Table of Contents

1 Introduction ........................................... 1

2 Functionality ........................................... 3
   2.1 In the FlowForce Workflow .................. 4
   2.2 As a Standalone Server ...................... 5

3 StyleVision Server Setup .......................... 6
   3.1 Setup on Windows ............................... 7
       3.1.1 Installation on Windows............... 8
       3.1.2 Licensing on Windows............... 10
   3.2 Setup on Linux ................................. 13
       3.2.1 Installation on Linux............... 14
       3.2.2 Licensing on Linux............... 18
       3.2.3 Notes about Environment ............ 20
   3.3 Setup on macOS ............................... 22
       3.3.1 Installation on macOS .............. 23
       3.3.2 Licensing on macOS ............ 26
       3.3.3 Notes about Environment ............ 28

4 StyleVision Server Command Line .............. 29
   4.1 assignlicense (Windows only) ............. 31
   4.2 exportresourcestrings ..................... 33
   4.3 generate ....................................... 35
   4.4 help ........................................... 38
   4.5 licenseserver .................................. 40
   4.6 pdfdata ....................................... 42
   4.7 setfopath ..................................... 44
5 StyleVision Server API

5.1 About the .NET Interface ........................................................................................................49
5.2 About the COM Interface ..........................................................................................................50
5.3 About the Java Interface ............................................................................................................51
5.4 Code Examples ...........................................................................................................................52
  5.4.1 C++ ......................................................................................................................................53
  5.4.2 C# .......................................................................................................................................55
  5.4.3 VBScript ..............................................................................................................................57
  5.4.4 Visual Basic .........................................................................................................................58
  5.4.5 Java .....................................................................................................................................59
5.5 API Reference ...........................................................................................................................61
  5.5.1 COM and .NET .....................................................................................................................62
  5.5.2 Java ....................................................................................................................................69

Index .................................................................................................................................73
1 Introduction

Altova StyleVision Server is an implementation of Altova StyleVision's built-in execution engine. It operates as a module of Altova's FlowForce Server, and it is also available as a standalone server product.

StyleVision Server executes transformation packages that have been deployed to a FlowForce Server. These transformations are initiated by FlowForce Server based on a variety of programmable time triggers, file triggers, or remote triggers. Additionally, StyleVision Server functionality can be invoked via the command line.

System requirements, installation and licensing
StyleVision Server is supported on the following operating systems:

- **Windows**
  - Windows 7 SP1 with Platform Update, Windows 8, Windows 10

- **Windows Server**
  - Windows Server 2008 R2 SP1 with Platform Update or newer

- **Linux**
  - CentOS 6 or newer
  - RedHat 6 or newer
  - Debian 8 or newer
  - Ubuntu 14.04 or newer

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 command `yum` (or `apt-get` if applicable) to install them.

<table>
<thead>
<tr>
<th>Required by</th>
<th>CentOS, RedHat</th>
<th>Debian</th>
<th>Ubuntu</th>
</tr>
</thead>
<tbody>
<tr>
<td>LicenseServer</td>
<td>krb5-libs</td>
<td>libgssapi-krb5-2</td>
<td>libgssapi-krb5-2</td>
</tr>
<tr>
<td>StyleVision Server</td>
<td>qt4, krb5-libs, qt-x11</td>
<td>libqtcore4, libqtgui4, libgssapi-krb5-2</td>
<td>libqtcore4, libqtgui4, libgssapi-krb5-2</td>
</tr>
</tbody>
</table>

**Note:** 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.

**Note:** On newly installed Ubuntu server 18.04.1 LTS, it was found that the `universe` repository needed to be enabled to apt (using the command: `sudo add-apt-repository universe`). Installation of `libqtgui4` was possible after this update.
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, Linux, and macOS.

Last updated: 29 April 2019
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. For more information about Altova FlowForce, visit the Altova website.
- As a standalone server product 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, Command Line Usage.
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, Command Line Usage.

Comma Line Usage
3 StyleVision Server Setup

This section describes procedures for setting up StyleVision Server. It describes the following:

- Installation and licensing of StyleVision Server on Windows systems
- Installation and licensing of StyleVision Server on Linux systems
- Installation and licensing of StyleVision Server on macOS systems
3.1 Setup on Windows

This section describes the installation and licensing of StyleVision Server on Windows systems.

**Installation on Windows**
- System requirements
- Installing StyleVision Server
- Altova LicenseServer
- LicenseServer versions
- Trial license
- Application folder location

**Licensing on Windows**
- Start ServiceController
- Start LicenseServer
- Register StyleVision Server
- Assign a license
3.1.1 Installation on Windows

StyleVision Server is available for installation on Windows systems. Its installation and setup procedure is described below.

▲ System requirements

▲ Windows
Windows 7 SP1 with Platform Update, Windows 8, Windows 10

▲ Windows Server
Windows Server 2008 R2 SP1 with Platform Update or newer

▲ Installing StyleVision Server

StyleVision Server can be installed on Windows systems as follows:

- As a separate standalone server product called StyleVision Server. To install StyleVision Server, download and run the StyleVision Server installer. Follow the on-screen instructions.
- As part of the FlowForce Server installation package. 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 (http://www.altova.com/download.html).

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

<ProgramFilesFolder>\Altova\StyleVisionServer2019\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.

▲ Altova LicenseServer

- 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, an option is available that allows you to download and install Altova LicenseServer together with StyleVision Server or FlowForce Server.
- If an Altova LicenseServer is already installed on your network, you do not need to install another one—unless a newer version of Altova 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 Altova LicenseServer as appropriate.
See the section, Licensing on Windows, for more information about how to register and license StyleVision Server with Altova LicenseServer.

