It contains the path to the StyleVision Server executable. You do not need to run this command if you install FlowForce Server before installing StyleVision Server.

Uninstall StyleVision Server

In the Applications folder in Finder, right-click the StyleVision Server icon and select **Move to Trash**. The application will be moved to Trash. You will, however, still need to remove the application from the `usr` folder. Do this with the following command:

```
sudo rm -rf /usr/local/Altova/StyleVisionServer2025/
```

If you need to uninstall an old version of Altova LicenseServer, you must first stop it running as a service. Do this with the following command:

```
sudo launchctl unload /Library/LaunchDaemons/com.altova.LicenseServer.plist
```

To check whether the service has been stopped, open the Activity Monitor in Finder and make sure that LicenseServer is not in the list. Then proceed to uninstall in the same way as described above for StyleVision Server.

Install StyleVision Server

To install StyleVision Server, do the following:

1. Download the disk image (`.dmg`) file of StyleVision Server from the Altova website (<https://www.altova.com/download.html>).
2. Click to open the downloaded disk image (`.dmg`). This causes the StyleVision Server installer to appear as a new virtual drive on your computer.
3. On the new virtual drive, double-click the installer package (`.pkg`).
4. Go through the successive steps of the installer wizard. These are self-explanatory and include one step in which you have to agree to the license agreement before being able to proceed.
5. To eject the drive after installation, right-click it and select **Eject**.

The StyleVision Server package will be installed in the folder:

```
/usr/local/Altova/StyleVisionServer2025 (application binaries)  
/var/Altova/StyleVisionServer (data files: database and logs)
```

The StyleVision Server server daemon starts automatically after installation and a re-boot of the machine. You can always start StyleVision Server as a daemon with the following command:

```
sudo launchctl load /Library/LaunchDaemons/com.altova.StyleVisionServer2025.plist
```

3.3.2 Install LicenseServer (macOS)

Altova LicenseServer can be downloaded from the Altova website (<https://www.altova.com/download.html>). Carry out the installation as described [here](#) ²⁷.

The LicenseServer package will be installed in the following folder:

```
/usr/local/Altova/LicenseServer
```

For information, see how to [register](#)²⁹ and [license](#)³⁰ StyleVision Server with [Altova LicenseServer](#). Also see the [LicenseServer documentation](#) for more detailed information.

LicenseServer versions

- Altova products must be licensed either (i) with a version of LicenseServer that corresponds to the installed StyleVision Server version or (ii) with a later version of LicenseServer.
- The LicenseServer version that corresponds to the current version of StyleVision Server is [3.17](#).
- On Windows, you can install the corresponding version of LicenseServer as part of the StyleVision Server installation or install LicenseServer separately. On Linux and macOS, you must install LicenseServer separately.
- Before a newer version of LicenseServer is installed, any older one must be de-installed.
- At the time of LicenseServer de-installation, all registration and licensing information held in the older version of LicenseServer will be saved to a database on your server machine. This data will be imported automatically into the newer version when the newer version is installed.
- LicenseServer versions are backwards compatible. They will work with older versions of StyleVision Server.
- The latest version of LicenseServer available on the Altova website. This version will work with any current or older version of StyleVision Server.
- The version number of the currently installed LicenseServer is given at the bottom of the [LicenseServer configuration page](#) (all tabs).

3.3.3 Start LicenseServer, StyleVision Server (macOS)

Start Altova LicenseServer and StyleVision Server either as `root` user or a user with `sudo` privileges.

Start LicenseServer

To correctly register and license StyleVision Server with LicenseServer, LicenseServer must be running as a daemon. Start LicenseServer as a daemon with the following command:

```
sudo launchctl load /Library/LaunchDaemons/com.altova.LicenseServer.plist
```

If at any time you need to stop LicenseServer, replace `load` with `unload` in the command above.

3.3.4 Register StyleVision Server (macOS)

To be able to license StyleVision Server from Altova LicenseServer, StyleVision Server must be registered with LicenseServer.

To register StyleVision Server from the command line interface, use the `licenseserver` command:

```
sudo /usr/local/Altova/StyleVisionServer2025/bin/StyleVisionServer licenseserver  
[options] ServerName-Or-IP-Address
```

For example, if `localhost` is the name of the server on which LicenseServer is installed:

```
sudo /usr/local/Altova/StyleVisionServer2025/bin/StyleVisionServer licenseserver
localhost
```

In the command above, `localhost` is the name of the server on which LicenseServer is installed. Notice also that the location of the StyleVision Server executable is:

```
/usr/local/Altova/StyleVisionServer2025/bin/
```

After successful registration, go to the [Client Management tab of LicenseServer's configuration page](#) to assign a license to StyleVision Server.

For more information about registering Altova products with LicenseServer, see the [LicenseServer user manual](#).

3.3.5 Assign License (macOS)

After successfully registering StyleVision Server, it will be listed in the Client Management tab of the configuration page of LicenseServer. Go there and [assign a license](#) to StyleVision Server.

The licensing of Altova server products is based on the number of processor cores available on the product machine. For example, a dual-core processor has two cores, a quad-core processor four cores, a hexa-core processor six cores, and so on. The number of cores licensed for a product must be greater than or equal to the number of cores available on that server machine, whether the server is a physical or virtual machine. For example, if a server has eight cores (an octa-core processor), you must purchase at least one 8-core license. You can also combine licenses to achieve the core count. So, two 4-core licenses can also be used for an octa-core server instead of one 8-core license.

If you are using a computer server with a large number of CPU cores but only have a low volume to process, you may also create a virtual machine that is allocated a smaller number of cores and purchase a license for that number. Such a deployment, of course, would have less processing speed than if all available cores on the server were utilized.

Note: Each Altova server product license can be used for only one client machine at a time, even if the license has unused licensing capacity. (A client machine is the machine on which the Altova server product is installed.) For example, if a 10-core license is used for a client machine that has 6 CPU cores, then the remaining 4 cores of licensing capacity cannot be used simultaneously for another client machine.

Single-thread execution

If an Altova server product allows single-thread execution, an option for *Single-thread execution* will be available. In these cases, if an Altova server-product license for only one core is available in the license pool, a machine with multiple cores can be assigned this one-core license. In such a case, the machine will run that product on a single core. Processing will therefore be slower, because multi-threading (which is possible on multiple cores) will not be available. The product will be executed in single thread mode on that machine.

To assign a single-core license to a multiple-core machine in LicenseServer, select the *Limit to single thread execution* check box for that product.

Estimate of core requirements

There are various external factors that influence the data volumes and processing times your server can handle (for example: the hardware, the current load on the CPU, and memory allocation of other applications running on the server). In order to measure performance as accurately as possible, test the applications in your environment with data volumes and in conditions that approximate as closely as possible to real business situations.

3.3.6 Notes about Environment (macOS)

Folders

Given below is a list of important folders in your StyleVision Server setup.

▣ Installation root

`/usr/local/Altova/StyleVisionServer2025/`

▣ License Files

`/var/Altova/StyleVisionServer`

▣ Environment settings

`/Library/LaunchDaemons/com.altova.StyleVisionServer.plist`

The environment settings file must be defined according to your specific environment. The example path above serves only as a general guide.

Note: These environment variables are only set for the StyleVision Server process and do not have an impact on other users.

Database connections

On MacOS, the following database connections are supported:

- JDBC — You can use JDBC for all supported databases except Microsoft Access
- Native connections — Currently available for SQLite and PostgreSQL databases

If you are using JDBC, note the following points:

- The Java Runtime Environment or SDK must be installed.
- The JDBC-Connects for the target database must be installed.
- The following environment variables must be set correctly for your environment:
 - **CLASSPATH:** to find the jar-files; the class path is set in the `plist` file.
 - **PATH:** to find the JRE, but might not be necessary depending on the installation
 - **JAVA_HOME:** if necessary, depending on the installation

Java 6 for fillable PDF forms

If the fillable parts of a fillable PDF are missing when the PDF is opened on a Mac OS system, one likely cause is that Java 6 is not installed on the machine. If this is the case, you can install Java 6 from https://support.apple.com/kb/dl1572?locale=en_US. If a version newer than Java 6 has already been installed, then the installation of the older Java 6 version will not affect the working of the newer version, which will be the default version of the system.

3.4 Upgrade StyleVision Server

The simplest way to carry over a license from the previous version of StyleVision Server to a newer version is via the installation process. The key steps during installation are:

1. Register the new version of StyleVision Server with the LicenseServer that holds the license of the older version of StyleVision Server.
2. Accept the license agreement of StyleVision Server. (If you do not accept the agreement, the new version will not be installed.)

Note: If you do not register StyleVision Server with LicenseServer during the installation process, you can do this later and then complete the licensing process.

3.5 Migrate StyleVision Server to a New Machine

If you want to migrate StyleVision Server from one machine to another (including across supported platforms), follow the guidelines below.

Migrating StyleVision Server to a new machine consists of re-assigning the license from the old machine to the new machine. Do this as follows:

1. Install StyleVision Server on the new machine. If it has already been installed as part of FlowForce Server installation, ignore this step.
2. On the new machine, register StyleVision Server with Altova LicenseServer.
3. On the old machine, make sure no clients are using the server.
4. Open the Altova LicenseServer administration page. Deactivate the license from the old StyleVision Server machine and re-assign it to the new machine.

Note: If you were using XML catalogs on the old machine, migrate these to the new machine.

3.6 FOP Requirements

By default, StyleVision Server uses the FO processor of the [Apache FOP Project](#) when it generates PDF documents via its `generate` command. (You can select some other FO processor with the `setfopath` command.)

FOP will have been installed with StyleVision Server. On Windows systems FOP is installed in the folder `c:\ProgramData\Altova\SharedBetweenVersions`. On Linux and macOS systems, it is installed in a descendant folder of `styleVisionServer2025`.

For information about running FOP 2.9, see the [page on this topic](#) at the Apache.org website. (If you are using another version of FOP, go to the corresponding page of that FOP version.)

Java Runtime Environment

To create PDFs, FOP requires that Java Runtime Environment (JRE) be installed on your machine. Check the minimum JRE version required for your version of FOP at the [Apache.org website](#). For example, for version FOP 2.8, a JRE 1.8 or later is required. If JRE is not installed and you try to run FOP to generate a PDF, you will get a message saying there was an error on calling FOP.

Setting up OpenJDK

StyleVision Server has been tried regularly with the latest available [OpenJDK version](#). At the time of writing, the latest version is 19.0.2. After you have downloaded and unzipped your OpenJDK package, make sure (i) to add the `java\bin` path to the system paths and (ii) to set the `JAVA_HOME` environment variable.

On Windows, set the `JAVA_HOME` environment variable as follows:

1. On the Windows taskbar, right-click the Windows icon and select *System*.
2. In the Settings window, under *Related Settings*, click *Advanced System Settings*.
3. On the *Advanced* tab, click *Environment Variables*.
4. Click *New* to create a new environment variable and name it `JAVA_HOME`.
5. Set the value of the environment variable. For example, on a `c:\Programs\openjdk\bin\java.exe` set the value to `c:\Programs\openjdk`.

3.7 Security Considerations

XSLT, XPath, XQuery are Turing-complete functional programming languages with local and remote file access and dynamic execution possibility — therefore, it is recommended to only permit access to them for transformations and/or file processing in a safe and regulated environment, where one has control over the input files and can ensure to execute only previously audited scripts. Should there be a need to access them from an external/public network (or a non-secure sub-network), then it is recommended to limit access with a reverse proxy that implements user authentication and authorization. Furthermore, it is recommended to run the process with a separate user account with access control configured at OS-level to restrict access only to authorized parts of the file system.

4 StyleVision Server Command Line

Default location of StyleVision Server executable

Given below are the default locations of the StyleVision Server executable:

Linux /opt/Altova/StyleVisionServer2025/bin/**stylevisionserver**

Mac /usr/local/Altova/StyleVisionServer2025/bin/**stylevisionserver**

Windows <ProgramFilesFolder>\Altova\StyleVisionServer2025\bin\StyleVisionServer.exe

Usage and list of CLI commands

The command line syntax is:

```
stylevisionserver --h | --help | --version | <command> [options] [arguments]
```

- `--help` (short form `--h`) displays the help text of the given command. If no command is named, then all commands of the executable are listed, each with a brief description of the command.
- `--version` displays the version number of StyleVision Server.
- `<command>` is the command to execute. Commands are described in the sub-sections of this section (*see list below*).
- `[options]` are the options of a command; they are listed and described with their respective commands.
- `[arguments]` are the arguments of a command; they are listed and described with their respective commands.

▼ Casing and slashes on the command line

StyleVisionServer on Windows

stylevisionserver on Windows and Unix (Linux, Mac)

* Note that lowercase (`stylevisionserver`) works on all platforms (Windows, Linux, and Mac), while upper-lower (`StyleVisionServer`) works only on Windows and Mac.

* Use forward slashes on Linux and Mac, backslashes on Windows.

CLI commands

Available commands are listed below and are explained in the sub-sections of this section.

- [accepteula](#)³⁹: (Linux only) Accepts the end user license agreement for the current installation of StyleVision Server
- [assignlicense](#)⁴⁰: Uploads a license to LicenseServer and assigns this license to StyleVision Server.
- [exportresourcestrings](#)⁴²: Exports all application resource strings to an XML file.
- [generate](#)⁴⁴: Generates one or several documents from an input XML file and an XSLT stylesheet in the input PXF file.
- [help](#)⁴⁹: Displays information about the command that is submitted in the argument (or about all commands if no argument is submitted).

- [licenseserver](#)⁵⁰: Registers StyleVision Server with a LicenseServer on the local network.
- [pdfdata](#)⁵²: Generates form data from a PDF file to an FDF or XML file.
- [setdeflang](#)⁵⁴: Sets the default language of StyleVision Server.
- [setfopath](#)⁵⁵: Selects an alternative FO processor for subsequent PDF generation.
- [verifylicense](#)⁵⁷: Checks if current StyleVision Server is licensed and, optionally, whether it is licensed with the given license key.
- [version](#)⁵⁸: Displays the version number of StyleVision Server.

4.1 accepteula (Linux only)

Syntax and description

In order to be able to run StyleVision Server, the application's end user license agreement (EULA) must be accepted. You can accept the application's EULA by running the `accepteula` command.

This command is useful, for example, if you want to license and run StyleVision Server directly via automated processes that use scripts.

```
stylevisionserver accepteula [options]
```

- The command works only for Altova server products that have been installed on Linux machines.
- You must register StyleVision Server with LicenseServer before running the `accepteula` command.
- Use the `--h, --help` option to display information about the command.
- Use lowercase `stylevisionserver`.
- Use forward slashes on Linux.

Examples

Examples of the `accepteula` command:

```
stylevisionserver accepteula
```

Options

Use the `--h, --help` option to display information about the command.

4.2 assignlicense

Syntax and description

The `assignlicense` command uploads a license file to the Altova LicenseServer with which StyleVision Server is registered (see the `licenseserver` command), and assigns the license to StyleVision Server. It takes the path of a license file as its argument. The command also allows you to test the validity of a license.

```
stylevisionserver assignlicense [options] FILE
```

- The `FILE` argument takes the path of the license file.
- The `--test-only` option uploads the license file to LicenseServer and validates the license, but does not assign the license to StyleVision Server.

For details about licensing, see the LicenseServer documentation (<https://www.altova.com/manual/en/licenseserver/3.17/>).

▼ Casing and slashes on the command line

`stylevisionserver` on Windows

`stylevisionserver` on Windows and Unix (Linux, Mac)

* Note that lowercase (`stylevisionserver`) works on all platforms (Windows, Linux, and Mac), while upper-lower (`StyleVisionServer`) works only on Windows and Mac.

* Use forward slashes on Linux and Mac, backslashes on Windows.

▼ Backslashes, spaces, and special characters on Windows systems

On Windows systems: When spaces or special characters occur in strings (for example in file or folder names, or company, person or product names), use quotes: for example, "`My File`". Note, however, that a backslash followed by a double-quotation mark (for example, "`C:\My directory\`") might not be read correctly. This is because the backslash character is also used to indicate the start of an escape sequence, and the escape sequence `\"` stands for the double-quotation mark character. If you want to escape this sequence of characters, use a preceding backslash, like this: `\\`". To summarize: If you need to write a file path that contains spaces or an end backslash, write it like this: "`C:\My Directory\`".

Examples

Examples of the `assignlicense` command:

```
stylevisionserver assignlicense C:\licensepool\mylicensekey.altova_licenses
stylevisionserver assignlicense --test-only=true C:
\licensepool\mylicensekey.altova_licenses
```

- The first command above uploads the specified license to LicenseServer and assigns it to StyleVision Server.
- The last command uploads the specified license to LicenseServer and validates it, without assigning it to StyleVision Server.

Options

Options are listed in short form (if available) and long form. You can use one or two dashes for both short and long forms. An option may or may not take a value. If it takes a value, it is written like this: `--option=value`. Values can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required. If an option takes a Boolean value and no value is specified, then the option's default value is `TRUE`. Use the `--h, --help` option to display information about the command.

▼ test-only [t]

`--t, --test-only = true|false`

Values are `true|false`. If `true`, then the license file is uploaded to LicenseServer and validated, but not assigned.

4.3 exportresourcestrings

Syntax and description

The `exportresourcestrings` command outputs an XML file containing the resource strings of the StyleVision Server application in the specified language. Available export languages are English (`en`), German (`de`), Spanish (`es`), French (`fr`), and Japanese (`ja`).

```
stylevisionserver exportresourcestrings [options] LanguageCode XMLOutputFile
```

- The `LanguageCode` argument gives the language of the resource strings in the output XML file; this is the *export language*. Allowed export languages (with their language codes in parentheses) are: English (`en`), German, (`de`), Spanish (`es`), French (`fr`), and Japanese (`ja`).
- The `XMLOutputFile` argument specifies the path and name of the output XML file.