❖ LicenseServer versions

- Altova server products must be licensed either with the version of LicenseServer that is appropriate to the installed StyleVision Server version, or with a later version of LicenseServer.
- The LicenseServer version that is appropriate for a particular version of StyleVision Server is displayed during the installation of StyleVision Server. You can install this version of LicenseServer along with StyleVision Server, or you can install LicenseServer separately.
- Before installing a newer version of LicenseServer, any older one must be de-installed. The LicenseServer installer will do this automatically if it detects an older version.
- LicenseServer versions are backwards compatible. They will work with older versions of StyleVision Server.
- If you install a new version of StyleVision Server and if your installed LicenseServer version is older than the appropriate LicenseServer, install the latest version available on the Altova website.
- 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.
- The version number of the currently installed LicenseServer is given at the bottom of the LicenseServer configuration page (all tabs).

Current version: 3.2

❖ Trial license

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

❖ Application folder location

The application will be installed in the following folder:

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Folder Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows 7, 8, 10</td>
<td>C:\Program Files\Altova\</td>
</tr>
<tr>
<td>32 bit Version on 64-bit OS</td>
<td>C:\Program Files (x86)\Altova\</td>
</tr>
</tbody>
</table>
3.1.2 Licensing on Windows

StyleVision Server must be licensed with Altova LicenseServer. Licensing is a two-step process:

1. **Register StyleVision Server** with LicenseServer. Registration is done from StyleVision Server.
2. **Assign a license** to StyleVision Server from LicenseServer. Download the latest version of LicenseServer from the [Altova website](https://altova.com), and install it on your local machine or a machine on your network.

The steps to carry out are given below in brief. For detailed information, see the LicenseServer user manual at the [Altova website](https://altova.com).

**Start ServiceController**

Altova ServiceController is started in order to start Altova LicenseServer.

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* (FlowForce Server, RaptorXML(+XBRL) Server, and Mobile Together Server). It can be started by clicking **Start | Altova LicenseServer | Altova ServiceController**. (This command is also available in the Start menu folders of *Altova server products that are installed as services* (FlowForce Server, RaptorXML(+XBRL) Server, and Mobile Together Server).) After ServiceController has been started, it can be accessed via the system tray (screenshot below).

![ServiceController Screenshot](https://example.com/servicecontroller_screenshot.png)

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**.

![ServiceController Exit Menu](https://example.com/servicecontroller_exit_menu.png)
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, the Start Service option will be disabled.

![ServiceController screenshot](image)

Register StyleVision Server

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

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

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

```
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 Server Management tab of LicenseServer's configuration page to assign a license to StyleVision Server.

Assign a license

After successfully registering StyleVision Server, it will be listed in the Server 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, will 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—the machine on which the Altova server product is installed—at a time, even if the license has unused licensing capacity. 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.
3.2 Setup on Linux

This section describes the installation and licensing of StyleVision Server on Linux systems (Debian, Ubuntu, CentOS, RedHat).

### Installation on Linux
- System requirements
- Uninstall old versions of Altova server products
- Download the Linux package
- Install StyleVision Server
- Altova LicenseServer
- LicenseServer versions

### Licensing on Linux
- Start LicenseServer
- Register StyleVision Server
- Assign a license

### Notes about Environment
3.2.1 Installation on Linux

StyleVision Server is available for installation on Linux systems. Its installation and setup procedure is described below.

- System requirements
  - Linux
    - CentOS 6 or newer
    - RedHat 6 or newer
    - Debian 8 or newer
    - Ubuntu 14.04 or newer

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 command `yum` (or `apt-get` if applicable) to install them.

<table>
<thead>
<tr>
<th>Required by</th>
<th>CentOS, RedHat</th>
<th>Debian</th>
<th>Ubuntu</th>
</tr>
</thead>
<tbody>
<tr>
<td>LicenseServer</td>
<td>krb5-libs</td>
<td>libgssapi-krb5-2</td>
<td>libgssapi-krb5-2</td>
</tr>
<tr>
<td>StyleVision Server</td>
<td>qt4, krb5-libs, qt-x11</td>
<td>libqtcore4, libqtgui4, libgssapi-krb5-2</td>
<td>libqtcore4, libqtgui4, libgssapi-krb5-2</td>
</tr>
</tbody>
</table>

**Note:** 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.

**Note:** On newly installed Ubuntu server 18.04.1 LTS, it was found that the `universe` repository needed to be enabled to apt (using the command: `sudo add-apt-repository universe`). Installation of `libqtgui4` was possible after this update.

- FlowForce Server integration

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

```
cp /opt/Altova/StyleVisionServer2019/etc/*.tool /opt/Altova/FlowForceServer2019/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 old versions of Altova server products
If you need to uninstall a previous version, do this as follows. On the Linux command line interface (CLI), you can check which Altova server products are installed with the following command:

[Debian, Ubuntu]: `dpkg --list | grep Altova`

[CentOS, RedHat]: `rpm -qa | grep server`

If StyleVision Server is not installed, go ahead with the installation as documented below in Installing StyleVision Server.

If StyleVision Server is installed and you wish to install a newer version of StyleVision Server, uninstall the old version with the command:

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

[CentOS, RedHat]: `sudo rpm -e stylevisionserver`

If you need to uninstall an old version of Altova LicenseServer, do this with the following command:

[Debian, Ubuntu]: `sudo dpkg --remove licenseserver`

[CentOS, RedHat]: `sudo rpm -e licenseserver`

Download the Linux package

StyleVision Server installation packages for the following Linux systems are available at the Altova website.