How to create localizations is described below.

▼ Casing and slashes on the command line

`styleVisionServer` on Windows

`stylevisionserver` on Windows and Unix (Linux, Mac)

* Note that lowercase (`stylevisionserver`) works on all platforms (Windows, Linux, and Mac), while upper-lower (`StyleVisionServer`) works only on Windows and Mac.

* Use forward slashes on Linux and Mac, backslashes on Windows.

▼ Backslashes, spaces, and special characters on Windows systems

On Windows systems: When spaces or special characters occur in strings (for example in file or folder names, or company, person or product names), use quotes: for example, "`My File`". Note, however, that a backslash followed by a double-quotation mark (for example, "`C:\My directory\"`) might not be read correctly. This is because the backslash character is also used to indicate the start of an escape sequence, and the escape sequence `\"` stands for the double-quotation mark character. If you want to escape this sequence of characters, use a preceding backslash, like this: `\\`". To summarize: If you need to write a file path that contains spaces or an end backslash, write it like this: "`C:\My Directory\`".

Examples

Examples of the `exportresourcestrings` command:

```
stylevisionserver exportresourcestrings de c:\Strings.xml
```

- The command above creates a file called `Strings.xml` at `c:\` that contains the resource strings of StyleVision Server in German.

Creating localized versions of StyleVision Server

You can create a localized version of StyleVision Server for any language of your choice. Five localized versions (English, German, Spanish, French, and Japanese) are already available in the `C:\Program Files`

(x86)\Altova\StyleVisionServer2025\bin folder, and therefore do not need to be created.

Create a localized version as follows:

1. Generate an XML file containing the resource strings by using the `exportresourcestrings` command (see *command syntax above*). The resource strings in this XML file will be one of the five supported languages: English (`en`), German (`de`), Spanish (`es`), French (`fr`), or Japanese (`ja`), according to the `LanguageCode` argument used with the command.
2. Translate the resource strings from one of the five supported languages into the target language. The resource strings are the contents of the `<string>` elements in the XML file. Do not translate variables in curly brackets, such as `{option}` or `{product}`.
3. Contact [Altova Support](#) to generate a localized StyleVision Server DLL file from your translated XML file.
4. After you receive your localized DLL file from [Altova Support](#), save the DLL in the `C:\Program Files (x86)\Altova\StyleVisionServer2025\bin` folder. Your DLL file will have a name of the form `StyleVisionServer2025_lc.dll`. The `_lc` part of the name contains the language code. For example, in `StyleVisionServer2025_de.dll`, the `de` part is the language code for German (Deutsch).
5. Run the `setdeflang` command to set your localized DLL file as the StyleVision Server application to use. For the argument of the `setdeflang` command, use the language code that is part of the DLL name.

Note: Altova StyleVision Server is delivered with support for five languages: English, German, Spanish, French, and Japanese. So you do not need to create a localized version of these languages. To set any of these languages as the default language, use StyleVision Server's `setdeflang` command.

4.4 generate

Syntax and description

The `generate` command (short form `gen`) generates one or more output files (HTML, PDF, RTF, and/or DOCX) by transforming the input XML file using the XSLT document/s contained in the input PXF file.

```
stylevisionserver generate | gen --inputxml=Filename [additional options] InputPXF
```

- The `--inputxml` option is mandatory; it gives the path to the XML file.
- The `InputPXF` argument specifies the path to the PXF file which contains the XSLT document/s that will be used to generate the output document/s. PXF files are created with [Altova's StyleVision application](#).
- Each output format is generated by specifying an option for that output (*see Options list below*). The value of each option is a path that specifies where the output is to be generated.

Note: StyleVision Server uses [Apache FOP](#), the FO processor of the Apache Project, to generate PDF files from FO. Apache FOP is installed with StyleVision Server at the following location: On Windows systems, `ProgramData\Altova\SharedBetweenVersions`; on Linux and macOS systems, in a descendant folder of the `StyleVisionServer2025` folder. Note that Apache FOP requires that **Java Runtime Environment 1.8 or later** be installed on the StyleVision Server machine. For 32-bit StyleVision Server, install the 32-bit Java; for 64-bit StyleVision Server, install the 64-bit Java. For more information about setting up FOP, see the topic [FOP Requirements](#) ³⁵.

▼ AltovaFOPServer for faster PDF generation

If you will be running multiple transformations to PDF, especially of large documents, you might want to use AltovaFOPServer to carry out the job faster. The increase in job speed occurs because AltovaFOPServer keeps the Java VM and FOP jar files loaded, thus saving the time required to load these files for each transformation. AltovaFOPServer will be installed with your StyleVision package. To use it for transformations, you must first start AltovaFOPServer via the command line.

Start AltovaFOPServer

The syntax of the command to start AltovaFOPServer is:

```
java -cp "{classPath}" {className} --pid "{pidFile}" --port {portNumbers}
```

An example CLI command to start AltovaFOPServer:

```
java -cp "C:\Program Files\Altova\Common2024\jar\gson\gson.jar;C:\Program Files\Altova\Common2024\jar\AltovaFOPServer.jar" com.altova.stylevision.fopserver.Main --pid "C:\Users\user\AppData\Local\Temp\AltovaFopServer.pid" --port 9090-9103,9999
```

The `classPath` argument must contain the paths to the `gson.jar` and `AltovaFOPServer.jar` files. Enter the correct locations of these files.

Call AltovaFOPServer in StyleVision Server

After the AltovaFOPServer has been started, you can call it from StyleVision Server by using the `generate` command with either the `altova-fopserver-pid-file` or `altova-fopserver-port` option (*see Options below*).

▼ Casing and slashes on the command line

`StyleVisionServer` on Windows

`stylevisionserver` on Windows and Unix (Linux, Mac)

* Note that lowercase (`stylevisionserver`) works on all platforms (Windows, Linux, and Mac), while upper-lower (`StyleVisionServer`) works only on Windows and Mac.

* Use forward slashes on Linux and Mac, backslashes on Windows.

▼ Backslashes, spaces, and special characters on Windows systems

On Windows systems: When spaces or special characters occur in strings (for example in file or folder names, or company, person or product names), use quotes: for example, "`My File`". Note, however, that a backslash followed by a double-quotation mark (for example, "`C:\My directory\"`) might not be read correctly. This is because the backslash character is also used to indicate the start of an escape sequence, and the escape sequence `\"` stands for the double-quotation mark character. If you want to escape this sequence of characters, use a preceding backslash, like this: `\\"`. To summarize: If you need to write a file path that contains spaces or an end backslash, write it like this: "`C:\My Directory\`".

Examples

Examples of the `generate` command:

```
stylevisionserver generate --inputxml=C:\MyFiles\ExpReport.xml --html=Test.html
ExpReport.pxf
stylevisionserver generate --inputxml=C:\ExpReport.pxf | zip\ExpReport.xml --
html=Test.html ExpReport.pxf
stylevisionserver generate --inputxml=altova://packagedfile/ExpReport.xml --
html=Test.html ExpReport.pxf
stylevisionserver generate --inputxml=ExternalXML.xml --html=Test.html Test.pxf
```

- The commands above contain the mandatory `--inputxml` option, the `InputPXF` argument (`Test.pxf`), and a minimum of one output-creation option (`--html` in all the examples above).
- The input XML file to use can be located inside the PXF file (see *second and third examples above*) or it can be an external XML file (located outside the PXF file; see *first and fourth examples above*).
- The `--inputxml` switch is ignored if the main schema source is DB or DB-XML, but it must be present for syntactical reasons, and you should use something like `--inputxml=database`.
- If the output-creation option `--html` takes a relative path, as in the examples above, then the output file's location will be relative to the folder in which the PXF file is.

Options

▼ inputxml [xml]

```
--xml, --inputxml = PathToXMLFile
```

This option is mandatory. It specifies the path to the XML file to process. The XML file can be located inside or outside the PXF file. To target XML files inside a PXF file, use the `|zip` locator (see the highlighted part in the examples above). The `--inputxml` option is ignored if the main schema source of the input PXF is a DB or DB-XML.

▼ `dbwhere [dbw]`

`--dbw, --dbwhere = WHEREClause`

An SQL `WHERE` clause that determines what rows of a DB-XML source to process.

▼ `param [p]`

`--p, --param = $ParamName:ParamValue`

Assigns a value to a parameter defined in the PXF file. The `--param` switch must be used before each parameter. Use quotes if `ParamName` or `ParamValue` contains a space. Example: `--p=$company:"Nanonull Inc"`

▼ `prohibit-output-outside-target-folder`

`--prohibit-output-outside-target-folder = true|false`

Values are `true|false`. If `true`, does not allow the creation of output in any folder other than that in which the main output file (HTML, PDF, RTF, DOCX, FO) is created. This provides protection for other folders if needed. Default is `false`.

▼ `outhtml [html]`

`--html, --outhtml = FilePath`

Path to the HTML file to generate.

▼ `outpdf [pdf]`

`--pdf, --outpdf = FilePath`

Path to the PDF file to generate.

▼ `outrtf [rtf]`

`--rtf, --outrtf = FilePath`

Path to the RTF file to generate.

▼ `outdocx [docx]`

`--docx, --outdocx = FilePath`

Path to the DOCX file to generate.

▼ `outtext [text]`

`--text, --outtext = FilePath`

Path to the Text file to generate.

▼ `outfo [fo]`

`--fo, --outfo = FilePath`

Path to the FO file to generate.

▼ `generate-html-output-as-mime`

`--generate-html-output-as-mime = true|false`

Values are `true|false`. If the option is not specified, default is `false`, if specified with no value, then `true`. If `true`, HTML output is generated as a mime stream.

▼ `altova-fopserver-pid-file`

`--altova-fopserver-pid-file = PathToFile`

The path to the PID file to use for connecting to a running AltovaFOPServer. You can connect to AltovaFOPServer via either the server's PID file or a port. This PID option is an alternative to the port option (see *next option*). For more information about using AltovaFOPServer, see the general description of the `generate` command above.

▼ `altova-fopserver-port`

`--altova-fopserver-port = Value`

A port number or a list of port numbers to use for connecting to a running AltovaFOPServer. The port numbers in a list are separated by commas. Instead of a single number in the list, you can use a range. So you can submit `9090-9100,9999` as the value of this option. Note that you can connect to AltovaFOPServer via either the AltovaFOPServer's PID file or a port. This port option is an alternative to the PID option (see *previous option*). For more information about using AltovaFOPServer, see the general description of the `generate` command above.

▼ `taxonomy-package`

`--taxonomy-package = FilePath`

Path to an additional taxonomy package. Add the option multiple times to specify more than one taxonomy package.

▼ `taxonomy-packages-config-file`

`--taxonomy-packages-config-file = FilePath`

Path to the `TaxonomyPackagesConfig.json` file, which is a common file used by the Altova products XMLSpy, MapForce, and StyleVision to configure XBRL taxonomies for use across these products.

▼ `verbose [v]`

`--v, --verbose = true|false`

Values are `true|false`. Turns the display of all messages, respectively, on or off. Default is `false` if the option is not provided, `true` if provided without a value.

▼ `lang [l]`

`--l, --lang = en|de|es|fr|ja`

The language used for displaying messages.

Use the `--h, --help` option to display information about the command.

Options are listed in short form (if available) and long form. You can use one or two dashes for both short and long forms. An option may or may not take a value. If it takes a value, it is written like this: `--option=value`. Values can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required. If an option takes a Boolean value and no value is specified, then the option's default value is `TRUE`. Use the `--h, --help` option to display

information about the command.

Catalogs

If you are using the Altova catalog mechanism, you can find the relevant catalog files in the `etc` folder of the StyleVision Server application folder. For detailed information, see the [Catalogs section of the Altova StyleVision manual](#).

You can create `CustomCatalog.xml` from the template file `CustomCatalog_template.xml`. Make sure that you rename the template file to `CustomCatalog.xml` since this latter file will be the file that is used in the catalog mechanism (not the template file).

Note the following:

- During a new installation of the same major version (same or different minor versions), the template file will be replaced by a new template file, but `CustomCatalog.xml` will be left untouched.
- However, if you are installing a new major version over a previous major version, then the previous major version folder will be deleted—together with its `CustomCatalog.xml`. So, if you want to continue using `CustomCatalog.xml`, make sure that you save `CustomCatalog.xml` from the previous major version folder to a safe place. After the new major version has been installed, you can copy the `CustomCatalog.xml` that you saved to the `etc` folder of the new major version and edit it there as required.

4.5 help

Syntax and description

The `help` command takes a single argument (`Command`), which is the name of the command for which help is required. It displays the command's syntax, its options, and other relevant information. If the `Command` argument is not specified, then all commands of the executable are listed, with each having a brief text description.

```
stylevisionserver help Command
```

▼ Casing and slashes on the command line

`stylevisionServer` on Windows

`stylevisionserver` on Windows and Unix (Linux, Mac)

* Note that lowercase (`stylevisionserver`) works on all platforms (Windows, Linux, and Mac), while upper-lower (`StyleVisionServer`) works only on Windows and Mac.

* Use forward slashes on Linux and Mac, backslashes on Windows.

Example

Example of the `help` command to display information about the `licenseserver` command:

```
stylevisionserver help licenseserver
```

The `--help` option

Help information about a command is also available by using the `--help` option of the command for which help information is required. The two commands below produce the same results:

```
stylevisionserver licenseserver --help
```

The command above uses the `--help` option of the `licenseserver` command.

```
stylevisionserver help licenseserver
```

The `help` command takes `licenseserver` as its argument.

Both commands display help information about the `licenseserver` command.

4.6 licenseserver

Syntax and description

The `licenseserver` command registers StyleVision Server with the Altova LicenseServer specified by the `Server-Or-IP-Address` argument. For the `licenseserver` command to be executed successfully, the two servers (StyleVision Server and LicenseServer) must be on the same network and LicenseServer must be running. You must also have administrator privileges in order to register StyleVision Server with LicenseServer.

```
stylevisionserver licenseserver [options] Server-Or-IP-Address
```

- The `Server-Or-IP-Address` argument takes the name or IP address of the LicenseServer machine.

Once StyleVision Server has been successfully registered with LicenseServer, you will receive a message to this effect. The message will also display the URL of the LicenseServer. You can now go to LicenseServer to assign StyleVision Server a license. For details about licensing, see the LicenseServer documentation (<https://www.altova.com/manual/en/licenseserver/3.17/>).

▼ Casing and slashes on the command line

`StyleVisionServer` on Windows

`stylevisionserver` on Windows and Unix (Linux, Mac)

* Note that lowercase (`stylevisionserver`) works on all platforms (Windows, Linux, and Mac), while upper-lower (`StyleVisionServer`) works only on Windows and Mac.

* Use forward slashes on Linux and Mac, backslashes on Windows.

▼ Backslashes, spaces, and special characters on Windows systems

On Windows systems: When spaces or special characters occur in strings (for example in file or folder names, or company, person or product names), use quotes: for example, "`My file`". Note, however, that a backslash followed by a double-quotation mark (for example, "`C:\My directory\"`) might not be read correctly. This is because the backslash character is also used to indicate the start of an escape sequence, and the escape sequence `\"` stands for the double-quotation mark character. If you want to escape this sequence of characters, use a preceding backslash, like this: `\\"`. To summarize: If you need to write a file path that contains spaces or an end backslash, write it like this: "`C:\My Directory\`".

Examples

Examples of the `licenseserver` command:

```
stylevisionserver licenseserver DOC.altova.com
stylevisionserver licenseserver localhost
stylevisionserver licenseserver 127.0.0.1
```

The commands above specify, respectively, the machine named `DOC.altova.com`, and the user's machine (`localhost` and `127.0.0.1`) as the machine running Altova LicenseServer. In each case, the command registers StyleVision Server with the LicenseServer on the machine specified. The last command calls the server-executable to execute the command.

Options

Options are listed in short form (if available) and long form. You can use one or two dashes for both short and long forms. An option may or may not take a value. If it takes a value, it is written like this: `--option=value`. Values can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required. If an option takes a Boolean value and no value is specified, then the option's default value is `TRUE`. Use the `--h, --help` option to display information about the command.

▼ json [j]

`--j, --json = true|false`

Values are `true|false`. If `true`, prints the result of the registration attempt as a machine-parsable JSON object.

4.7 pdfdata

Syntax and description

The `pdfdata` command generates an FDF file or XML file from the PDF file that is submitted as the *InputPDF* argument.

```
stylevisionserver pdfdata [options] InputPDF
```

- The *InputPDF* argument specifies the path to the PDF file, from which the output FDF or XML file will be generated. If the PDF file does not have any form data, the generated file will contain no form data.
- Use the `--outfdf` option to specify the location of the generated FDF file or the `--outxml` option to specify the location of the generated XML file.

For information about setting up FOP, which StyleVision Server uses by default to generate PDF, see the topic [FOP Requirements](#) ³⁵.

For more information about FDF files and designing fillable PDF forms, see the [Altova StyleVision \(Enterprise Edition\)](#) documentation.

▼ Casing and slashes on the command line

`stylevisionserver` on Windows

`stylevisionserver` on Windows and Unix (Linux, Mac)

* Note that lowercase (`stylevisionserver`) works on all platforms (Windows, Linux, and Mac), while upper-lower (`StyleVisionServer`) works only on Windows and Mac.

* Use forward slashes on Linux and Mac, backslashes on Windows.