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Package extension</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debian</td>
<td>.deb</td>
</tr>
<tr>
<td>Ubuntu</td>
<td>.deb</td>
</tr>
<tr>
<td>CentOS</td>
<td>.rpm</td>
</tr>
<tr>
<td>RedHat</td>
<td>.rpm</td>
</tr>
</tbody>
</table>

After downloading the Linux package, copy it to any directory on the Linux system. Since you will need an Altova LicenseServer in order to run StyleVision Server, you may want to download LicenseServer from the Altova website at the same time as you download StyleVision Server, rather than download it at a later time.

Install StyleVision Server

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

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

Install StyleVision Server with the following command:

[Debian]: `sudo dpkg --install stylevisionserver-2019-debian.deb`

[Ubuntu]: `sudo dpkg --install stylevisionserver-2019-ubuntu.deb`

[CentOS]: `sudo rpm -ivh stylevisionserver-2019-1.x86_64.rpm`

[RedHat]: `sudo rpm -ivh stylevisionserver-2019-1.x86_64.rpm`
The StyleVision Server package will be installed in the folder:
/opt/Altova/StyleVisionServer2019

Altova LicenseServer

In order for any Altova Server product—including StyleVision Server—to run, that server product must be licensed via an Altova LicenseServer on your network.

On Linux systems, Altova LicenseServer will need to be installed separately. Download LicenseServer from the Altova website and copy the package to any directory on the Linux system. Install it just like you installed StyleVision Server (see previous step).

[Debian]: sudo dpkg --install licenseserver-3.2-debian.deb
[Ubuntu]: sudo dpkg --install licenseserver-3.2-ubuntu.deb
[CentOS]: sudo rpm -ivh licenseserver-3.2-1.x86_64.rpm
[RedHat]: sudo rpm -ivh licenseserver-3.2-1.x86_64.rpm

The LicenseServer package will be installed in:
/opt/Altova/LicenseServer

For information about how to register StyleVision Server with Altova LicenseServer and license it, see the section, Licensing on Linux. Also see the LicenseServer documentation for more detailed information.

LicenseServer versions

- Altova server products must be licensed either with the version of LicenseServer that is appropriate to the installed StyleVision Server version, or with a later version of LicenseServer.
- The LicenseServer version that is appropriate for a particular version of StyleVision Server is displayed during the installation of StyleVision Server. You can install this version of LicenseServer along with StyleVision Server, or you can install LicenseServer separately.
- Before installing a newer version of LicenseServer, any older one must be de-installed. The LicenseServer installer will do this automatically if it detects an older version.
- LicenseServer versions are backwards compatible. They will work with older versions of StyleVision Server.
- If you install a new version of StyleVision Server and if your installed LicenseServer version is older than the appropriate LicenseServer, install the latest version available on the Altova website.
- 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.
- The version number of the currently installed LicenseServer is given at the bottom of the LicenseServer configuration page (all tabs).

Current version: 3.2
3.2.2 Licensing on Linux

StyleVision Server must be licensed with Altova LicenseServer. Licensing is a two-step process:

1. **Register StyleVision Server** with LicenseServer. Registration is done from StyleVision Server.
2. **Assign a license** to StyleVision Server from LicenseServer. Download the latest version of LicenseServer from the Altova website, and install it on your local machine or a machine on your network.

The steps to carry out are given below in brief. For detailed information, see the LicenseServer user manual at the Altova website.

▶ 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:

<table>
<thead>
<tr>
<th>Platform</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; Debian 8</td>
<td>sudo /etc/init.d/licenseserver start</td>
</tr>
<tr>
<td>≥ Debian 8</td>
<td>sudo systemctl start licenseserver</td>
</tr>
<tr>
<td>&lt; CentOS 7</td>
<td>sudo initctl start licenseserver</td>
</tr>
<tr>
<td>≥ CentOS 7</td>
<td>sudo systemctl start licenseserver</td>
</tr>
<tr>
<td>&lt; Ubuntu 15</td>
<td>sudo initctl start licenseserver</td>
</tr>
<tr>
<td>≥ Ubuntu 15</td>
<td>sudo systemctl start licenseserver</td>
</tr>
<tr>
<td>RedHat</td>
<td>sudo initctl start licenseserver</td>
</tr>
</tbody>
</table>

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

```
sudo /etc/init.d/licenseserver stop
```

▶ Register StyleVision Server

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

```
sudo /opt/Altova/StyleVisionServer2019/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/StyleVisionServer2019/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/StyleVisionServer2019/bin/
```

After successful registration, go to the Server Management tab of LicenseServer's configuration page to assign a license to StyleVision Server.
Assign a license

After successfully registering StyleVision Server, it will be listed in the Server 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, will 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—the machine on which the Altova server product is installed—at a time, even if the license has unused licensing capacity. 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.
3.2.3 Notes about Environment

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

- **Installation root**
  
  `/opt/Altova/StyleVisionServer2019/`

- **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/` 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 will need to be adjusted to match your environment:
Shell script (unit file)

```bash
#!/bin/bash

# - jdbc - environment -
export PATH=/usr/local/jdk1.7.0_17/bin:/usr/lib64/qt-3.3/bin:/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:/home/qa/bin
export JAVA_HOME=/usr/local/jdk1.7.0_17
```
3.3 Setup on macOS

This section describes the installation and licensing of StyleVision Server on macOS systems.

**Installation on macOS**
- System requirements
- Uninstall old versions of Altova server products
- Download the macOS package
- Install StyleVision Server
- Altova LicenseServer
- LicenseServer versions

**Licensing on macOS**
- Start LicenseServer
- Register StyleVision Server
- Assign a license

**Notes about Environment**
3.3.1 Installation on macOS

StyleVision Server is available for installation on macOS systems. Its installation and setup procedure is described below.

- **System requirements**
  - macOS
    - macOS 10.12 or newer

- **FlowForce Server integration**

  If you install 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 /usr/local/Altova/StyleVisionServer2019/etc/*.tool /usr/local/Altova/
  FlowForceServer2019/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 old versions of Altova server products**

  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 command:

  ```
  sudo rm -rf /usr/local/Altova/StyleVisionServer2019/
  ```

  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.