▼ Backslashes, spaces, and special characters on Windows systems

On Windows systems: When spaces or special characters occur in strings (for example in file or folder names, or company, person or product names), use quotes: for example, "**My File**". Note, however, that a backslash followed by a double-quotation mark (for example, "**C:\My directory**") might not be read correctly. This is because the backslash character is also used to indicate the start of an escape sequence, and the escape sequence `\"` stands for the double-quotation mark character. If you want to escape this sequence of characters, use a preceding backslash, like this: `\\"`. To summarize: If you need to write a file path that contains spaces or an end backslash, write it like this: "**C:\My Directory**
****".

Examples

Examples of the `pdfdata` command:

```
stylevisionserver pdfdata --outfdf=C:\test\forms\FDFData.fdf C:\test\forms\TestForm.pdf
stylevisionserver pdfdata --outxml=C:\test\forms\XMLData.xml C:\test\forms\TestForm.pdf
```

The examples above create, respectively, an FDF file and an XML file from the same PDF input.

Options

▼ outfdf

`--outfdf = FilePath`

The path to the generated FDF file.

▼ outxml

`--outxml = FilePath`

The path to the generated XML file.

▼ verbose [v]

`--v, --verbose = true|false`

Values are `true|false`. Turns the display of all messages, respectively, on or off. Default is `false` if the option is not provided, `true` if provided without a value.

Use the `--h, --help` option to display information about the command.

Options are listed in short form (if available) and long form. You can use one or two dashes for both short and long forms. An option may or may not take a value. If it takes a value, it is written like this: `--option=value`. Values can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required. If an option takes a Boolean value and no value is specified, then the option's default value is `TRUE`. Use the `--h, --help` option to display information about the command.

4.8 setdeflang

Syntax and description

The `setdeflang` command (short form is `sd1`) sets the default language of StyleVision Server. Available languages are English (`en`), German (`de`), Spanish (`es`), French (`fr`), and Japanese (`ja`). The command takes a mandatory `LanguageCode` argument.

```
stylevisionserver setdeflang [options] LanguageCode
```

- The `LanguageCode` argument is required and sets the default language of StyleVision Server. The respective values to use are: `en`, `de`, `es`, `fr`, `ja`.
- Use the `--h, --help` option to display information about the command.

▼ Casing and slashes on the command line

`StyleVisionServer` on Windows

`stylevisionserver` on Windows and Unix (Linux, Mac)

* Note that lowercase (`stylevisionserver`) works on all platforms (Windows, Linux, and Mac), while upper-lower (`StyleVisionServer`) works only on Windows and Mac.

* Use forward slashes on Linux and Mac, backslashes on Windows.

Examples

Examples of the `setdeflang` (`sd1`) command:

```
stylevisionserver sd1 de
stylevisionserver setdeflang es
```

- The first command sets the default language of StyleVision Server to German.
- The second command sets the default language of StyleVision Server to Spanish.

Options

Use the `--h, --help` option to display information about the command.

4.9 setfopath

Syntax and description

The `setfopath` command (short form is `sfp`) specifies the path to an Apache FOP processor other than that included in the StyleVision Server package.

```
stylevisionserver setfopath | sfp [options] Path
```

- By default the Apache FOP processor that is included with StyleVision Server is used for processing FO documents and generating PDF output. If you wish to use some other Apache FOP processor instance than the processor supplied with StyleVision Server, use the `setfopath` command with the `Path` argument giving the path to the FO processor you want to use.
- After an alternative FO processor has been specified with the `setfopath` command, it is this processor that will be used when PDF is generated with subsequent [generate](#)⁴⁴ commands. To change processors again, use the `setfopath` command again. To switch back to StyleVision Server's FOP processor, locate the FOP folder on your system and use this path as the argument of `setfopath`.
- On Windows systems, the FOP folder that was installed with StyleVision Server will be located under `ProgramData\Altova\SharedBetweenVersions`; on Linux and macOS systems in a descendant folder of the `StyleVisionServer2025` folder.

For information about setting up FOP, which StyleVision Server uses by default to generate PDF, see the topic [FOP Requirements](#)³⁵.

For more information about FDF files and designing fillable PDF forms, see the [Altova StyleVision \(Enterprise Edition\)](#) documentation.

▼ Casing and slashes on the command line

`stylevisionserver` on Windows

`stylevisionserver` on Windows and Unix (Linux, Mac)

* Note that lowercase (`stylevisionserver`) works on all platforms (Windows, Linux, and Mac), while upper-lower (`StyleVisionServer`) works only on Windows and Mac.

* Use forward slashes on Linux and Mac, backslashes on Windows.

▼ Backslashes, spaces, and special characters on Windows systems

On Windows systems: When spaces or special characters occur in strings (for example in file or folder names, or company, person or product names), use quotes: for example, `"My File"`. Note, however, that a backslash followed by a double-quotation mark (for example, `"C:\My directory\"`) might not be read correctly. This is because the backslash character is also used to indicate the start of an escape sequence, and the escape sequence `\"` stands for the double-quotation mark character. If you want to escape this sequence of characters, use a preceding backslash, like this: `\\`". To summarize: If you need to write a file path that contains spaces or an end backslash, write it like this: `"C:\My Directory\ \"`.

Examples

After running the `setfopath` command, you can use the [generate](#) ⁴⁴ command to generate a PDF using the just-specified FO processor:

```
stylevisionserver setfopath C:\FOP\FOP.bat
stylevisionserver generate --inputxml=Test.xml --pdf=Test.pdf Test.pxf
```

The commands above do the following:

1. The `setfopath` command specifies that the FO processor at the location `C:\FOP\FOP.bat` is to be used to generate PDF in subsequent PDF-generation commands.
2. The `generate` command generates a PDF file from the specified input XML, using transformation files contained in the PXF file. The FO processor specified in the previous command is used for generating the PDF.

Options

Use the `--h, --help` option to display information about the command.

Options are listed in short form (if available) and long form. You can use one or two dashes for both short and long forms. An option may or may not take a value. If it takes a value, it is written like this: `--option=value`. Values can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required. If an option takes a Boolean value and no value is specified, then the option's default value is `TRUE`. Use the `--h, --help` option to display information about the command.

4.10 verifylicense

Syntax and description

The `verifylicense` command checks whether the current product is licensed. Additionally, the `--license-key` option enables you to check whether a specific license key is already assigned to the product.

```
stylevisionserver verifylicense [options]
```

- To check whether a specific license is assigned to StyleVision Server, supply the license key as the value of the `--license-key` option.

For details about licensing, see the LicenseServer documentation (<https://www.altova.com/manual/en/licenseserver/3.17/>).

▼ Casing and slashes on the command line

`StyleVisionServer` on Windows

`stylevisionserver` on Windows and Unix (Linux, Mac)

* Note that lowercase (`stylevisionserver`) works on all platforms (Windows, Linux, and Mac), while upper-lower (`StyleVisionServer`) works only on Windows and Mac.

* Use forward slashes on Linux and Mac, backslashes on Windows.

Examples

Example of the `verifylicense` command:

```
stylevisionserver verifylicense
stylevisionserver verifylicense --license-key=ABCD123-ABCD123-ABCD123-ABCD123-ABCD123-ABCD123
```

- The first command checks whether StyleVision Server is licensed.
- The second command checks whether StyleVision Server is licensed with the license key specified with the `--license-key` option.

Options

Options are listed in short form (if available) and long form. You can use one or two dashes for both short and long forms. An option may or may not take a value. If it takes a value, it is written like this: `--option=value`. Values can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required. If an option takes a Boolean value and no value is specified, then the option's default value is `TRUE`. Use the `--h, --help` option to display information about the command.

▼ license-key [l]

`--l, --license-key = Value`

Checks whether StyleVision Server is licensed with the license key specified as the value of this option.

4.11 version

Syntax and description

The `version` command displays the version number of StyleVision Server.

```
stylevisionserver version
```

▼ Casing and slashes on the command line

`stylevisionServer` on Windows

`stylevisionserver` on Windows and Unix (Linux, Mac)

* Note that lowercase (`stylevisionserver`) works on all platforms (Windows, Linux, and Mac), while upper-lower (`StylevisionServer`) works only on Windows and Mac.

* Use forward slashes on Linux and Mac, backslashes on Windows.

Example

Example of the `version` command:

```
stylevisionserver version
```

5 StyleVision Server API

StyleVision Server provides an application programming interface (API) that you can access programmatically from your .NET, COM, or Java-based code.

This reference section is organized as follows:

- [About the .NET Interface](#) ⁶⁰
- [About the COM Interface](#) ⁶¹
- [About the Java Interface](#) ⁶²
- [Code Examples](#) ⁶⁴
- [API Reference](#) ⁷²

5.1 About the .NET Interface

The .NET interface is built as a wrapper around the COM interface. It is provided as a primary interop assembly signed by Altova and uses the namespace `Altova.StyleVisionServer`.

During installation, StyleVision Server will be registered automatically as a COM server object, so there is no need for a manual registration. If you receive an access error, open the Component Services and give permissions to the same account that runs the application pool containing StyleVision Server.

In order to use StyleVision Server in your .NET project, add a reference to the `Altova.StyleVisionServer.dll` file (see the instructions below). The `Altova.StyleVisionServer.dll` is located in the `bin` folder of the StyleVision Server installation folder. This `.dll` file is automatically added to the global assembly cache (GAC) during StyleVision Server installation (the GAC is typically located in the `c:\WINDOWS\assembly` folder).

Once StyleVision Server has been registered as a COM server object, and the `Altova.StyleVisionServer.dll` is available to the .NET interface, StyleVision Server API functionality becomes available in your .NET project.

Note: If you have installed a 64-bit StyleVision Server, then the 32-bit version of `Altova.StyleVisionServer.dll` will be located in the `bin\API_32bit` folder. Similarly, if you have installed a 32-bit StyleVision Server, then the 64-bit version files of `Altova.StyleVisionServer.dll` will be located in the `bin\API_64bit` folder.

To add a reference to the StyleVision Server DLL in a Visual Studio .NET project

1. With the .NET project open in Visual Studio, click **Project | Add Reference**. The Add Reference dialog box pops up.
2. On the Browse tab, browse for the folder: `<StyleVisionServer application folder>/bin`, select `Altova.StyleVisionServer.dll`, and click **OK**.

You can view the structure of the `Altova.StyleVisionServer` assembly using the Visual Studio Object Browser (to display the Object Browser, click **Object Browser** on the **View** menu).

5.2 About the COM Interface

StyleVision Server is automatically registered as a COM server object during installation. To check whether the registration was successful, open the Registry Editor (for example, by typing `regedit.exe` command at the command line). If registration was successful, the Registry will contain the class `StyleVision.Server`. This class will typically be found under `HKEY_LOCAL_MACHINE\SOFTWARE\Classes`.

Once the COM server object is registered, you can invoke it from within applications and scripting languages that have programming support for COM calls. If you wish to change the location of the StyleVision Server installation package, it is best to uninstall StyleVision Server and then reinstall it at the required location. In this way, the necessary de-registration and registration are carried out by the installer process.

5.3 About the Java Interface

The API consists of a JAR file (`styleVisionServer.jar`) and a JNI file (`styleVisionServer.dll`). Both these files, as well as other related API files, are available in the `bin` folder of the StyleVision Server installation folder. You can either reference these files from their original location or copy them to another location if this fits your project setup. (On Windows systems, you will need administrative rights to run the program from its original location.)

Note: If you have installed a 64-bit StyleVision Server, then the 32-bit version files of `styleVisionServer.jar` and (`styleVisionServer.dll` will be located in the `bin\API_32bit` folder of the StyleVision Server installation folder. You would need these files if you are using a 32-bit Java version. Similarly, if you have installed a 32-bit StyleVision Server, then the 64-bit version files of `styleVisionServer.jar` and (`styleVisionServer.dll` will be located in the `bin\API_64bit` folder. You would need to use these files if you are using a 64-bit Java version.

To access the StyleVision Server API from Java code, add the following references to the `.classpath` file of your Java project.

<code>StyleVisionServer.jar</code>	The library that communicates with StyleVision Server
<code>StyleVisionServer_JavaDoc.zip</code>	Documentation of the StyleVision Server API

Additionally, the `java.library.path` needs to include the folder where the JNI library file (`styleVisionServer.dll`) is located.

If you deploy your project to an application server, make sure that `styleVisionServer.jar` and `styleVisionServer.dll` are correctly configured with Java on the server machine.

For an example of how to use the API's library files, see the example batch file `buildAndRun.bat` (listed below), which is located in the `etc\Examples\Java` folder of your StyleVision Server installation folder.

Build and run a Java program to use the API

For guidance about how to build and run a Java program that uses the StyleVision Server API, see the example batch file `buildAndRun.bat`. You can re-use this file to run your own Java programs by modifying it as required.

Start the batch file in a command line interface with the following command:

```
buildAndRun "path_to_Java_bin_folder"
```

Note: To check whether Java is in your classpath, you can run the command `java --version`. If Java is not in your classpath, then you must provide the path to it as a parameter of the `buildAndRun` command. If the path contains spaces, then uses quotes around the path.

Listing of `buildAndRun.bat`

```
@echo off
if %1.==. goto error

REM The location of the JAVA API binaries, the JAR file and the JNI library.
REM Adapt to your needs.
```

```
SETLOCAL
Set JavaAPIBinPath=%PROGRAMFILES%\Altova\StyleVisionServer2025\bin

REM Compile sample java
REM The -cp option (classpath) needs to point to the installed jar file (here, in its
original location)
REM "Program.java" is the Java program you want to compile
%1\javac.exe -cp "%JavaAPIBinPath%\StyleVisionServer.jar" Program.java

REM Run sample java
REM The -cp option (classpath) needs to point to the StyleVisionServer.jar file
REM The java.library.path needs to include the folder where the JNI library
StyleVisionServer.dll is located.
%1\java.exe -cp "%JavaAPIBinPath%\StyleVisionServer.jar;." -Djava.library.path="%
JavaAPIBinPath%" Program

@echo off
goto end

:error
echo Usage: buildAndRun "<path_to_java_bin_folder>"

:end
```

Adding library references in Eclipse

In Eclipse, you can add the classpath references by editing the properties of the Java project. The sample instructions below apply to Eclipse 4.4.

1. With the project open in Eclipse, on the **Project** menu, click **Properties**, and then select the Java Build Path.
2. On the Libraries tab, click **Add External JARs**, and then browse for the `StyleVisionServer.jar` file located in the StyleVision Server installation folder.
3. Under *JARs and class folders on the build path*, expand the `StyleVisionServer.jar` record, and then double-click the `Javadoc location: (None)` record.
4. Ensure that the *Javadoc in archive* and *External file* options are selected, and then browse for the `StyleVisionServer_JavaDoc.zip` file located in the StyleVision Server installation folder.
5. Click **OK**. The reference to the StyleVision Server library and Javadoc archive is added to the `.classpath` file of the project.

5.4 Code Examples

The examples in this section are for the following programming languages:

- [C#](#) ⁶⁴
- [C++](#) ⁶⁵
- [Java](#) ⁶⁷
- [VBScript](#) ⁶⁸
- [Visual Basic](#) ⁷⁰

After you have installed StyleVision Server, copy the folder `etc/Examples` into your home directory or any other suitable folder.