- **Download the disk image file**

  Download the disk image (.dmg) file from the Altova website (http://www.altova.com/download.html).

- **Install StyleVision Server**

  Click to open the downloaded disk image (.dmg). This causes the StyleVision Server
installer to appear as a new virtual drive on your computer. On the new virtual drive, double-click the installer package (.pkg). 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. 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/StyleVisionServer2019` (application binaries)
- `/var/Altova/StyleVisionServer` (data files: database and logs)

**Altova LicenseServer**

In order for any Altova Server product—including StyleVision Server—to run, that server product must be licensed via an Altova LicenseServer on your network.

The Altova LicenseServer installation package is available on the virtual drive you have mounted in the previous step. To install Altova LicenseServer, double-click the installer package included on the virtual drive and follow the on-screen instructions. You will need to accept the license agreement for installation to proceed.

Altova LicenseServer can also be downloaded and installed separately from the Altova website ([http://www.altova.com/download.html](http://www.altova.com/download.html)).

The LicenseServer package will be installed in the folder:

- `/usr/local/Altova/LicenseServer`

For information about how to register StyleVision Server with Altova LicenseServer and license it, see the section, **Licensing on macOS**.

**LicenseServer versions**

- Altova server products must be licensed either with the version of LicenseServer that is appropriate to the installed StyleVision Server version, or with a later version of LicenseServer.
- The LicenseServer version that is appropriate for a particular version of StyleVision Server is displayed during the installation of StyleVision Server. You can install this version of LicenseServer along with StyleVision Server, or you can install LicenseServer separately.
- Before installing a newer version of LicenseServer, any older one must be de-installed. The LicenseServer installer will do this automatically if it detects an older version.
- LicenseServer versions are backwards compatible. They will work with older versions of StyleVision Server.
- If you install a new version of StyleVision Server and if your installed LicenseServer version is older than the appropriate LicenseServer, install the latest version available on the Altova website.
- 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.
- The version number of the currently installed LicenseServer is given at the bottom of the LicenseServer configuration page (all tabs).

  **Current version: 3.2**
3.3.2 Licensing on macOS

StyleVision Server must be licensed with Altova License Server. Licensing is a two-step process:

1. **Register StyleVision Server** with License Server. Registration is done from StyleVision Server.
2. **Assign a license** to StyleVision Server from License Server. Download the latest version of License Server from the Altova website, and install it on your local machine or a machine on your network.

The steps to carry out are given below in brief. For detailed information, see the License Server user manual at the Altova website.

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

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

If at any time you need to stop License Server, replace load with unload in the above command:

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

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

```
sudo /usr/local/Altova/StyleVisionServer2019/bin/StyleVisionServer licenseserver [options] ServerName-Or-IP-Address
```

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

```
sudo /usr/local/Altova/StyleVisionServer2019/bin/StyleVisionServer licenseserver localhost
```

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

```
/usr/local/Altova/StyleVisionServer2019/bin/
```

After successful registration, go to the Server Management tab of License Server’s configuration page to assign a license to StyleVision Server.

**Assign a license**
After successfully registering StyleVision Server, it will be listed in the Server Management tab of the configuration page of License Server. 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, will 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—the machine on which the Altova server product is installed—at a time, even if the license has unused licensing capacity. 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.
3.3.3 Notes about Environment

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

- **Installation root**
  `/usr/local/Altova/StyleVisionServer2019/`

- **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.
4 StyleVision Server Command Line

This topic:
- Default location of StyleVision Server executable
- Usage and list of CLI commands

Default location of StyleVision Server executable

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

- **Windows**
  `<ProgramFilesFolder>\Altova\StyleVisionServer2019\bin\StyleVisionServer.exe`

- **Linux**
  `/opt/Altova/StyleVisionServer2019/bin/stylevisionserver`

- **Mac**
  `/usr/local/Altova/StyleVisionServer2019/bin/stylevisionserver`

Usage and list of CLI commands

General command line syntax for `StyleVisionServer` is:

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

**Casing and slashes on the command line**

<table>
<thead>
<tr>
<th>Casing and slashes on Windows</th>
<th>Casing and slashes on Unix (Linux, Mac)</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Note that lowercase (<code>stylevisionserver</code>) works on all platforms (Windows, Linux, and Mac), while upper-lower (<code>StyleVisionServer</code>) works only on Windows and Mac.</td>
<td></td>
</tr>
<tr>
<td>* Use forward slashes on Linux and Mac, backslashes on Windows.</td>
<td></td>
</tr>
</tbody>
</table>

where

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>`--h</td>
<td>--help`</td>
</tr>
<tr>
<td><code>--version</code></td>
<td>Displays the version of StyleVision Server.</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>licenseserver</code></td>
<td>Register StyleVision Server with LicenseServer on the local network.</td>
</tr>
<tr>
<td><code>assignlicense</code></td>
<td>Upload a license to LicenseServer and assign it to StyleVision Server on this machine.</td>
</tr>
<tr>
<td><code>verifylicense</code></td>
<td>Check if the current StyleVision Server is licensed;</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>`generate</td>
<td>gen`</td>
</tr>
<tr>
<td>`setfopath</td>
<td>sfp`</td>
</tr>
<tr>
<td><code>pdfdata</code></td>
<td>Generates form data from a PDF file to an FDF or XML file.</td>
</tr>
<tr>
<td><code>exportresourcestrings</code></td>
<td>Exports all application resource strings to an XML file.</td>
</tr>
<tr>
<td>`setdeflang</td>
<td>sdl`</td>
</tr>
<tr>
<td><code>help</code></td>
<td>Displays help for a specific command. For example: <code>help generate</code></td>
</tr>
</tbody>
</table>
### 4.1 assignlicense (Windows only)

On execution, the `assignlicense` command uploads the license file specified by the `FILE` argument to the registered LicenseServer, and assigns the license to StyleVision Server on this machine. The `FILE` argument takes the filepath of the license file. The `--test-only` option allows you to upload to LicenseServer and validate the license, without assigning the license to StyleVision Server. For details about licensing, see the LicenseServer documentation (https://www.altova.com/documentation).

**Note:** This command is supported only on Windows systems. It is not supported on Linux or Mac systems.

#### Syntax

```
StyleVisionServer assignlicense [options] FILE
```

#### Casing and slashes on the command line
- **StyleVisionServer** on Windows
- **stylevisionserver** on 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.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>File</td>
<td>The path to the license file</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-t, --test-only</code></td>
<td>Values are <code>true</code></td>
</tr>
<tr>
<td><code>--h, --help</code></td>
<td>Displays information about the command</td>
</tr>
</tbody>
</table>

Options are listed in their short forms (if available) and long forms. You can use one or two dashes for both short and long forms. An option that takes a value is written like this: `--option=value`.

**Note:** On Windows systems: Avoid using the end backslash and closing quote on the command line `\"`, for example, as in: "C:\My directory\". These two characters are interpreted by the command line parser as a literal double-quotation mark. Use the double backslash `\` if spaces occur in the command line and you need the quotes (for example: "C:\My Directory\"). Alternatively, try to avoid using spaces and, therefore, quotes at all.
Examples

Examples of the `assignlicense` command:

```
stylevisionserver assignlicense C:\StyleVision Server12345.altova_licenses
stylevisionserver assignlicense --test-only=true C:\StyleVision Server12345.altova_licenses
```

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

The `exportresourcestrings` command outputs an XML file containing the resource strings of the StyleVision Server application. It takes two arguments:

- 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 path and name of the output XML file

How to create localizations is described below.

Syntax

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

Casing and slashes on the command line

* 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.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LanguageCode</td>
<td>Specifies the language of resource strings in the exported XML file.</td>
</tr>
<tr>
<td></td>
<td>Supported languages are: en, de, es, fr, ja</td>
</tr>
<tr>
<td>XMLOutputFile</td>
<td>Specifies the location and name of the exported XML file.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--h, --help</td>
<td>Displays information about the command</td>
</tr>
</tbody>
</table>

Options are listed in their short forms (if available) and long forms. You can use one or two dashes for both short and long forms. An option that takes a value is written like this: `--option=value`.

Note: On Windows systems: Avoid using the end backslash and closing quote on the command line `\"`, for example, as in: "C:\My directory\". These two characters are interpreted by the command line parser as a literal double-quotation mark. Use the double backslash `\` if spaces occur in the command line and you need the quotes (for example: "C:\My Directory\""). Alternatively, try to avoid using spaces and, therefore, quotes at all.
Examples

An example of the `exportresoucestrings` command:

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

This command creates a file called `Strings.xml` at `c:\` that contains all 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\StyleVisionServer2019\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 `exportresoucestrings` 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 four 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<%APPFOLDER\bin` folder. Your DLL file will have a name of the form `StyleVisionServer2019_lc.dll`. The `_lc` part of the name contains the language code. For example, in `StyleVisionServer2019_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.
4.3 generate

The `generate` command (short form is `gen`) takes:

- an input XML file as a **mandatory option**
- an input PXF file as the command's argument; PXF files are created with Altova's StyleVision application.
- at least one **output-creation option**

The command 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.

**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\StyleVisionServer2019\SharedBetweenVersions`; on Linux and macOS systems, in a descendant folder of the `StyleVisionServer2019` folder. Note that Apache FOP requires **Java Runtime Environment 1.6 or later** to 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.

**Syntax**

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

**InputPXPF**

**Casing and slashes on the command line**

- `StyleVisionServer` on Windows
- `stylevisionserver` on 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.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputPXF</td>
<td>Path to the PXF file which contains the XSLT document/s that will be used to generate the output document/s.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-- xml, --inputxml</td>
<td>The XML file to process. This option is mandatory. The XML file can be located inside or outside the PXF file. To target XML files inside a PXF file, see the highlighted parts in the examples below. The <code>--inputxml</code> option is ignored if the main schema source of the input PXF is a DB or DB-XML.</td>
</tr>
<tr>
<td><strong>Mandatory option</strong></td>
<td></td>
</tr>
<tr>
<td>--dbw, --dbwhere</td>
<td>A WHERE clause that determines what rows of a DB-XML source to</td>
</tr>
</tbody>
</table>
process. Form: --dbwhere=WHEREClause

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

--prohibit-output-outside-target-folder
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.

--html, --outhtml
The output HTML file to create. Form: --outhtml=Filename

--pdf, --outpdf
The output PDF file to create. Form: --outpdf=Filename

--rtf, --outrtf
The output RTF file to create. Form: --outrtf=Filename

--docx, --outdocx
The output DOCX file to create. Form: --outdocx=Filename

--fo, --outfo
The FO file to create. Form: --outfo=Filename

--generate-html-output-as-mime
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.

--v, --verbose
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.

--l, --lang
The language used for displaying messages. Form: --lang=languagecode. Languages supported on installation: en, de, fr, es, ja

--h, --help
Displays information about the command

**Note:** If the output-creation options (--html, --pdf, --rtf, --docx, --fo) are given as a relative path, the output file's location will be relative to the folder in which the PXF file is.

Options are listed in their short forms (if available) and long forms. You can use one or two dashes for both short and long forms. An option that takes a value is written like this: --option=value.