5.4.1 C#

The example below shows how to use C# code to generate an output RTF file using a PXF file and an input XML file. Ensure that StyleVision Server is installed and licensed and that it is available as a COM server object. Registration as a COM server object usually takes place during installation of StyleVision Server. To check if registration was successful, see [About the COM Interface](#) ⁶¹. Also see [About the .NET Interface](#) ⁶⁰.

```
namespace StyleVisionServerAPI_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            try
            {
                //Create a StyleVision Server object
                Altova.StyleVisionServer.Server objSVS = new
Altova.StyleVisionServer.Server();

                //Set a working directory - used for output and for intermediate files
                objSVS.WorkingDirectory = "..\\..\\..";

                //Default path to the StyleVision Server executable is the installation path
                (same dir with the StyleVisionServer.dll)
                //In case you moved the binaries on the disk, you need to explicitly set the
                path to the .exe file
                //objSVS.ServerPath = "C:\\Program Files (x86)\\Altova\\
\\StyleVisionServer2025\\bin\\StyleVisionServer.exe";

                //Prepare the name of the working XML
                // This can be an absolute/relative path if the file is stored externally
                (not inside PXF)
                // objSVS.InputXML = "ExpReport.xml";
                // Or it can contain the path INSIDE the PXF
                // objSVS.InputXML = "ExpReport.pxf|zip\\ExpReport.xml";
            }
        }
    }
}
```



```
// Easiest way is to refer to the file as being embedded in the
transformation file
objSVS.InputXML = "altova://packagedfile/ExpReport.xml";

//Add output paths (absolute or relative to WorkingDirectory) for all
formats that should be generated
objSVS.OutputRTF = "C:\\tmp\\ExpReport.rtf";

//Prepare the parameters, if your design uses parameters
//objSVS.AddParameter( "testparam1", "value 1" );

//Run the transformation; the output will be stored at C:\temp\ExpReport.rtf
// NOTE Please adapt the path to the input file in order to run the sample
if (objSVS.Generate("ExpReport.pxf"))
    System.Console.WriteLine("Success - finished execution");
else
    System.Console.WriteLine(objSVS.LastExecutionMessage);
}
catch (System.Runtime.InteropServices.COMException ex)
{
    // some general error like an invalid license happened
    System.Console.WriteLine("Internal Error - " + ex.Message);
}
}
}
}
```

5.4.2 C++

The example below shows how to use C++ code to generate an output RTF file using a PFX file and an input XML file. Ensure that StyleVision Server is installed and licensed and that it is available as a COM server object. Registration as a COM server object usually takes place during installation of StyleVision Server. To check if registration was successful, see [About the COM Interface](#)⁶¹. Also see [About the .NET Interface](#)⁶⁰.

```
// StyleVisionServerAPI_Sample.cpp : Defines the entry point for the console application.
//
#include <iostream>
#include "atlbase.h"

// The following import statements require the corresponding C++ tool-chain to be selected
in the project configuration file.
#ifdef _WIN64
// 32-bit StyleVisionServer
#import "progid:StyleVision.Server"
#else
// 64-bit StyleVisionServer
#import "progid:StyleVision_x64.Server"
#endif
```

```

int _tmain(int argc, _TCHAR* argv[])
{
    CoInitialize( NULL );

    try
    {
        //Create a StyleVision Server object
        StyleVisionServerLib::IServerPtr pSVS;
        CoCreateInstance( __uuidof( StyleVisionServerLib::Server ), NULL, CLSCTX_ALL,
__uuidof( StyleVisionServerLib::IServer ), reinterpret_cast< void** >( &pSVS ) );

        //Set a working directory - used for output and for intermediate files
        pSVS->WorkingDirectory = ".."; // this is relative to this applications' working
directory (the project folder)

        //Default path to the StyleVision Server executable is the installation path (same dir
with the StyleVisionServer.dll)
        //In case you moved the binaries on the disk, you need to explicitly set the path to
the .exe file
        //pSVS->ServerPath = "C:\\Program Files (x86)\\Altova\\StyleVisionServer2025\\bin\\
\\StyleVisionServer.exe";
        //pSVS->ServerPath = "C:\\Program Files\\Altova\\StyleVisionServer2025\\bin\\
\\StyleVisionServer.exe";

        //Prepare the name of the working XML
        // This can be an absolute/relative path if the file is stored externally (not
inside PXF)
        // pSVS->InputXML = "ExpReport.xml";
        // Or it can contain the path INSIDE the PXF
        // pSVS->InputXML = "ExpReport.pxf|zip\\ExpReport.xml";
        // Easiest way is to refer to the file as being embedded in the transformation file
        pSVS->InputXML = "altova://packagedfile/ExpReport.xml";

        //Add output paths (absolute or relative to WorkingDirectory) for all formats that
should be generated
        pSVS->OutputRTF = "ExpReport.rtf";
        pSVS->OutputPDF = "ExpReport.pdfrtf";
        pSVS->OutputHTML = "ExpReport.html";

        //Prepare the parameters, if your design uses parameters
        //pSVS->AddParameter( "testparam1", "value 1" );

        //Run the transformation; the output will be stored at C:\\temp\\ExpReport.rtf
        // NOTE Please adapt the path to the input file in order to run the sample
        if (pSVS->Generate("ExpReport.pxf"))
        {
            std::cout << pSVS->LastExecutionMessage << std::endl;
            std::cout << "Success - finished execution" << std::endl;
        }
        else
            std::cout << pSVS->LastExecutionMessage << std::endl;
    }
}

```

```

}
catch (_com_error& err )
{
    BSTR bstrMessage;
    (err).ErrorInfo()->GetDescription( &bstrMessage );
    std::cout << "Exception occurred: " << _com_util::ConvertBSTRToString( bstrMessage )
<< std::endl;
}

CoUninitialize();
return 0;
}

```

5.4.3 Java

The example below shows how to use Java code to generate an output RTF file using a PXF file and an input XML file. Ensure that StyleVision Server is installed and licensed and that it is available as a server object. Registration as a server object usually takes place during installation of StyleVision Server. The example program below can be built and run using the batch file named `buildAndRun.bat`, which is located in the `etc\Examples\Java` folder of your StyleVision Server installation folder.

For information about the interface, see [About the Java Interface](#)⁶².

Note: We recommend that you copy the Examples folder to your home directory or any other suitable folder, and then navigate to the Java folder to access `Program.java` and `buildAndRun.bat`.

Program.java

```

public class Program
{

    public static void main(String[] args)
    {
        com.altova.stylevisionserver.StyleVisionServer objSVS
        try
        {
            //Create a StyleVision Server object
            objSVS = new com.altova.stylevisionserver.StyleVisionServer();

            //The default location of server binary is the folder containing the Java native
library
            //Select a different server binary with the following line:
            //obj.SVS.setServerPath(strServerPath);

            //The sample data is located in the parent folder of the Java sample code
            //Set this parent folder to be the working directory:
            objSVS.setWorkingDirectory( ".." );

            System.out.println("Running " + objSVS.getProductNameAndVersion());

            //Prepare the name of the working XML

```

```

        //This can be an absolute/relative path if the file is external (not inside PXF)
        //  objSVS.setInputXML( "ExpReport.xml" );
        //Or it can contain the path INSIDE the PXF
        //  objSVS.setInputXML( "C:\\Program Files (x86)\\Altova\\StyleVisionServer" +
majorVersionYear + "\\etc\\Examples\\ExpReport.pxf|zip\\ExpReport.xml" );
        //The easiest way is to refer to the file as being embedded in the transformation
file
        objSVS.setInputXML( "altova://packagedfile/ExpReport.xml" );

        //Add output paths (absolute or relative to WorkingDirectory) for all formats
that should be generated
        objSVS.setOutputRTF( "ExpReport.rtf" );

        //Prepare the parameters, if your design uses parameters
        //objSVS.AddParameter( "testparam1", "value 1" );

        //Run the transformation; the output will be stored at C:\\temp\\ExpReport.rtf
        // NOTE Please adapt the path to the input file in order to run the sample
        if ( objSVS.generate( "ExpReport.pxf" ) )
            System.out.println( "Success" + objSVS.getLastExecutionMessage() );
        else
            System.out.println( objSVS.getLastExecutionMessage() );
    }
    catch ( Exception e )
    {
        System.out.println(e.getMessage());
    }
}
}

```

5.4.4 VBScript

The example below shows how to use VB Script code to generate an output RTF file using a PXF file and an input XML file. Ensure that StyleVision Server is installed and licensed and that it is available as a COM server object. Registration as a COM server object usually takes place during installation of StyleVision Server. To check if registration was successful, see [About the COM Interface](#) ⁶¹.

Option Explicit

```

'Create a StyleVision Server object; use "StyleVision_x64.Server" if you want to use the
64-bit installation
Dim objSVS
' Since we load a COM-DLL we need care about the process architecture
On Error Resume Next ' ignore any COM errors avoiding uncontrolled script termination
Dim WshShell
Dim WshProcEnv
Set WshShell = CreateObject("WScript.Shell")
Set WshProcEnv = WshShell.Environment("Process")
Dim process_architecture

```

```
process_architecture= WshProcEnv("PROCESSOR_ARCHITECTURE")
If process_architecture = "x86" Then
  Set objSVS = WScript.GetObject( "", "StyleVision.Server" )
  If Err.Number <> 0 then
    WScript.Echo("You are running in a 32-bit process but StyleVision Server COM-API 32-
bit seems not to be installed on your system.")
    WScript.Quit -1
  End If
Else
  Set objSVS = WScript.GetObject( "", "StyleVision_x64.Server" )
  If Err.Number <> 0 then
    WScript.Echo("You are running in a 64-bit process but StyleVision Server COM-API 64-
bit seems not to be installed on your system.")
    WScript.Echo("If you have installed 32-bit StyleVision Server consider calling your
script from the 32-bit console 'C:\Windows\SysWOW64\cmd.exe.'")
    WScript.Quit -1
  End If
End If
On Error Goto 0      ' re-enable default error promotion

'Set a working directory - used for input, output and for intermediate files
'objSVS.WorkingDirectory = "C:\Program Files (x86)
\Altova\StyleVisionServer2020\etc\examples"
objSVS.WorkingDirectory = ".."

'Default path to the StyleVision Server executable is the installation path (same dir with
the StyleVisionServer.dll)
'In case you moved the binaries on the disk, you need to explicitly set the path to the
.exe file
'objSVS.ServerPath = "C:\Program Files (x86)
\Altova\StyleVisionServer2020\bin\StyleVisionServer_DebugDLL.exe"

' The Generate method will return 'True' if generation was successful otherwise 'False'.
' In the case of fundamental errors like termination of the server process a COM error will
be raised which
' can be handled using the VBScript Err object.
On Error Resume Next ' ignore any COM errors avoiding uncontrolled script termination
Err.Clear

WScript.Echo("Running " & objSVS.ProductNameAndVersion & vbCrLf)

'Prepare the name of the working XML
' This can be an absolute/relative path if the file is stored externally (not inside
PXF)
' objSVS.InputXML = "ExpReport.xml"
' or it can contain the path INSIDE the PXF
objSVS.InputXML = "ExpReport.pxf|zip\ExpReport.xml"
' or refer to the file as being embedded in the transformation file
'objSVS.InputXML = "altova://packagedfile/ExpReport.xml"

'Add output paths (absolute or relative to WorkingDirectory) for all formats that should be
generated
' make sure you have write permissions
```

```

'objSVS.OutputRTF = "C:\tmp\ExpReport.rtf"
objSVS.OutputPDF = "C:\tmp\ExpReport.pdf"
'objSVS.OutputHTML = "C:\tmp\ExpReport.html"

'Prepare the parameters, if your design uses parameters
'Call objSVS.AddParameter( "testparam1", "value_1" )

' Run the transformation. The PXF file path can be relative to the working folder or
absolute.
WScript.Echo("Generating output from ExpReport.pxf...")
If ( objSVS.Generate( "ExpReport.pxf" ) ) Then
    WScript.Echo( objSVS.LastExecutionMessage )
    WScript.Echo( "Success - finished execution" )
Else
    WScript.Echo( objSVS.LastExecutionMessage )
End If

' handle COM errors
If Err.Number <> 0 Then
    WScript.Echo("Internal error - " & Err.Description )
    WScript.Quit -1
End If

On Error Goto 0      ' re-enable default error promotion

```

5.4.5 Visual Basic

The example below shows how to use Visual Basic code to generate an output RTF file using a PXF file and an input XML file. Ensure that StyleVision Server is installed and licensed and that it is available as a COM server object. Registration as a COM server object usually takes place during installation of StyleVision Server. To check if registration was successful, see [About the COM Interface](#) ⁶¹.

Option Explicit On

Module Program

Sub Main()

Try

'Create a StyleVision Server object

Dim objSVS As Altova.StyleVisionServer.Server = New

Altova.StyleVisionServer.Server

'Set a working directory - used for output and for intermediate files

objSVS.WorkingDirectory = "C:\Program Files (x86)

\Altova\MapForceServer2020\etc\Examples"

objSVS.WorkingDirectory = "..\..\..\."

```
'Default path to the StyleVision Server executable is the installation path
(same dir with the StyleVisionServer.dll)
'In case you moved the binaries on the disk, you need to explicitly set the
path to the .exe file
'objSVS.ServerPath = "C:\Program Files (x86)
\Altova\StyleVisionServer2020\bin\StyleVisionServer.exe"
'objSVS.ServerPath = "C:\Program
Files\Altova\StyleVisionServer2020\bin\StyleVisionServer.exe"

'Prepare the name of the working XML
' This can be an absolute/relative path if the file is stored externally
(not inside PXF)
' objSVS.InputXML = "ExpReport.xml"
' Or it can contain the path INSIDE the PXF
objSVS.InputXML = "ExpReport.pxf|zip\ExpReport.xml"
' Easiest way is to refer to the file as being embedded in the
transformation file
' objSVS.InputXML = "altova://packagedfile/ExpReport.xml"

'Add output paths (absolute or relative to WorkingDirectory) for all formats
that should be generated
objSVS.OutputRTF = "C:\tmp\ExpReport.rtf"
objSVS.OutputPDF = "C:\tmp\ExpReport.pdf"
objSVS.OutputHTML = "C:\tmp\ExpReport.html"

'Prepare the parameters, if your design uses parameters
'objSVS.AddParameter( "testparam1", "value 1" )

' Run the transformation; the output will be stored at C:\temp
If (objSVS.Generate("ExpReport.pxf")) Then
    System.Console.WriteLine(objSVS.LastExecutionMessage)
    System.Console.WriteLine("Success - finished execution")
Else
    System.Console.WriteLine(objSVS.LastExecutionMessage)
End If

Catch ex As Exception
    System.Console.WriteLine("Internal Error - " & ex.Message())
End Try

End Sub

End Module
```

5.5 API Reference

This section is a user's reference for the StyleVision Server API.

- [COM and .NET](#)⁷²
- [Java](#)⁷⁹

5.5.1 COM and .NET

The StyleVisionServer API exposes the **IServer interface**, which creates a new StyleVision Server object instance, and provides access to StyleVision Server.

The `IServer` interface has the following methods and properties.

Methods

▼ AddParameter

Assigns a value to a parameter defined in the PXF file.

▣ C#

```
void AddParameter(string bstrName, string bstrValue)
```

▣ C++

```
HRESULT AddParameter([in] BSTR bstrName, [in] BSTR bstrValue );
```

▣ VB

```
Sub AddParameter(ByVal bstrName As String, ByVal bstrValue As String)
```

▼ ClearParameterList

Clears the list of parameters.

▣ C#

```
void ClearParameterList()
```

▣ C++

```
HRESULT ClearParameterList();
```

▣ VB

```
Sub ClearParameterList()
```

▼ Generate

Generates one or more output files (HTML, PDF, RTF, and/or DOCX) by using the PXF file specified with `TransfPath`. It transforms the input XML file (Working XML File in the PXF file) using the XSLT document contained in the PXF file. Returns `TRUE` in case of success; `FALSE` otherwise.

▣ C#

```
bool Generate(string bstrTransfPath)
```

▣ C++

```
HRESULT Generate( [in] BSTR bstrTransfPath, [out, retval] VARIANT_BOOL*  
pbSuccess );
```

▣ VB

```
Function Generate(ByVal bstrTransfPath As String) As Boolean
```

Properties

▼ APIMajorVersion

Gets the major version of the StyleVision Server API. It can be different from the product version if the API is connected to another server.

▣ C#

```
int APIMajorVersion { get; }
```

▣ C++

```
HRESULT APIMajorVersion([out, retval] INT* pnVal);
```

▣ VB

```
ReadOnly Property APIMajorVersion As Integer
```

▼ APIMinorVersion

Gets the minor version of the StyleVision Server API. It can be different from the product version if the API is connected to another server.

▣ C#

```
int APIMinorVersion { get; }
```

▣ C++

```
HRESULT APIMinorVersion([out, retval] INT* pnVal);
```

▣ VB

```
ReadOnly Property APIMinorVersion As Integer
```

▼ APIServicePackVersion

Gets the service pack version of the StyleVision Server API. It can be different from the product version if the API is connected to another server.

▣ C#

```
int APIServicePackVersion { get; }
```

▣ C++

```
HRESULT APIServicePackVersion([out, retval] INT* pnVal);
```

▣ VB

```
ReadOnly Property APIServicePackVersion As Integer
```

▼ InputXML

Sets the path and name of the XML file to be processed (the Working XML File in the PXF file).

▣ C#

```
string InputXML { set; }
```

▣ C++

```
HRESULT InputXML([in] BSTR bstrPath );
```

VB

Property InputXML As String

▼ Is64Bit

Returns `TRUE` if the StyleVision Server engine is a 64-bit executable.

C#

```
bool Is64Bit { get; }
```

C++

```
HRESULT Is64Bit([out, retval] VARIANT_BOOL* pbVal);
```

VB

ReadOnly Property Is64Bit As Boolean

▼ LastExecutionMessage

Gets the message received during the last `Generate` command.

C#

```
string LastExecutionMessage { get; }
```

C++

```
HRESULT LastExecutionMessage([out, retval] BSTR* pbstrResult );
```

VB

ReadOnly Property LastExecutionMessage As String

▼ MajorVersion

Gets the major version of StyleVision Server.

C#

```
int MajorVersion { get; }
```

C++

```
HRESULT MajorVersion([out, retval] INT* pnVal);
```

VB

ReadOnly Property MajorVersion As Integer

▼ MinorVersion

Gets the minor version of StyleVision Server.

- ▣ C#
int MinorVersion { get; }
- ▣ C++
HRESULT MinorVersion([out, retval] INT* pnVal);
- ▣ VB
ReadOnly Property MinorVersion As Integer

▼ OutputDOCX

Sets the path and name of the output DOCX file.

- ▣ C#
string OutputDOCX { set; }
- ▣ C++
HRESULT OutputDOCX([in] BSTR bstrPath);
- ▣ VB
Property OutputDOCX As String

▼ OutputFO

Sets the path and name of the output FO file.

- ▣ C#
string OutputFO { set; }
- ▣ C++
HRESULT OutputFO([in] BSTR bstrPath);
- ▣ VB
Property OutputFO As String

▼ OutputHTML

Sets the path and name of the output HTML file.

- ▣ C#
string OutputHTML { set; }
- ▣ C++
HRESULT OutputHTML([in] BSTR bstrPath);
- ▣ VB
Property OutputHTML As String

▼ OutputPDF

Sets the path and name of the output PDF file.

▣ C#

```
string OutputPDF { set; }
```

▣ C++

```
HRESULT OutputPDF([in] BSTR bstrPath );
```

▣ VB

```
Property OutputPDF As String
```

▼ OutputRTF

Sets the path and name of the output RTF file.

▣ C#

```
string OutputRTF { set; }
```

▣ C++

```
HRESULT OutputRTF([in] BSTR bstrPath );
```

▣ VB

```
Property OutputRTF As String
```

▼ OutputText

Sets the path and name of the output Text file.

▣ C#

```
string OutputText { set; }
```

▣ C++

```
HRESULT OutputText([in] BSTR bstrPath );
```

▣ VB

```
Property OutputText As String
```

▼ ProductName

Gets the name of the product: "StyleVision Server"

▣ C#

```
string ProductName { get; }
```

▣ C++

```
HRESULT ProductName([out, retval] BSTR* pstrVal);
```

VB

```
ReadOnly Property ProductName As String
```

▼ ProductNameAndVersion

Gets the complete name of the product, including the version number: "StyleVision Server 2014r2 sp1 (x64)".

C#

```
string ProductNameAndVersion { get; }
```

C++

```
HRESULT ProductNameAndVersion([out, retval] BSTR* pstrVal);
```

VB

```
ReadOnly Property ProductNameAndVersion As String
```

▼ ServerPath

Gets or sets the path to the StyleVision Server executable.

C#

```
string ServerPath { set; get; }
```

C++

```
HRESULT ServerPath([in] BSTR bstrServerFile );  
HRESULT ServerPath([out, retval] BSTR* pbstrServerFile );
```

VB

```
Property ServerPath As String
```

▼ ServicePackVersion

Gets the service pack version of StyleVision Server (for example: **1** for Altova StyleVision Server 2014 r2 sp1 (x64).)

▣ C#

```
int ServicePackVersion { get; }
```

▣ C++

```
HRESULT ServicePackVersion([out, retval] INT* pnVal);
```

▣ VB

```
ReadOnly Property ServicePackVersion As Integer
```

▼ WhereClause

Sets an SQL `WHERE` clause that determines the rows of a DB-XML schema source to process.

▣ C#

```
string WhereClause { set; }
```

▣ C++

```
HRESULT WhereClause([in] BSTR bstrPath );
```

▣ VB

```
Property WhereClause As String
```

▼ WorkingDirectory

Gets or sets the current directory for running jobs. Relative paths are evaluated against the working directory.

▣ C#

```
string WorkingDirectory { set; get; }
```

▣ C++

```
HRESULT WorkingDirectory([in] BSTR bstrWorkingDirectory );
HRESULT WorkingDirectory([out, retval] BSTR* pbstrWorkingDirectory );
```

▣ VB

```
Property WorkingDirectory As String
```

5.5.2 Java

The package `com.altova.stylevisionserver` consists of the following classes:

- `public class StyleVisionServer (described below)`

- `public class StyleVisionServerException` extends `Exception`

StyleVisionServer class

The `StyleVisionServer` class creates a new StyleVision Server object instance, and provides access to StyleVision Server. The methods of the `StyleVisionServer` interface are described below.

Methods of styleVisionServer class

The methods of the `StyleVisionServer` class are listed alphabetically below.

▼ addParameter

```
public void addParameter(String name, String value)
```

Adds the name and value of a new parameter. Each parameter and its value is specified in a separate call to the method. Parameters must be declared in the XSLT document.

Parameters:

name: Holds the name of the parameter as a string.

value: Holds the value of the parameter as a string.

▼ clearParameterList

```
public void clearParameterList()
```

Clears the list of parameters.

▼ generate

```
public boolean generate(String transfPath)
```

Processes the PXF file specified in `transfPath`. Throws `StyleVisionServerException`.

Parameters:

`transfPath`: An absolute URL giving the location of the PXF file.

Returns:

`true()` if execution is successful

`false()` if execution fails

In case of an error, use `getLastExecutionMessage()`

▼ getAPIMajorVersion

```
public int getAPIMajorVersion()
```

Gets the major version of the StyleVision Server API. It can be different from the product version if the API is connected to another server.

▼ getAPIMinorVersion

```
public int getAPIMinorVersion()
```

Gets the minor version of the StyleVision Server API. It can be different from the product version if the API is connected to another server.

▼ getAPIServicePackVersion

```
public int getAPIServicePackVersion()
```

Gets the service pack version of the StyleVision Server API. It can be different from the product version if the API is connected to another server.

▼ getLastExecutionMessage

```
public String getLastExecutionMessage()
```

Gets the message received during the last generate command.

▼ getMajorVersion

```
public int getMajorVersion()
```

Gets the major version of the application.

▼ getMinorVersion

```
public int getMinorVersion()
```

Gets the minor version of the application.

▼ getProductName

```
public String getProductName()
```

Gets the product name.

▼ getProductNameAndVersion

```
public String getProductNameAndVersion()
```

Gets the complete name and version number of the product.

▼ getServerPath

```
public String getServerPath()
```

Gets the path to the server's binary executable file.

▼ getServicePackVersion

```
public int getServicePackVersion()
```

Gets the service pack version of the StyleVision Server.

▼ getWorkingDirectory

```
public String getWorkingDirectory()
```

Gets the current working directory.

▼ is64bit

```
public boolean is64bit()
```

Checks whether the executable is 64-bit.

Returns:

true() for StyleVision Server (x64), false() otherwise.

▼ setInputXML

```
public void setInputXML(String path)
```

Sets the XML file to process. This must be the path of the Working XML File that is specified in the PXF file.

Parameters:

path: Holds the path of the Working XML file in the PXF file.

▼ **setOutputDOCX**

```
public void setOutputDOCX(String path)
```

Sets the path and name of the DOCX file to generate.

Parameters:

path: The path and name of the DOCX file to generate.

▼ **setOutputFO**

```
public void setOutputFO(String path)
```

Sets the path and name of the FO file to generate.

Parameters:

path: The path and name of the FO file to generate.

▼ **setOutputHTML**

```
public void setOutputHTML(String path)
```

Sets the path and name of the HTML file to generate.

Parameters:

path: The path and name of the HTML file to generate.

▼ **setOutputPDF**

```
public void setOutputPDF(String path)
```

Sets the path and name of the PDF file to generate.

Parameters:

path: The path and name of the PDF file to generate.

▼ **setOutputRTF**

```
public void setOutputRTF(String path)
```

Sets the path and name of the RTF file to generate.

Parameters:

path: The path and name of the RTF file to generate.

▼ **setOutputText**

```
public void setOutputText(String path)
```

Sets the path and name of the Text file to generate.

Parameters:

path: The path and name of the Text file to generate.

▼ **setServerPath**

```
public void setServerPath(String serverFile)
```

Sets the path of the StyleVisionServer executable.

Parameters:

serverFile: The path of the StyleVisionServer executable.

▼ **setWhereClause**

```
public void setWhereClause(String whereClause)
```

Sets an SQL `WHERE` clause that determines the rows of a DB-XML schema source to process.

Parameters:

`whereClause`: The SQL `WHERE` clause that determines the rows of a DB-XML schema source to process.

▼ `setWorkingDirectory`

```
public void setWorkingDirectory(String workingDirectory)
```

Sets a default directory. Relative paths are resolved relative to this directory.

Parameters:

`workingDirectory`: The path of the default (working) directory.

▼ `stop`

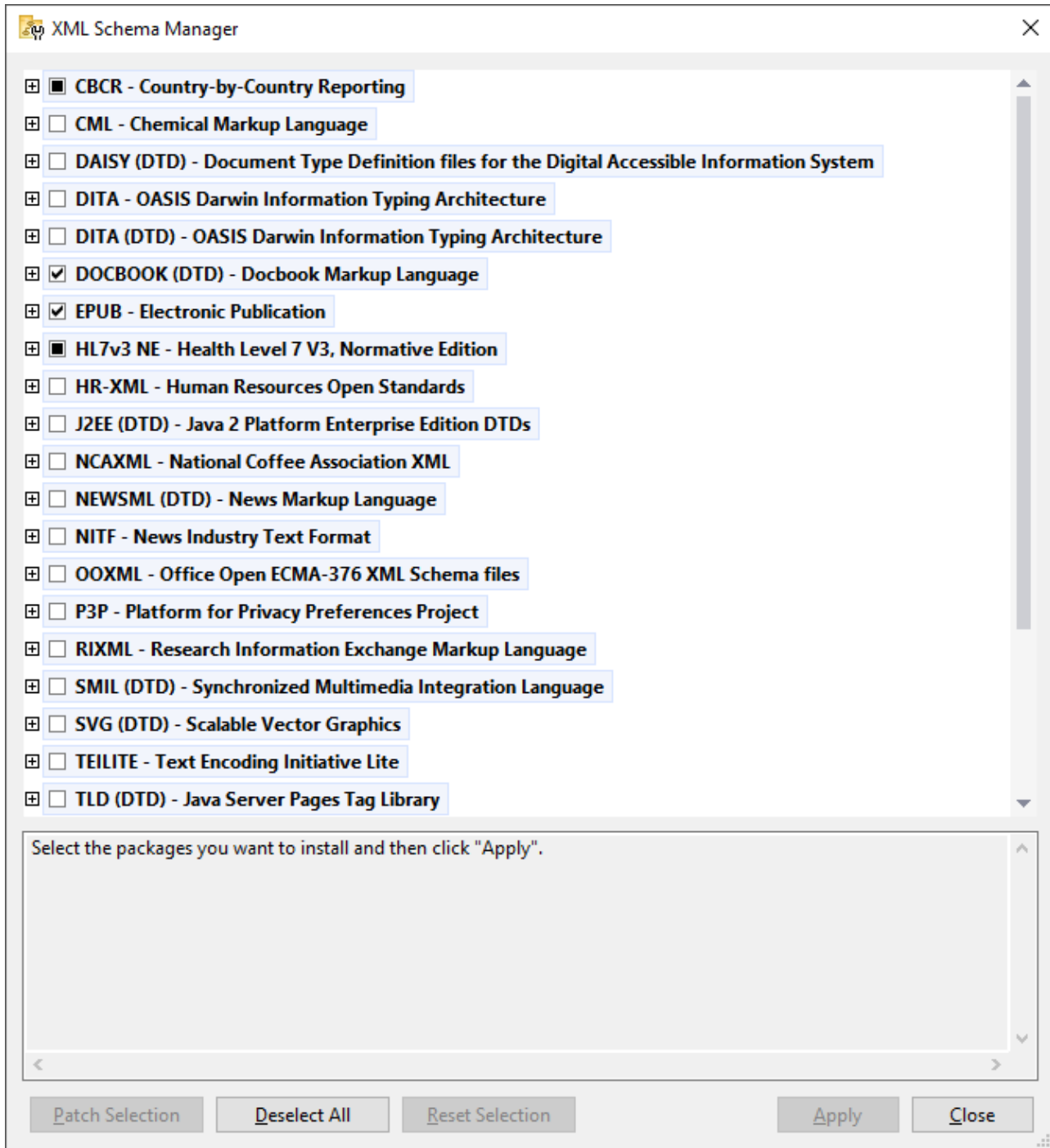
```
public void stop()
```

Stops the server process.

6 Schema Manager

XML Schema Manager is an Altova tool that provides a centralized way to install and manage XML schemas (DTDs for XML and XML Schemas) for use across all Altova's XML-Schema-aware applications, including StyleVision Server.

- On Windows, Schema Manager has a graphical user interface (*screenshot below*) and is also available at the command line. (Altova's desktop applications are available on Windows only; *see list below*.)
- On Linux and macOS, Schema Manager is available at the command line only. (Altova's server applications are available on Windows, Linux, and macOS; *see list below*.)



Altova applications that operate with Schema Manager

Desktop applications (Windows only)	Server applications (Windows, Linux, macOS)
XMLSpy (all editions)	RaptorXML Server, RaptorXML+XBRL Server

MapForce (all editions)	StyleVision Server
StyleVision (all editions)	
Authentic Desktop Enterprise Edition	

Installation and de-installation of Schema Manager

Schema Manager is installed automatically when you first install a new version of Altova Mission Kit or of any of Altova's XML-schema-aware applications (see *table above*).

Likewise, it is removed automatically when you uninstall the last Altova XML-schema-aware application from your computer.

Schema Manager features

Schema Manager provides the following features:

- Shows XML schemas installed on your computer and checks whether new versions are available for download.
- Downloads newer versions of XML schemas independently of the Altova product release cycle. (Altova stores schemas online, and you can download them via Schema Manager.)
- Install or uninstall any of the multiple versions of a given schema (or all versions if necessary).
- An XML schema may have dependencies on other schemas. When you install or uninstall a particular schema, Schema Manager informs you about dependent schemas and will automatically install or remove them as well.
- Schema Manager uses the [XML catalog](#) mechanism to map schema references to local files. In the case of large XML schemas, processing will therefore be faster than if the schemas were at a remote location.
- All major schemas are available via Schema Manager and are regularly updated for the latest versions. This provides you with a convenient single resource for managing all your schemas and making them readily available to all of Altova's XML-schema-aware applications.
- Changes made in Schema Manager take effect for all Altova products installed on that machine.
- In an Altova product, if you attempt to validate on a schema that is not installed but which is available via Schema Manager, then installation is triggered automatically. However, if the schema package contains namespace mappings, then there will be no automatic installation; in this case, you must start Schema Manager, select the package/s you want to install, and run the installation. If, after installation, your open Altova application does not restart automatically, then you must restart it manually.

How it works

Altova stores all XML schemas used in Altova products online. This repository is updated when new versions of the schemas are released. Schema Manager displays information about the latest available schemas when invoked in both its GUI form as well as on the CLI. You can then install, upgrade or uninstall schemas via Schema Manager.

Schema Manager also installs schemas in one other way. At the Altova website (<https://www.altova.com/schema-manager>) you can select a schema and its dependent schemas that you want to install. The website will prepare a file of type `.altova_xmlschemas` for download that contains information about your schema selection. When you double-click this file or pass it to Schema Manager via the CLI as an argument of the `install` ⁹⁸ command, Schema Manager will install the schemas you selected.

Local cache: tracking your schemas

All information about installed schemas is tracked in a centralized cache directory on your computer, located here:

<i>Windows</i>	C:\ProgramData\Altova\pkgs\.cache
<i>Linux</i>	/var/opt/Altova/pkgs\.cache
<i>macOS</i>	/var/Altova/pkgs

This cache directory is updated regularly with the latest status of schemas at Altova's online storage. These updates are carried out at the following times:

- Every time you start Schema Manager.
- When you start StyleVision Server for the first time on a given calendar day.
- If StyleVision Server is open for more than 24 hours, the cache is updated every 24 hours.
- You can also update the cache by running the [update](#)¹⁰¹ command at the command line interface.

The cache therefore enables Schema Manager to continuously track your installed schemas against the schemas available online at the Altova website.

Do not modify the cache manually!

The local cache directory is maintained automatically based on the schemas you install and uninstall. It should not be altered or deleted manually. If you ever need to reset Schema Manager to its original "pristine" state, then, on the command line interface (CLI): (i) run the [reset](#)⁹⁹ command, and (ii) run the [initialize](#)⁹⁷ command. (Alternatively, run the `reset` command with the `--i` option.)

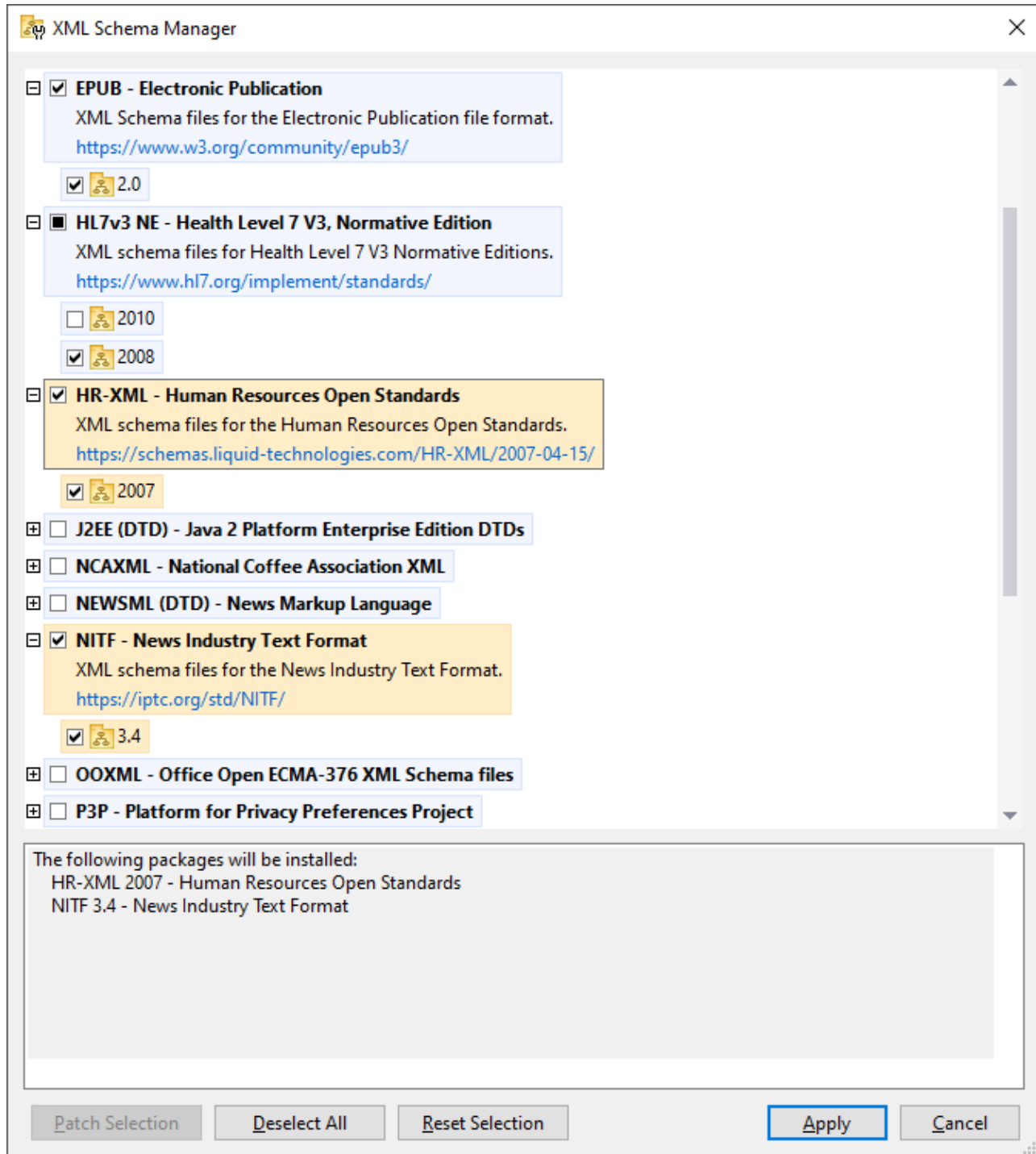
6.1 Run Schema Manager

Graphical User Interface

You can access the GUI of Schema Manager in any of the following ways:

- *During the installation of StyleVision Server:* Towards the end of the installation procedure, select the check box *Invoke Altova XML-Schema Manager* to access the Schema Manager GUI straight away. This will enable you to install schemas during the installation process of your Altova application.
- Via the `.altova_xmlschemas` file downloaded from the [Altova website](#): Double-click the downloaded file to run the Schema Manager GUI, which will be set up to install the schemas you selected (at the website) for installation.