**Note:** On Windows systems: Avoid using the end backslash and closing quote on the command line ", for example, as in: "C:\My directory\". These two characters are interpreted by the command line parser as a literal double-quotation mark. Use the double backslash \ if spaces occur in the command line and you need the quotes (for example: "C:\My Directory\"). Alternatively, try to avoid using spaces and, therefore, quotes at all.

**Examples**

Example of the generate command:
The command above contains the mandatory **--inputxml** option, the InputPXF argument, and a minimum of one output-creation option (**--html**). The input XML file to use can be located inside the PXF file (see first and second examples above) or it can be an external XML file (located outside the PXF file; see third example 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 example above, the output file's location will be relative to the folder in which the PXF file is.
4.4 help

The help command takes a single argument (Command): the name of the command for which help is required. It displays the correct syntax of the command and other information relevant to the correct execution of the command.

Syntax

```
stylevisionserver help [options] Command
```

Casing and slashes on the command line

* 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.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command</td>
<td>The name of the command for which you want help information</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--h, --help</td>
<td>Displays information about the command</td>
</tr>
</tbody>
</table>

Options are listed in their short forms (if available) and long forms. You can use one or two dashes for both short and long forms. An option that takes a value is written like this: `--option=value`.

Examples

An example of the help command:

```
stylevisionserver help licenseserver
```

The command above contains one argument: the command `licenseserver`, for which help is required. When the example command above is executed, information about the `licenseserver` command will be displayed in the terminal.

The --help option

Help information about a command is also available by using the --help option with the command for which help information is required. For example: Using the --help option with the `licenseserver` Command, as follows:

```
stylevisionserver licenseserver --help
```
achieves the same result as does using the `help` command with an argument of `licenseserver`:

```bash
stylevisionserver help licenseserver
```

In both cases, help information about the `licenseserver` command is displayed.
# 4.5 licenseserver

On execution, the `licenseserver` command registers StyleVision Server with the LicenseServer specified by the `Server-Or-IP-Address` argument. For the `licenseserver` command to be executed successfully, the two servers must be connected on the network and LicenseServer must be running. You must have administrator privileges to be able to register StyleVision Server with LicenseServer.

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/documentation](https://www.altova.com/documentation)).

## Syntax

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

▼ Casing and slashes on the command line

<table>
<thead>
<tr>
<th>StyleVisionServer on Windows</th>
<th>stylevisionserver on Unix (Linux, Mac)</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Note that lowercase (stylevisionserver) works on all platforms (Windows, Linux, and Mac), while upper-lower (StyleVisionServer) works only on Windows and Mac.</td>
<td></td>
</tr>
<tr>
<td>* Use forward slashes on Linux and Mac, backslashes on Windows.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server-Or-IP-Address</td>
<td>Identifies the machine on the network on which Altova LicenseServer is installed and running. It can be the machine's name or its IP address.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--h, --help</td>
<td>Displays information about the command</td>
</tr>
<tr>
<td>-- j, --json</td>
<td>Values are <code>true</code></td>
</tr>
</tbody>
</table>

Options are listed in their short forms (if available) and long forms. You can use one or two dashes for both short and long forms. An option that takes a value is written like this: `--option=value`.

## 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.
4.6 pdfdata

The `pdfdata` command generates an FDF file or XML file from the PDF file that is submitted as the `InputPDF` argument. 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. If the PDF file does not have any form data, the generated XML file will contain no form data.

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

Syntax

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

> Casing and slashes on the command line

* **StyleVisionServer** on Windows

* **stylevisionserver** on 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.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputPDF</td>
<td>Path to the PDF file from which to generate the FDF or XML file</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>--outfdf</code></td>
<td>The path to the generated FDF file. Form: <code>--outfdf=FILE</code></td>
</tr>
<tr>
<td><code>--outxml</code></td>
<td>The path to the generated XML file. Form: <code>--outxml=FILE</code></td>
</tr>
<tr>
<td><code>--h, --help</code></td>
<td>Displays information about the command</td>
</tr>
</tbody>
</table>

Options are listed in their short forms (if available) and long forms. You can use one or two dashes for both short and long forms. An option that takes a value is written like this: `--option=value`.

**Note:** On Windows systems: Avoid using the end backslash and closing quote on the command line `\"`, for example, as in: "C:\My directory\". These two characters are interpreted by the command line parser as a literal double-quotation mark. Use the double backslash `\` if spaces occur in the command line and you need the quotes (for example: "C:\My Directory\") Alternatively, try to avoid using spaces and, therefore, quotes at all.

**Examples**

Examples of the `pdfdata` command:
StyleVision Server Command Line

```
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
```
4.7 setfopath

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

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.

After an FO processor has been specified with the `setfopath` command, it is this processor that will be used when PDF is generated with subsequent `generate` 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 will be located under `ProgramData\Altova\SharedBetweenVersions`; on Linux and macOS systems in a descendant folder of the `StyleVisionServer2019` folder.

Syntax

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

Casing and slashes on the command line

- StyleVisionServer on Windows
- stylevisionserver on 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.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Path to the Apache FO processor to use in subsequent FO generation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--h, --help</td>
<td>Displays information about the command</td>
</tr>
</tbody>
</table>

Options are listed in their short forms (if available) and long forms. You can use one or two dashes for both short and long forms. An option that takes a value is written like this: `--option=value`.

**Note:** On Windows systems: Avoid using the end backslash and closing quote on the command line `"`, for example, as in: "C:\My directory". These two characters are interpreted by the command line parser as a literal double-quatation mark. Use the double
If spaces occur in the command line and you need the quotes (for example: "C:\My Directory\"). Alternatively, try to avoid using spaces and, therefore, quotes at all.

**Examples**

After running the `setfopath` command, you can use the `generate` command to generate a PDF output-file 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.
4.8 setdeflang

The `setdeflang` command (short form is `sdl`) sets the default language of StyleVision Server. It takes a mandatory `LanguageCode` argument.

Syntax

```
stylevisionserver setdeflang | sdl [options] LanguageCode
```

Casing and slashes on the command line

- **StyleVisionServer on Windows**
- **stylevisionserver on 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.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LanguageCode</td>
<td>sets the default language of StyleVision Server.</td>
</tr>
<tr>
<td></td>
<td>Supported languages are: en, de, es, fr, ja</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--h, --help</td>
<td>Displays information about the command</td>
</tr>
</tbody>
</table>

Options are listed in their short forms (if available) and long forms. You can use one or two dashes for both short and long forms. An option that takes a value is written like this: `--option=value`.

Examples

An example of the `setdeflang` command:

```
stylevisionserver setdeflang DE
```

The command above sets the default language for messages to German.
4.9 verifylicense (Windows only)

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. This command is supported only on Windows systems. It is not supported on Linux or Mac systems. For details about licensing, see the LicenseServer documentation (https://www.altova.com/documentation).

Syntax

```
StyleVisionServer verifylicense [options]
```

Casing and slashes on the command line

* 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.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--l, license-key</td>
<td>The license key to check: whether it has been assigned to the product</td>
</tr>
<tr>
<td>--h, --help</td>
<td>Displays information about the command</td>
</tr>
</tbody>
</table>

Options are listed in their short forms (if available) and long forms. You can use one or two dashes for both short and long forms. An option that takes a value is written like this: `--option=value`.

Examples

Examples of the `verifylicense` command:

```
stylevisionserver verifylicense
stylevisionserver verifylicense --license-key=ABCDEFG-HIJKLMN-OPQRSTUVWXYZ12-3456789
```

The commands above check, respectively, whether the product has been licensed, and whether the product has been licensed with the license key given as the value of the `--license-key` option.
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
- About the COM Interface
- About the Java Interface
- Code Examples
- API Reference
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.

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 classes `StyleVision.Server`. These two classes 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

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

- **StyleVisionServer.jar** The library that communicates with StyleVision Server
- **StyleVisionServer_JavaDoc.zip** Documentation of the StyleVision Server API

Both files are available in the bin folder of the StyleVision Server installation folder. You can either reference the file from their original location or copy them to another location if this fits your project setup.

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**.
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++
- C#
- VBScript
- Visual Basic
- Java
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](#).

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

// 32-bit StyleVision Server
#import "progid:StyleVision.Server"
// 64-bit StyleVision Server
//#import "progid:StyleVision_x64.Server"

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

    try
    {
        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 = "c:\\temp";

        //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\StyleVisionServer2019\bin\StyleVisionServer.exe";

        //Prepare the name of the working XML
        pSVS->InputXML = "ExpReport.xml";

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

        //Prepare the parameters, if your design uses parameters
        //pSVS->AddParameter( "testparam1", "value 1" );
    }
    catch (...) 
    {
        // Handle exception
    }
    return 0;
}
```

/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->Run("C:\Program Files (x86)\Altova\StyleVisionServer2019\etc\Examples\ExpReport.pxf"))
    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;
5.4.2 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.

```csharp
namespace StyleVisionServerAPI_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            //Create a StyleVision Server object

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

            //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\StyleVisionServer2019\bin\StyleVisionServer.exe";

            //Prepare the name of the working XML
            objSVS.InputXML = "ExpReport.xml";

            //Add output paths (absolute or relative to WorkingDirectory) for all formats that should be generated
            objSVS.OutputRTF = "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.Run("C:\Program Files (x86)\Altova\StyleVisionServer2019\etc\Examples\ExpReport.pxf"))
                System.Console.WriteLine("Success - finished execution");
            else
                System.Console.WriteLine(objSVS.LastExecutionMessage);
        }
    }
}
```
5.4.3 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.

```vbscript
Option Explicit On

'Create a StyleVision Server object; use "StyleVision_x64.Server" if you want to use the 64-bit installation
Dim objSVS
Set objSVS = WScript.GetObject( "", "StyleVision.Server" )

'Set a working directory - used for output and for intermediate files
objSVS.WorkingDirectory = "C:\temp"

'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\StyleVisionServer2019\bin\StyleVisionServer.exe"

'Prepare the name of the working XML
objSVS.InputXML = "ExpReport.xml"

'Add output paths (absolute or relative to WorkingDirectory) for all formats that should be generated
objSVS.OutputRTF = "ExpReport.rtf"

'Prepare the parameters, if your design uses parameters
'Call 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.Run( "C:\Program Files (x86)\Altova\StyleVisionServer2019\etc\Examples\ExpReport.pxf" ) ) Then
    WScript.Echo( "Success - finished execution" )
Else
    WScript.Echo( objSVS.LastExecutionMessage )
End If
```
5.4.4 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.

Option Explicit On

Module Program

    Sub Main()

        'Create a StyleVision Server object

        'Set a working directory - used for output and for intermediate files
        objSVS.WorkingDirectory = "C:\temp"

        '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\StyleVisionServer2019\bin\StyleVisionServer.exe"

        'Prepare the name of the working XML
        objSVS.InputXML = "ExpReport.xml"

        'Add output paths (absolute or relative to WorkingDirectory) for all formats that should be generated
        objSVS.OutputRTF = "ExpReport.rtf"

        'Prepare the parameters, if your design uses parameters
        'Call 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.Run("C:\Program Files (x86)\Altova\StyleVisionServer2019\etc\Examples\ExpReport.pxf")) Then
            System.Console.WriteLine("Success - finished execution")
        Else
            System.Console.WriteLine(objSVS.LastExecutionMessage)
        End If

    End Sub

End Module
5.4.5 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. To check if registration was successful, see About the Java Interface.