After the Schema Manager GUI (*screenshot below*) has been opened, already installed schemas will be shown selected. If you want to install an additional schema, select it. If you want to uninstall an already installed schema, deselect it. After you have made your selections and/or deselections, you are ready to apply your changes. The schemas that will be installed or uninstalled will be highlighted and a message about the upcoming changes will be posted to the Messages pane at the bottom of the Schema Manager window (see *screenshot*).



When you click **Apply**, the progress of the installation is displayed. If there is an error (for example, a connection error), then an error message is displayed. In this case, click the **Advanced** button that appears in the dialog, check the schema selection and retry with **Apply**.

Command line interface

You can run Schema Manager from a command line interface by sending commands to its executable file, `xmlschemamanager.exe`.

The `xmlschemamanager.exe` file is located in the following folder:

- *On Windows:* `C:\ProgramData\Altova\SharedBetweenVersions`
- *On Linux or macOS (server applications only):* `%INSTALLDIR%/bin`, where `%INSTALLDIR%` is the program's installation directory.

You can then use any of the commands listed in the [CLI command reference section](#) ⁹⁶.

To display help for the commands, run the following:

- *On Windows:* `xmlschemamanager.exe --help`
- *On Linux or macOS (server applications only):* `sudo ./xmlschemamanager --help`

6.2 Status Categories

Schema Manager categorizes the schemas under its management as follows:

- *Installed schemas.* These are shown in the GUI with their check boxes selected (*in the screenshot below the checked and blue versions of the EPUB and HL7v3 NE schemas are installed schemas*). If all the versions of a schema are selected, then the selection mark is a tick. If at least one version is unselected, then the selection mark is a solid colored square. You can deselect an installed schema to **uninstall** it; (*in the screenshot below, the DocBook DTD is installed and has been deselected, thereby preparing it for de-installation*).
- *Uninstalled available schemas.* These are shown in the GUI with their check boxes unselected. You can select the schemas you want to **install**.



- *Upgradeable schemas* are those which have been revised by their issuers since they were installed. They are indicated in the GUI by a 🔄 icon. You can **patch** an installed schema with an available revision.

Points to note

- In the screenshot above, both CBCR schemas are checked. The one with the blue background is already installed. The one with the yellow background is uninstalled and has been selected for installation. Note that the HL7v3 NE 2010 schema is not installed and has not been selected for installation.
- A yellow background means that the schema will be modified in some way when the **Apply** button is clicked. If a schema is unchecked and has a yellow background, it means that it will be uninstalled when the **Apply** button is clicked. In the screenshot above the DocBook DTD has such a status.

- When running Schema Manager from the command line, the [list](#)⁹⁸ command is used with different options to list different categories of schemas:

<code>xmlschemamanager.exe list</code>	Lists all installed and available schemas; upgradeables are also indicated
<code>xmlschemamanager.exe list -i</code>	Lists installed schemas only; upgradeables are also indicated
<code>xmlschemamanager.exe list -u</code>	Lists upgradeable schemas




Note: On Linux and macOS, use `sudo ./xmlschemamanager list`

6.3 Patch or Install a Schema

Patch an installed schema

Occasionally, XML schemas may receive patches (upgrades or revisions) from their issuers. When Schema Manager detects that patches are available, these are indicated in the schema listings of Schema Manager and you can install the patches quickly.

In the GUI

Patches are indicated by the  icon. (Also see the previous topic about [status categories](#)⁹¹.) If patches are available, the **Patch Selection** button will be enabled. Click it to select and prepare all patches for installation. In the GUI, the icon of each schema that will be patched changes from  to , and the Messages pane at the bottom of the dialog lists the patches that will be applied. When you are ready to install the selected patches, click **Apply**. All patches will be applied together. Note that if you deselect a schema marked for patching, you will actually be uninstalling that schema.

On the CLI

To apply a patch at the command line interface:

1. Run the `list -u`⁹⁸ command. This lists any schemas for which upgrades are available.
2. Run the `upgrade`¹⁰¹ command to install all the patches.

Install an available schema

You can install schemas using either the Schema Manager GUI or by sending Schema Manager the install instructions via the command line.

Note: If the current schema references other schemas, the referenced schemas are also installed.

In the GUI

To install schemas using the Schema Manager GUI, select the schemas you want to install and click **Apply**.

You can also select the schemas you want to install at the [Altova website](#) and generate a downloadable `.altova_xmlschemas` file. When you double-click this file, it will open Schema Manager with the schemas you wanted pre-selected. All you will now have to do is click **Apply**.

On the CLI

To install schemas via the command line, run the `install`⁹⁸ command:

```
xmlschemamanager.exe install [options] Schema+
```

where `schema` is the schema (or schemas) you want to install or a `.altova_xmlschemas` file. A schema is referenced by an identifier of format `<name>-<version>`. (The identifiers of schemas are displayed when you run the `list`⁹⁸ command.) You can enter as many schemas as you like. For details, see the description of the `install`⁹⁸ command.

Note: On Linux or macOS, use the `sudo ./xmlschemamanager` command.

Installing a required schema

When you run an XML-schema-related command in StyleVision Server and StyleVision Server discovers that a schema it needs for executing the command is not present or is incomplete, Schema Manager will display information about the missing schema/s. You can then directly install any missing schema via Schema Manager.

In the Schema Manager GUI, you can view all previously installed schemas at any time by running Schema Manager from **Tools | Schema Manager**.

6.4 Uninstall a Schema, Reset

Uninstall a schema

You can uninstall schemas using either the Schema Manager GUI or by sending Schema Manager the uninstall instructions via the command line.

Note: If the schema you want to uninstall references other schemas, then the referenced schemas are also uninstalled.

In the GUI

To uninstall schemas in the Schema Manager GUI, clear their check boxes and click **Apply**. The selected schemas and their referenced schemas will be uninstalled.

To uninstall all schemas, click **Deselect All** and click **Apply**.

On the CLI

To uninstall schemas via the command line, run the [uninstall](#)¹⁰⁰ command:

```
xmlschemamanager.exe uninstall [options] Schema+
```

where each `schema` argument is a schema you want to uninstall or a `.altova_xmlschemas` file. A schema is specified by an identifier that has a format of `<name>-<version>`. (The identifiers of schemas are displayed when you run the [list](#)⁹⁸ command.) You can enter as many schemas as you like. For details, see the description of the [uninstall](#)¹⁰⁰ command.

Note: On Linux or macOS, use the `sudo ./xmlschemamanager` command.

Reset Schema Manager

You can reset Schema Manager. This removes all installed schemas and the cache directory.

- In the GUI, click **Reset Selection**.
- On the CLI, run the [reset](#)⁹⁹ command.

After running this command, make sure to run the [initialize](#)⁹⁷ command in order to recreate the cache directory. Alternatively, run the [reset](#)⁹⁹ command with the `-i` option.

Note that [reset -i](#)⁹⁹ restores the original installation of the product, so it is recommended to run the [update](#)¹⁰¹ command after performing a reset. Alternatively, run the [reset](#)⁹⁹ command with the `-i` and `-u` options.

6.5 Command Line Interface (CLI)

To call Schema Manager at the command line, you need to know the path of the executable. By default, the Schema Manager executable is installed here:

<i>Windows</i>	C:\ProgramData\Altova\SharedBetweenVersions\XMLSchemaManager.exe
<i>Linux</i>	/opt/Altova/StyleVisionServer2025/bin/xmlschemamanager
<i>macOS</i>	/usr/local/Altova/StyleVisionServer2025/bin/xmlschemamanager

Note: On Linux and macOS systems, once you have changed the directory to that containing the executable, you can call the executable with `sudo ./xmlschemamanager`. The prefix `./` indicates that the executable is in the current directory. The prefix `sudo` indicates that the command must be run with root privileges.

Command line syntax

The general syntax for using the command line is as follows:

```
<exec> -h | --help | --version | <command> [options] [arguments]
```

In the listing above, the vertical bar `|` separates a set of mutually exclusive items. The square brackets `[]` indicate optional items. Essentially, you can type the executable path followed by either `--h`, `--help`, or `--version` options, or by a command. Each command may have options and arguments. The list of commands is described in the following sections.

6.5.1 help

This command provides contextual help about commands pertaining to Schema Manager executable.

Syntax

```
<exec> help [command]
```

Where `[command]` is an optional argument which specifies any valid command name.

Note the following:

- You can invoke help for a command by typing the command followed by `-h` or `--help`, for example:
`<exec> list -h`
- If you type `-h` or `--help` directly after the executable and before a command, you will get general help (not help for the command), for example: `<exec> -h list`

Example

The following command displays help about the `list` command:

```
xmlschemamanager help list
```


6.5.2 info

This command displays detailed information for each of the schemas supplied as a `Schema` argument. This information for each submitted schema includes the title, version, description, publisher, and any referenced schemas, as well as whether the schema has been installed or not.

Syntax

```
<exec> info [options] Schema+
```

- The `schema` argument is the name of a schema or a part of a schema's name. (To display a schema's package ID and detailed information about its installation status, you should use the [list](#)⁹⁸ command.)
- Use `<exec> info -h` to display help for the command.

Example

The following command displays information about the latest `DocBook-DTD` and `NITF` schemas:

```
xmlschemamanager info doc nitf
```

6.5.3 initialize

This command initializes the Schema Manager environment. It creates a cache directory where information about all schemas is stored. Initialization is performed automatically the first time a schema-cognizant Altova application is installed. You would not need to run this command under normal circumstances, but you would typically need to run it after executing the `reset` command.

Syntax

```
<exec> initialize | init [options]
```

Options

The `initialize` command takes the following options:

<code>--silent, --s</code>	Display only error messages. The default is <code>false</code> .
<code>--verbose, --v</code>	Display detailed information during execution. The default is <code>false</code> .
<code>--help, --h</code>	Display help for the command.

Example

The following command initializes Schema Manager:

```
xmlschemamanager initialize
```

6.5.4 install

This command installs one or more schemas.

Syntax

```
<exec> install [options] Schema+
```

To install multiple schemas, add the `schema` argument multiple times.

The `schema` argument is one of the following:

- A schema identifier (having a format of `<name>-<version>`, for example: `cbcr-2.0`). To find out the schema identifiers of the schemas you want, run the [list](#)⁹⁸ command. You can also use an abbreviated identifier if it is unique, for example `docbook`. If you use an abbreviated identifier, then the latest version of that schema will be installed.
- The path to a `.altova_xmlschemas` file downloaded from the Altova website. For information about these files, see [Introduction to SchemaManager: How It Works](#)⁸⁴.

Options

The `install` command takes the following options:

<code>--silent, --s</code>	Display only error messages. The default is <code>false</code> .
<code>--verbose, --v</code>	Display detailed information during execution. The default is <code>false</code> .
<code>--help, --h</code>	Display help for the command.

Example

The following command installs the CBCR 2.0 (Country-By-Country Reporting) schema and the latest DocBook DTD:

```
xmlschemamanager install cbcr-2.0 docbook
```

6.5.5 list

This command lists schemas under the management of Schema Manager. The list displays one of the following

- All available schemas
- Schemas containing in their name the string submitted as a `schema` argument
- Only installed schemas
- Only schemas that can be upgraded

Syntax

```
<exec> list | ls [options] Schema?
```

If no **schema** argument is submitted, then all available schemas are listed. Otherwise, schemas are listed as specified by the submitted options (*see example below*). Note that you can submit the **schema** argument multiple times.

Options

The **list** command takes the following options:

<code>--installed, --i</code>	List only installed schemas. The default is false .
<code>--upgradeable, --u</code>	List only schemas where upgrades (patches) are available. The default is false .
<code>--help, --h</code>	Display help for the command.

Examples

- To list all available schemas, run: **xmlschemamanager list**
- To list installed schemas only, run: **xmlschemamanager list -i**
- To list schemas that contain either "doc" or "nitf" in their name, run: **xmlschemamanager list doc nitf**

6.5.6 reset

This command removes all installed schemas and the cache directory. You will be completely resetting your schema environment. After running this command, be sure to run the [initialize](#)⁹⁷ command to recreate the cache directory. Alternatively, run the **reset** command with the **-i** option. Since **reset -i** restores the original installation of the product, we recommend that you run the [update](#)¹⁰¹ command after performing a reset and initialization. Alternatively, run the **reset** command with both the **-i** and **-u** options.

Syntax

```
<exec> reset [options]
```

Options

The **reset** command takes the following options:

<code>--init, --i</code>	Initialize Schema Manager after reset. The default is false .
<code>--update, --u</code>	Updates the list of available schemas in the cache. The default is false .
<code>--silent, --s</code>	Display only error messages. The default is false .
<code>--verbose, --v</code>	Display detailed information during execution. The default is false .

`--help, --h` Display help for the command.

Examples

- To reset Schema Manager, run: `xmlschemamanager reset`
- To reset Schema Manager and initialize it, run: `xmlschemamanager reset -i`
- To reset Schema Manager, initialize it, and update its schema list, run: `xmlschemamanager reset -i -u`

6.5.7 uninstall

This command uninstalls one or more schemas. By default, any schemas referenced by the current one are uninstalled as well. To uninstall just the current schema and keep the referenced schemas, set the option `--k`.

Syntax

```
<exec> uninstall [options] Schema+
```

To uninstall multiple schemas, add the `schema` argument multiple times.

The `schema` argument is one of the following:

- A schema identifier (having a format of `<name>-<version>`, for example: `cbr-2.0`). To find out the schema identifiers of the schemas that are installed, run the `list -i` ⁹⁸ command. You can also use an abbreviated schema name if it is unique, for example `docbook`. If you use an abbreviated name, then all schemas that contain the abbreviation in its name will be uninstalled.
- The path to a `.altova_xmlschemas` file downloaded from the Altova website. For information about these files, see [Introduction to SchemaManager: How It Works](#) ⁸⁴.

Options

The `uninstall` command takes the following options:

<code>--keep-references, --k</code>	Set this option to keep referenced schemas. The default is <code>false</code> .
<code>--silent, --s</code>	Display only error messages. The default is <code>false</code> .
<code>--verbose, --v</code>	Display detailed information during execution. The default is <code>false</code> .
<code>--help, --h</code>	Display help for the command.

Example

The following command uninstalls the CBR 2.0 and EPUB 2.0 schemas and their dependencies:

```
xmlschemamanager uninstall cbr-2.0 epub-2.0
```

The following command uninstalls the `eba-2.10` schema but not the schemas it references:

```
xmlschemamanager uninstall --k cbr-2.0
```

6.5.8 update

This command queries the list of schemas available from the online storage and updates the local cache directory. You should not need to run this command unless you have performed a [reset](#)⁹⁹ and [initialize](#)⁹⁷.

Syntax

```
<exec> update [options]
```

Options

The `update` command takes the following options:

<code>--silent, --s</code>	Display only error messages. The default is <code>false</code> .
<code>--verbose, --v</code>	Display detailed information during execution. The default is <code>false</code> .
<code>--help, --h</code>	Display help for the command.

Example

The following command updates the local cache with the list of latest schemas:

```
xmlschemamanager update
```

6.5.9 upgrade

This command upgrades all installed schemas that can be upgraded to the latest available *patched* version. You can identify upgradeable schemas by running the [list -u](#)⁹⁸ command.

Note: The `upgrade` command removes a deprecated schema if no newer version is available.

Syntax

```
<exec> upgrade [options]
```

Options

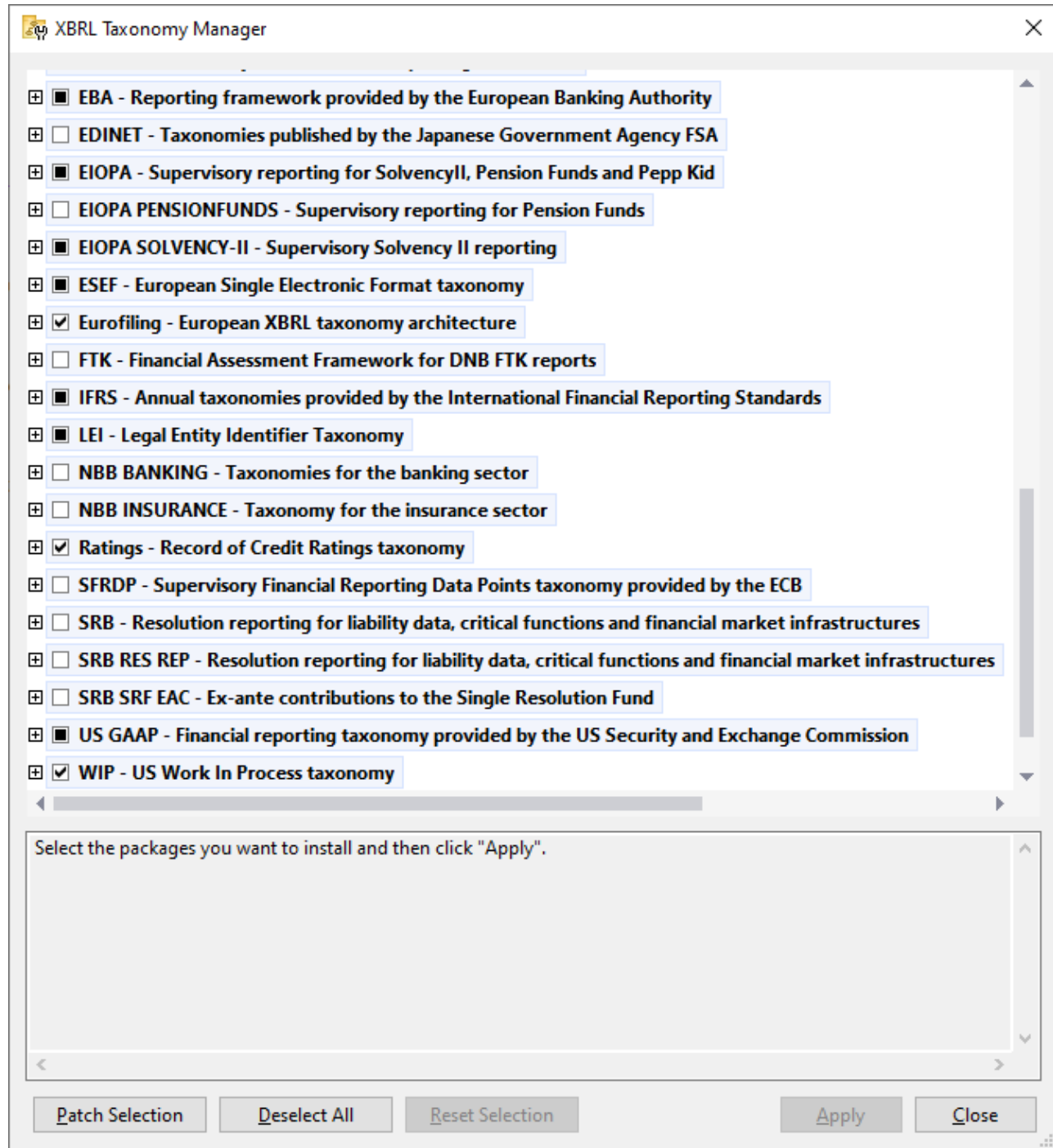
The `upgrade` command takes the following options:

<code>--silent, --s</code>	Display only error messages. The default is <code>false</code> .
<code>--verbose, --v</code>	Display detailed information during execution. The default is <code>false</code> .
<code>--help, --h</code>	Display help for the command.

7 Taxonomy Manager

XBRL Taxonomy Manager is an Altova tool that provides a centralized way to install and manage XBRL taxonomies for use across all Altova's XBRL-enabled applications, including StyleVision Server.

- On Windows, Taxonomy Manager has a graphical user interface (*screenshot below*) and is also available at the command line. (Altova's desktop applications are available on Windows only; *see list below*.)
- On Linux and macOS, Taxonomy Manager is available at the command line only. (Altova's server applications are available on Windows, Linux, and macOS; *see list below*.)



Altova's XBRL-enabled applications

Desktop applications (Windows only)	Server applications (Windows, Linux, macOS)
Altova XBRL Add-ins for Excel (EBA, ESEF, EIOPA, WIP)	MapForce Server (Standard and Advanced Editions)

MapForce Enterprise Edition	RaptorXML+XBRL Server
StyleVision Enterprise Edition	StyleVision Server
XMLSpy Enterprise Edition	

Installation and de-installation of Taxonomy Manager

Taxonomy Manager is installed automatically when you first install a new version of Altova Mission Kit Enterprise Edition or of any of Altova's XBRL-enabled applications (*see table above*).

Likewise, it is removed automatically when you uninstall the last Altova XBRL-enabled application from your computer.

Taxonomy Manager features

Taxonomy Manager provides the following features:

- Shows XBRL taxonomies installed on your computer and checks whether new versions are available for download.
- Downloads newer versions of XBRL taxonomies independently of the Altova product release cycle. (Altova stores taxonomies online, and you can download them via Taxonomy Manager.)
- Install or uninstall any of the multiple versions of a given taxonomy (or all versions if necessary).
- An XBRL taxonomy may have dependencies on other taxonomies. When you install or uninstall a particular taxonomy, Taxonomy Manager informs you about dependent taxonomies and will automatically install or remove them as well.
- Taxonomy Manager uses the [XML catalog](#) mechanism to map schema references to local files. In the case of large XBRL taxonomies, processing will therefore be faster than if the taxonomies were at a remote location.
- All major taxonomies are available via Taxonomy Manager and are regularly updated for the latest versions. This provides you with a convenient single resource for managing all your taxonomies and making them readily available to all of Altova's XBRL-enabled applications.
- Changes made in Taxonomy Manager take effect for all Altova products installed on that machine.

Custom XBRL Taxonomies

If you need to work with custom XBRL taxonomies that are not included with Taxonomy Manager, you can add these taxonomies to the set of custom packages that StyleVision Server can reference. Do this as follows:

- *In Altova desktop applications:* Select the **Tools | Options** menu command, and go to the *XBRL | Taxonomy Packages* section. Browse for the ZIP package of your custom XBRL taxonomy. For more information, see the description of this command in your desktop product documentation.
- *In Altova server applications:* When running commands from the command line that support custom taxonomies, provide the `--taxonomy-package` or `--taxonomy-package-config-file` option. For example: In RaptorXML+XBRL Server, these options are supported by XBRL validation commands such as `valxbml` or `valxbritaxonomy`; in MapForce, they are supported by `run` command.

How it works

Altova stores all XBRL taxonomies used in Altova products online. This repository is updated when new versions of the taxonomies are released. Taxonomy Manager displays information about the latest available

taxonomies when invoked in both its GUI form as well as on the CLI. You can then install, upgrade or uninstall taxonomies via Taxonomy Manager.

Taxonomy Manager also installs taxonomies in one other way. At the Altova website (<https://www.altova.com/taxonomy-manager>) you can select a taxonomy and its dependent taxonomies that you want to install. The website will prepare a file of type `.altova_taxonomies` for download that contains information about your taxonomy selection. When you double-click this file or pass it to Taxonomy Manager via the CLI as an argument of the `install`¹¹⁶ command, Taxonomy Manager will install the taxonomies you selected.

Local cache: tracking your taxonomies

All information about installed taxonomies is tracked in a centralized cache directory on your computer, located here:

<i>Windows</i>	C:\ProgramData\Altova\pkgs\.cache
<i>Linux</i>	/var/opt/Altova/pkgs\.cache
<i>macOS</i>	/var/Altova/pkgs

This cache directory is updated regularly with the latest status of taxonomies at Altova's online storage. These updates are carried out at the following times:

- Every time you start Taxonomy Manager.
- When you start StyleVision Server for the first time on a given calendar day.
- If StyleVision Server is open for more than 24 hours, the cache is updated every 24 hours.
- You can also update the cache by running the `update`¹¹⁹ command at the command line interface.

The cache therefore enables Taxonomy Manager to continuously track your installed taxonomies against the taxonomies available online at the Altova website.

Do not modify the cache manually!

The local cache directory is maintained automatically based on the taxonomies you install and uninstall. It should not be altered or deleted manually. If you ever need to reset Taxonomy Manager to its original "pristine" state, then, on the command line interface (CLI): (i) run the `reset`¹¹⁷ command, and (ii) run the `initialize`¹¹⁵ command. (Alternatively, run the `reset` command with the `--i` option.)

HTTP proxy

You can use an HTTP proxy for Taxonomy Manager connections. The system's proxy settings will be used.

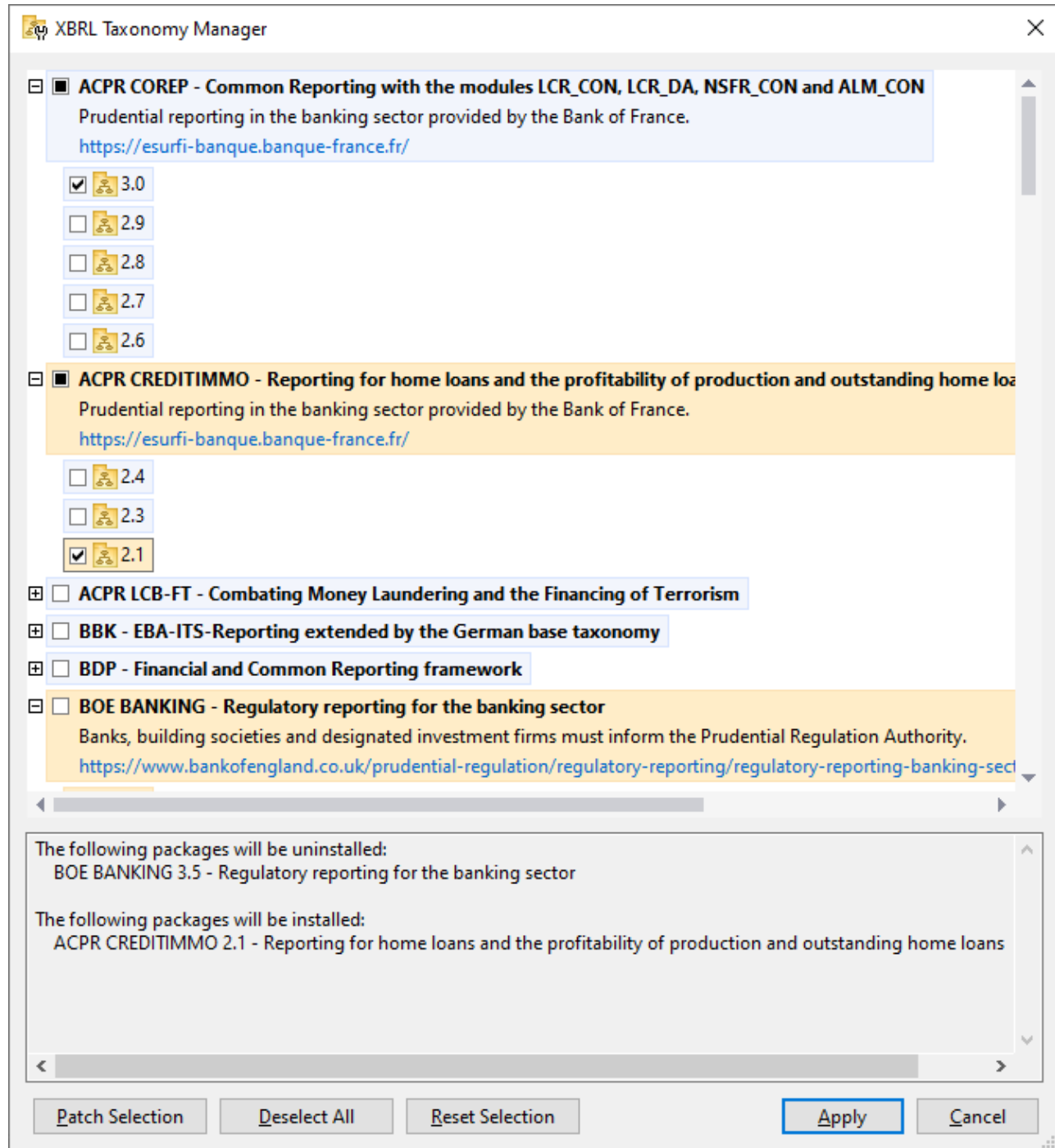
7.1 Run Taxonomy Manager

Graphical User Interface

You can access the GUI of Taxonomy Manager in any of the following ways:

- *During the installation of StyleVision Server:* Towards the end of the installation procedure, select the check box *Invoke Altova Taxonomy Manager* to access the XBRL Taxonomy Manager GUI straight away. This will enable you to install taxonomies during the installation process of your Altova application.
- Via the `.altova_taxonomies` file downloaded from the [Altova's XBRL Taxonomy Download Center](#): Double-click the downloaded file to run the Taxonomy Manager GUI, which will be set up to install the taxonomies you selected (at the website) for installation.

After the Taxonomy Manager GUI (*screenshot below*) has been opened, already installed taxonomies will be shown selected. If you want to install an additional taxonomy, select it. If you want to uninstall an already installed taxonomy, deselect it. After you have made your selections and/or deselections, you are ready to apply your changes. The taxonomies that will be installed or uninstalled will be highlighted and a message about the upcoming changes will be posted to the Messages pane at the bottom of the Taxonomy Manager window (*see screenshot*).



When you click **Apply**, the progress of the installation is displayed in a simplified Taxonomy Manager dialog. If there is an error (for example, a connection error), then an error message will be displayed in the dialog. In this case, fix the error and then click the **Advanced** button, check the taxonomy selection, and retry with **Apply**.

Command line interface

You can run Taxonomy Manager from a command line interface by sending commands to its executable file, `taxonomymanager.exe`.

The `taxonomymanager.exe` file is located in the following folder:

- *On Windows:* `C:\ProgramData\Altova\SharedBetweenVersions`
- *On Linux or macOS (server applications only):* `%INSTALLDIR%/bin`, where `%INSTALLDIR%` is the program's installation directory.

You can then use any of the commands listed in the CLI command reference section.

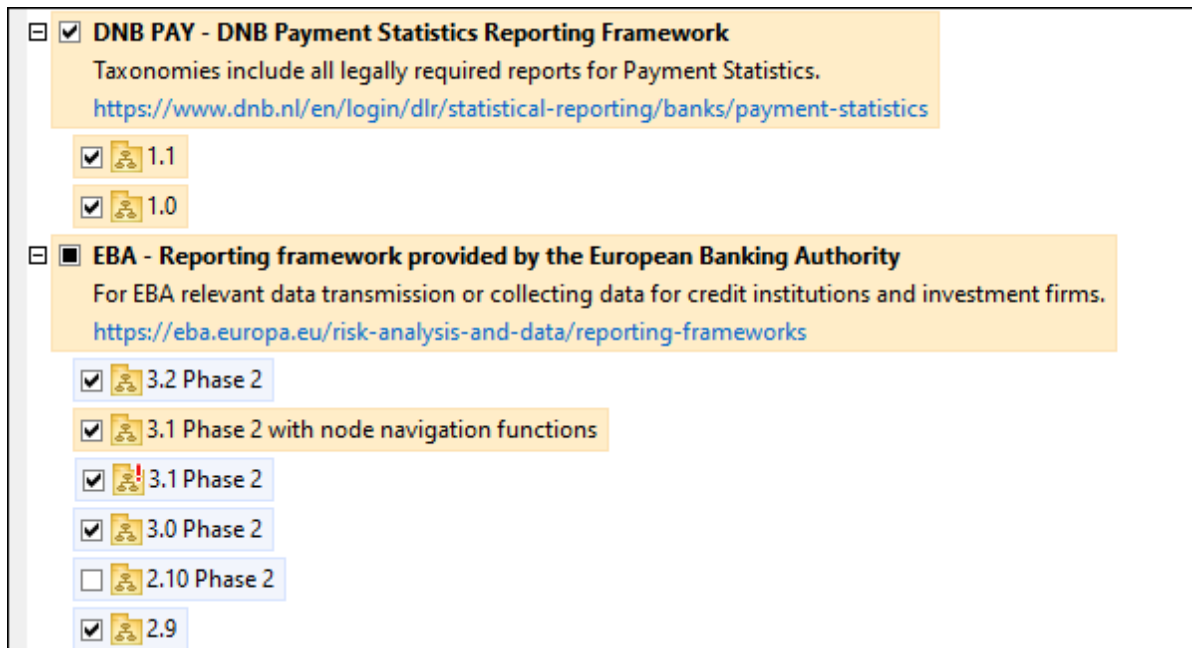
To display help for the commands, run the following:


- *On Windows:* `taxonomymanager.exe --help`
- *On Linux or macOS (server applications only):* `sudo ./taxonomymanager --help`

7.2 Status Categories

Taxonomy Manager categorizes the taxonomies under its management as follows:

- *Installed taxonomies.* These are shown in the GUI with their check boxes selected (*in the screenshot below the checked versions of the DNB and EBA taxonomies are installed taxonomies*). If all the versions of a taxonomy are selected, then the selection mark is a tick. If at least one version is unselected, then the selection mark is a solid colored square, as with EBA in the screenshot below. You can deselect an installed taxonomy to **uninstall** it.
- *Uninstalled available taxonomies.* These are shown in the GUI with their check boxes unselected. You can select the taxonomies you want to **install**.



- *Upgradeable taxonomies* are those which have been revised by their issuers since they were installed. They are indicated in the GUI by a  icon (see screenshot above). You can **patch** an installed taxonomy with an available revision.