```java
public class Program {
    public static void main(String[] args) {
        try {
            // Create a StyleVision Server object

            // Set a working directory - used for output and for intermediate files
            objSVS.setWorkingDirectory( "C:\\temp" );

            // Default path to the StyleVision Server executable is the installation path (same dir with the StyleVisionServer.jar)
            // In case you copied the JAR file to a new location, you need to explicitly set the path to the .exe file
            // objSVS.setServerPath( "C:\Program Files (x86)\Altova\StyleVisionServer2019\bin\StyleVisionServer.exe" );

            // Prepare the name of the working XML
            objSVS.setInputXML( "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("C:\Program Files (x86)\Altova\StyleVisionServer2019\etc\Examples\ExpReport.pxf") ){
                System.out.println( "Success - finished execution" );
            } else {
                System.out.println( objSVS.getLastExecutionMessage() );
            }
        }
        catch( Exception e ) {
            e.printStackTrace();
        }
    }
}
```
5.5 API Reference

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

- [COM and .NET](#)
- Java
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#**
  ```csharp
  string InputXML { set; }
  ```

- **C++**
  ```cpp
  HRESULT InputXML([in] BSTR bstrPath );
  ```

- **VB**
  ```vb
  Property InputXML As String
  ```

**Is64Bit**
Returns **TRUE** if the StyleVision Server engine is a 64-bit executable.

- **C#**
  ```csharp
  bool Is64Bit { get; }
  ```

- **C++**
  ```cpp
  HRESULT Is64Bit([out, retval] VARIANT_BOOL* pbVal);
  ```

- **VB**
  ```vb
  ReadOnly Property Is64Bit As Boolean
  ```

**LastExecutionMessage**
Gets the message received during the last *Generate* command.

- **C#**
  ```csharp
  string LastExecutionMessage { get; }
  ```

- **C++**
  ```cpp
  HRESULT LastExecutionMessage([out, retval] BSTR* pbstrResult );
  ```

- **VB**
  ```vb
  ReadOnly Property LastExecutionMessage As String
  ```

**MajorVersion**
Gets the major version of StyleVision Server.

- **C#**
  ```csharp
  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#
  ```csharp
  int MinorVersion { get; }
  ```

- C++
  ```cpp
  HRESULT MinorVersion([out, retval] INT* pnVal);
  ```

- VB
  ```vb
  ReadOnly Property MinorVersion As Integer
  ```

### OutputDOCX

Sets the path and name of the output DOCX file.

- C#
  ```csharp
  string OutputDOCX { set; }
  ```

- C++
  ```cpp
  HRESULT OutputDOCX([in] BSTR bstrPath);
  ```

- VB
  ```vb
  Property OutputDOCX As String
  ```

### OutputFO

Sets the path and name of the output FO file.

- C#
  ```csharp
  string OutputFO { set; }
  ```

- C++
  ```cpp
  HRESULT OutputFO([in] BSTR bstrPath);
  ```

- VB
  ```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

- ProductName

  Gets the name of the product: "StyleVision Server"

  - C#

    string ProductName { get; }

  - C++

    HHRESULT 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#**
  ```csharp
  string ProductNameAndVersion { get; }
  ```

- **C++**
  ```c++
  HRESULT ProductNameAndVersion([out, retval] BSTR* pstrVal);
  ```

- **VB**
  ```vbnet
  ReadOnly Property ProductNameAndVersion As String
  ```

### ServerPath

Gets or sets the path to the StyleVision Server executable.

- **C#**
  ```csharp
  string ServerPath { set; get; }
  ```

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

- **VB**
  ```vbnet
  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#**
  ```csharp
  int ServicePackVersion { get; }
  ```

- **C++**
  ```c++
  HRESULT ServicePackVersion([out, retval] INT* pnVal);
  ```

- **VB**
  ```vbnet
  ReadOnly Property ServicePackVersion As Integer
  ```

### WhereClause

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

- **C#**
  ```csharp
  string WhereClause { set; }
  ```
WhereClause

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
  ```java
  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
  ```java
  public void clearParameterList()
  ```
  Clears the list of parameters.

- generate
  ```java
  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
  ```java
  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
  ```java
  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.

getName
  public String getName()
  Gets the product name.

getNameAndVersion
  public String getNameAndVersion()
  Gets the complete name and version number of the product.

getServicePackVersion
  public int getServicePackVersion()
  Gets the service pack version of the StyleVision Server.

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**

```java
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**

```java
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**

```java
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**

```java
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.

**setServerPath**

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

Sets the path of the **StyleVisionServer** executable.

**Parameters:**
- `serverFile`: The path of the **StyleVisionServer** executable.

**setWhereClause**

```java
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**

```java
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.
Index

C
Command line usage, 29
Commands,
  exportresourcestrings, 33
  generate, 35
  help, 38
  licenseserver, 40
  pdfdata, 42
  setdeflang, 46
  setfopath, 44

E
exportresourcestrings command, 33

F
FlowForce Server,
  and StyleVision Server, 4

G
generate command, 35

H
help command, 38

I
Installation on Linux, 14
Installation on macOS, 23
Installation on Windows, 8

L
licenseserver command, 40
Licensing on Linux, 18
Licensing on macOS, 26
Licensing on Windows, 10
Linux,
  installation on, 14
  licensing on, 18
Localization, 33, 46

M
macOS,
  installation on, 23
  licensing on, 26

P
pdfdata command, 42

S
setdeflang command, 46
setfopath command, 44
Setup,
  on Linux, 13
  on macOS, 22
  on Windows, 7
StyleVision Server,
  features, 3
  functionality, 3
  in the FlowForce workflow, 4

W
Windows,
  installation on, 8
  licensing on, 10