Points to note

- In the screenshot above, both DNB taxonomies and some of the EBA taxonomies are checked. Those with the blue background are already installed. Those with the yellow background are uninstalled and have been selected for installation. Note that (i) the EBA 2.10 Phase 2 taxonomy is not installed and has not been selected for installation, (ii) the EBA 3.1 Phase 2 taxonomy has been installed, but it has been patched by its issuer since it was installed and the patch has not yet been installed.
- When running Taxonomy Manager from the command line, the `list` ¹¹⁶ command is used with different options to list different categories of taxonomies:

<code>taxonomymanager.exe list</code>	Lists all installed and available taxonomies; upgradeables are also indicated
---------------------------------------	-------------------------------------------------------------------------------

<code>taxonomymanager.exe list -i</code>	Lists installed taxonomies only; upgradeables are also indicated
<code>taxonomymanager.exe list -u</code>	Lists upgradeable taxonomies

Note: On Linux and macOS, use `sudo ./taxonomymanager list`

7.3 Patch or Install a Taxonomy

Patch an installed taxonomy

Occasionally, XBRL taxonomies may receive patches (upgrades or revisions) from their issuers. When Taxonomy Manager detects that patches are available, these are indicated in the taxonomy listings of Taxonomy Manager and you can install the patches quickly.

In the GUI

Patches are indicated by the 📄🚨 icon. (Also see the previous topic about [status categories](#) ¹⁰⁹.) If patches are available, the **Patch Selection** button will be enabled. Click it to select and prepare all patches for installation. In the GUI, the icon of each taxonomy that will be patched changes from 📄🚨 to 📄📈, and the Messages pane at the bottom of the dialog lists the patches that will be applied. When you are ready to install the selected patches, click **Apply**. All patches will be applied together. Note that if you deselect a taxonomy marked for patching, you will actually be uninstalling that taxonomy.

On the CLI

To apply a patch at the command line interface:

1. Run the `list -u` ¹¹⁶ command. This lists any taxonomies where patch upgrades are available.
2. Run the `upgrade` ¹¹⁹ command to install all the patches.

Install an available taxonomy

You can install taxonomies using either the Taxonomy Manager GUI or by sending Taxonomy Manager the install instructions via the command line.

Note: If the current taxonomy references other taxonomies, the referenced taxonomies are also installed.

In the GUI

To install taxonomies using the Taxonomy Manager GUI, select the taxonomies you want to install and click **Apply**.

You can also select the taxonomies you want to install at the [Altova website](#) and generate a downloadable `.altova_taxonomies` file. When you double-click this file, it will open Taxonomy Manager with the taxonomies you wanted pre-selected. All you will now have to do is click **Apply**.

On the CLI

To install taxonomies via the command line, run the `install` ¹¹⁶ command:

```
taxonomymanager.exe install [options] Taxonomy+
```

where `Taxonomy` is the taxonomy (or taxonomies) you want to install or a `.altova_taxonomies` file. A taxonomy is referenced by an identifier of format `<name>-<version>`. (The identifiers of taxonomies are displayed when you run the `list` ¹¹⁶ command.) You can enter as many taxonomies as you like. For details, see the description of the `install` ¹¹⁶ command.

Note: On Linux or macOS, use the `sudo ./taxonomymanager` command.

Installing a required taxonomy

When you run an XBRL-related command in StyleVision Server and StyleVision Server discovers that a taxonomy it needs for executing the command is not present or is incomplete, Taxonomy Manager will display information about the missing taxonomy. You can then directly install any missing taxonomy via Taxonomy Manager.

In the Taxonomy Manager GUI, you can view all previously installed taxonomies at any time by running Taxonomy Manager from **Tools | Taxonomy Manager**.

7.4 Uninstall a Taxonomy, Reset

Uninstall a taxonomy

You can uninstall taxonomies using either the Taxonomy Manager GUI or by sending Taxonomy Manager the uninstall instructions via the command line.

Note: If the taxonomy you want to uninstall references other taxonomies, then the referenced taxonomies are also uninstalled.

In the GUI

To uninstall taxonomies in the Taxonomy Manager GUI, clear their check boxes and click **Apply**. The selected taxonomies and their referenced taxonomies will be uninstalled.

To uninstall all taxonomies, click **Deselect All** and click **Apply**.

On the CLI

To uninstall taxonomies via the command line, run the `uninstall` command:

```
taxonomymanager.exe uninstall [options] Taxonomy+
```

where each `Taxonomy` argument is a taxonomy you want to uninstall or a `.altova_taxonomies` file. A taxonomy is specified by an identifier that has a format of `<name>-<version>`. (The identifiers of taxonomies are displayed when you run the `list`¹¹⁶ command.) You can enter as many taxonomies as you like. For details, see the description of the `uninstall`¹¹⁸ command.

Note: On Linux or macOS, use the `sudo ./taxonomymanager` command.

Reset Taxonomy Manager

You can reset Taxonomy Manager.

- In the GUI, click **Reset Selection**. This resets the the GUI to show what taxonomies are currently installed. Any selections or de-selections that the user has made in the current session will be canceled.
- On the CLI, run the `reset`¹¹⁷ command. This removes all installed taxonomies and the cache directory.

After running this command, make sure to run the `initialize`¹¹⁵ command in order to recreate the cache directory. Alternatively, run the `reset`¹¹⁷ command with the `-i` option.

Note that `reset -i`¹¹⁷ restores the original installation of the product, so it is recommended that you run the `update`¹¹⁹ command after performing a reset. Alternatively, run the `reset`¹¹⁷ command with the `-i` and `-u` options.

7.5 Command Line Interface (CLI)

To call Taxonomy Manager at the command line, you need to know the path of the executable. By default, the Taxonomy Manager executable is installed here:

<i>Windows</i>	C:\ProgramData\Altova\SharedBetweenVersions\TaxonomyManager.exe
<i>Linux</i>	/opt/Altova/StyleVisionServer2025/bin/taxonomymanager
<i>macOS</i>	/usr/local/Altova/StyleVisionServer2025/bin/taxonomymanager

Note: On Linux and macOS systems, once you have changed the directory to that containing the executable, you can call the executable with `sudo ./taxonomymanager`. The prefix `./` indicates that the executable is in the current directory. The prefix `sudo` indicates that the command must be run with root privileges.

Command line syntax

The general syntax for using the command line is as follows:

```
<exec> -h | --help | --version | <command> [options] [arguments]
```

In the listing above, the vertical bar `|` separates a set of mutually exclusive items. The square brackets `[]` indicate optional items. Essentially, you can type the executable path followed by either `--h`, `--help`, or `--version` options, or by a command. Each command may have options and arguments. The list of commands is described in the following sections.

7.5.1 help

This command provides contextual help about commands pertaining to Taxonomy Manager executable.

Syntax

```
<exec> help [command]
```

Where `[command]` is an optional argument which specifies any valid command name.

Note the following:

- You can invoke help for a command by typing the command followed by `-h` or `--help`, for example:
`<exec> list -h`
- If you type `-h` or `--help` directly after the executable and before a command, you will get general help (not help for the command), for example: `<exec> -h list`

Example

The following command displays help about the `list` command:

```
taxonomymanager help list
```

7.5.2 info

This command displays detailed information for each of the taxonomies supplied as a `Taxonomy` argument. This information for each submitted taxonomy includes the title, version, description, publisher, and any dependent taxonomies, as well as whether the taxonomy has been installed or not.

Syntax

```
<exec> info [options] Taxonomy+
```

- The `Taxonomy` argument is the name of a taxonomy or a part of a taxonomy's name. (To display a taxonomy's package ID and detailed information about its installation status, you should use the [list](#) ¹¹⁶ command.)
- Use `<exec> info -h` to display help for the command.

Example

The following command displays information about the `eba-2.10` and `us-gaap-2020.0` taxonomies:

```
taxonomymanager info eba-2.10 us-gaap-2020.0
```

7.5.3 initialize

This command initializes the Taxonomy Manager environment. It creates a cache directory where information about all taxonomies is stored. Initialization is performed automatically the first time an XBRL-enabled Altova application is installed. You would not need to run this command under normal circumstances, but you would typically need to run it after executing the `reset` command.

Syntax

```
<exec> initialize | init [options]
```

Options

The `initialize` command takes the following options:

<code>--silent, --s</code>	Display only error messages. The default is <code>false</code> .
<code>--verbose, --v</code>	Display detailed information during execution. The default is <code>false</code> .
<code>--help, --h</code>	Display help for the command.

Example

The following command initializes Taxonomy Manager:

```
taxonomymanager initialize
```

7.5.4 install

This command installs one or more taxonomies.

Syntax

```
<exec> install [options] Taxonomy+
```

To install multiple taxonomies, add the **taxonomy** argument multiple times.

The **taxonomy** argument is one of the following:

- A taxonomy identifier (having a format of **<name>-<version>**, for example: **eba-2.10**). To find out the taxonomy identifiers of the taxonomies you want, run the [list](#)¹¹⁶ command. You can also use an abbreviated identifier if it is unique, for example **eba**. If you use an abbreviated identifier, then the latest version of that taxonomy will be installed.
- The path to a **.altova_taxonomies** file downloaded from the Altova website. For information about these files, see [Introduction to TaxonomyManager: How It Works](#)¹⁰².

Options

The **install** command takes the following options:

<code>--silent, --s</code>	Display only error messages. The default is false .
<code>--verbose, --v</code>	Display detailed information during execution. The default is false .
<code>--help, --h</code>	Display help for the command.

Example

The following command installs the latest **eba** (European Banking Authority) and **us-gaap** (US Generally Accepted Accounting Principles) taxonomies:

```
taxonomymanager install eba us-gaap
```

7.5.5 list

This command lists taxonomies under the management of Taxonomy Manager. The list displays one of the following

- All available taxonomies
- Taxonomies containing in their name the string submitted as a **taxonomy** argument
- Only installed taxonomies
- Only taxonomies that can be upgraded

Syntax

```
<exec> list | ls [options] Taxonomy?
```

If no **taxonomy** argument is submitted, then all available taxonomies are listed. Otherwise, taxonomies are listed as specified by the submitted options (see *example below*). Note that you can submit the **taxonomy** argument multiple times.

Options

The **list** command takes the following options:

<code>--installed, --i</code>	List only installed taxonomies. The default is false .
<code>--upgradeable, --u</code>	List only taxonomies where upgrades (patches) are available. The default is false .
<code>--help, --h</code>	Display help for the command.

Examples

- To list all available taxonomies, run: `taxonomymanager list`
- To list installed taxonomies only, run: `taxonomymanager list -i`
- To list taxonomies that contain either "eba" or "us-gaap" in their name, run: `taxonomymanager list eba us-gaap`

7.5.6 reset

This command removes all installed taxonomies and the cache directory. You will be completely resetting your taxonomy environment. After running this command, be sure to run the [initialize](#)¹¹⁵ command to recreate the cache directory. Alternatively, run the `reset` command with the `-i` option. Since `reset -i` restores the original installation of the product, we recommend that you run the [update](#)¹¹⁹ command after performing a reset and initialization. Alternatively, run the `reset` command with both the `-i` and `-u` options.

Syntax

```
<exec> reset [options]
```

Options

The **reset** command takes the following options:

<code>--init, --i</code>	Initialize Taxonomy Manager after reset. The default is false .
<code>--update, --u</code>	Updates the list of available taxonomies in the cache. The default is false .
<code>--silent, --s</code>	Display only error messages. The default is false .
<code>--verbose, --v</code>	Display detailed information during execution. The default is false .

`--help, --h` Display help for the command.

Examples

- To reset Taxonomy Manager, run: `taxonomymanager reset`
- To reset Taxonomy Manager and initialize it, run: `taxonomymanager reset -i`
- To reset Taxonomy Manager, initialize it, and update its taxonomy list, run: `taxonomymanager reset -i -u`

7.5.7 uninstall

This command uninstalls one or more taxonomies. By default, any taxonomies referenced by the current one are uninstalled as well. To uninstall just the current taxonomy and keep the referenced taxonomies, set the option `--k`.

Syntax

```
<exec> uninstall [options] Taxonomy+
```

To uninstall multiple taxonomies, add the `Taxonomy` argument multiple times.

The `Taxonomy` argument is one of the following:

- A taxonomy identifier (having a format of `<name>-<version>`, for example: `eba-2.10`). To find out the taxonomy identifiers of the taxonomies that are installed, run the `list -i` ¹¹⁶ command. You can also use an abbreviated taxonomy name if it is unique, for example `eba`. If you use an abbreviated name, then all taxonomies that contain the abbreviation in its name will be uninstalled.
- The path to a `.altova_taxonomies` file downloaded from the Altova website. For information about these files, see [Introduction to TaxonomyManager: How It Works](#) ¹⁰².

Options

The `uninstall` command takes the following options:

<code>--keep-references, --k</code>	Set this option to keep referenced taxonomies. The default is <code>false</code> .
<code>--silent, --s</code>	Display only error messages. The default is <code>false</code> .
<code>--verbose, --v</code>	Display detailed information during execution. The default is <code>false</code> .
<code>--help, --h</code>	Display help for the command.

Example

The following command uninstalls the `eba-2.10` and `us-gaap-2020.0` taxonomies and their dependencies:

```
taxonomymanager uninstall eba-2.10 us-gaap-2020.0
```

The following command uninstalls the `eba-2.10` taxonomy but not the taxonomies it references:

```
taxonomymanager uninstall --k eba-2.10
```

7.5.8 update

This command queries the list of taxonomies available from the online storage and updates the local cache directory. You should not need to run this command unless you have performed a [reset](#)¹¹⁷ and [initialize](#)¹¹⁵.

Syntax

```
<exec> update [options]
```

Options

The `update` command takes the following options:

<code>--silent, --s</code>	Display only error messages. The default is <code>false</code> .
<code>--verbose, --v</code>	Display detailed information during execution. The default is <code>false</code> .
<code>--help, --h</code>	Display help for the command.

Example

The following command updates the local cache with the list of latest taxonomies:

```
taxonomymanager update
```

7.5.9 upgrade

This command upgrades all installed taxonomies that can be upgraded to the latest available *patched* version. You can identify upgradeable taxonomies by running the [list -u](#)¹¹⁶ command.

Note: The `upgrade` command removes a deprecated taxonomy if no newer version is available.

Syntax

```
<exec> upgrade [options]
```

Options

The `upgrade` command takes the following options:

<code>--silent, --s</code>	Display only error messages. The default is <code>false</code> .
<code>--verbose, --v</code>	Display detailed information during execution. The default is <code>false</code> .
<code>--help, --h</code>	Display help for the command.

Index

A

- Altova ServiceController, 16
- Assigning a license to StyleVision Server on Linux, 24
- Assigning a license to StyleVision Server on macOS, 30
- Assigning a license to StyleVision Server on Windows, 17

C

- Chrome,
 - limitations of Altova Help in, 7
 - Same-Origin Policy (SOP), 7

D

- Database connections on Linux, 24
- Database connections on macOS, 31
- Deinstallation, 12

E

- Environment settings on Linux, 24
- Environment settings on macOS, 31

F

- FlowForce Server,
 - and StyleVision Server, 9
- FOP requirements, 35

G

- Google Chrome,

see Chrome, 7

I

- Installation of StyleVision Server, 11
- Installation on Linux, 20
- Installation on macOS, 27
- Installing LicenseServer on Linux, 22
- Installing LicenseServer on macOS, 28
- Installing LicenseServer on Windows, 15
- Installing on Windows, 12
- Installing on Windows Server Core, 13

J

- JRE, 35

L

- License for StyleVision Server,
 - assigning on Linux, 24
 - assigning on macOS, 30
 - assigning on Windows, 17
- LicenseServer versions, 15, 22, 28
- Licensing of StyleVision Server, 11
- Linux,
 - installation on, 20

M

- macOS,
 - installation on, 27
- Migrating StyleVision Server to a new machine, 34

O

- OpenJDK, 35

R

Register StyleVision Server with LicenseServer on Linux, 23

Register StyleVision Server with LicenseServer on macOS, 29

Register StyleVision Server with LicenseServer on Windows, 17

S

Schema Manager,

- CLI Help command, 96
- CLI Info command, 97
- CLI Initialize command, 97
- CLI Install command, 98
- CLI List command, 98
- CLI overview, 96
- CLI Reset command, 99
- CLI Uninstall command, 100
- CLI Update command, 101
- CLI Upgrade command, 101
- how to run, 88
- installing a schema, 93
- listing schemas by status in, 91
- overview of, 84
- patching a schema, 93
- resetting, 95
- status of schemas in, 91
- uninstalling a schema, 95
- upgrading a schema, 93

screenshot viewing limitations in Altova Help,

- see note about Chrome's SOP, 7

Security considerations, 36

Setup,

- on Linux, 20
- on macOS, 27
- on Windows, 12

Setup of StyleVision Server, 11

Start LicenseServer on Linux, 23

Start LicenseServer on macOS, 29

Start LicenseServer on Windows, 16

Start StyleVision Server on Linux, 23

Start StyleVision Server on macOS, 29

Start StyleVision Server on Windows, 16

StyleVision Server,

- features, 8
- functionality, 8
- in the FlowForce workflow, 9
- migrating to a new machine, 34

StyleVision Server DLL,

- registration of, 18

T

Taxonomy Manager,

- CLI Help command, 114
- CLI Info command, 115
- CLI Initialize command, 115
- CLI Install command, 116
- CLI List command, 116
- CLI overview, 114
- CLI Reset command, 117
- CLI Uninstall command, 118
- CLI Update command, 119
- CLI Upgrade command, 119
- how to run, 106
- installing a taxonomy, 111
- listing taxonomies by status in, 109
- overview of, 102
- patching a taxonomy, 111
- resetting, 113
- status of taxonomies in, 109
- uninstalling a taxonomy, 113
- upgrading a taxonomy, 111

TOC expand/collapse,

- see note about Chrome's SOP, 7

U

Uninstalling, 12

Upgrading StyleVision Server on Windows, 33

W

Windows,

- installation on, 12

Windows,

upgrading StyleVision Server on, 33