

Altova FlowForce

User and Reference Manual

Altova FlowForce User & Reference Manual

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

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

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

Published: 2013

© 2013 Altova GmbH

Table of Contents

1	Altova FlowForce Server®	3
2	Administrator Guide	6
2.1	Architecture	7
2.2	FlowForce Server concepts	9
2.3	Getting started	11
2.3.1	Installation Windows	13
	<i>Initial setup - Windows</i>	14
2.3.2	Installation Linux	19
	<i>Initial setup - Linux</i>	20
2.3.3	Installation Mac OS X	26
	<i>Initial setup - Mac OS X</i>	27
2.4	Altova LicenseServer	29
2.4.1	Network Information	30
2.4.2	Installation (Windows)	31
2.4.3	Installation (Linux)	32
2.4.4	Installation (Mac OS X)	33
2.4.5	Altova ServiceController	34
2.4.6	How to Assign Licenses	35
	<i>Start LicenseServer</i>	35
	<i>Open LicenseServer Config Page (Windows)</i>	37
	<i>Open LicenseServer Config Page (Linux)</i>	39
	<i>Open LicenseServer Config Page (Mac OS X)</i>	41
	<i>Upload Licenses to LicenseServer</i>	43
	<i>Register FlowForce Server with LicenseServer</i>	46
	<i>Register MapForce Server with LicenseServer</i>	50
	<i>Register RaptorXML(+XBRL) Server with LicenseServer</i>	52
	<i>Register StyleVision Server with LicenseServer</i>	53
	<i>Assign Licenses to Registered Products</i>	54
2.4.7	Configuration Page Reference	59
	<i>License Pool</i>	59
	<i>Server Management</i>	62
	<i>Server Monitoring</i>	65
	<i>Settings</i>	65

	<i>Messages, Log Out</i>	67
2.5	FlowForce Server data storage	68
2.6	FlowForce Administration Interface	69
2.7	FlowForce access control	71
2.7.1	Users and Roles	72
	<i>How to add Users</i>	72
	<i>How to add Roles</i>	75
	<i>Defining restricted user rights</i>	77
2.7.2	Privileges	79
2.7.3	Permissions	81
	<i>How to add Permissions</i>	83
2.7.4	Credentials	86
2.8	Settings	89
2.9	Command Line Usage	90
2.9.1	help	92
2.9.2	createdb	93
2.9.3	debug	94
2.9.4	exportresourcestrings	95
2.9.5	foreground	97
2.9.6	initdb	98
2.9.7	install	99
2.9.8	licenseserver	100
2.9.9	repair	101
2.9.10	setdeflang (sdl)	102
2.9.11	start	103
2.9.12	uninstall	104
2.9.13	upgradedb	105
2.10	Altova MapForce Server	106
2.10.1	Installation on Windows	107
	<i>Licensing on Windows</i>	108
2.10.2	Installation on Linux	110
	<i>Licensing on Linux</i>	111
2.10.3	Installation Mac OS X	113
	<i>Initial setup - Mac OS X</i>	114
2.10.4	Deploying mappings to Servers	115
2.10.5	Command Line Usage	116
	<i>run</i>	116
	<i>licenseserver</i>	117
	<i>setdeflang</i>	118
	<i>help</i>	119

	<i>exportresourcestrings</i>	119
2.11	Altova StyleVision Server	121
2.11.1	Functionality	122
2.11.2	Installation on Windows	124
2.11.3	Installation on Linux	125
2.11.4	Installation on Mac OS X	127
2.11.5	Licensing on Windows	129
2.11.6	Licensing on Linux	130
2.11.7	Licensing on Mac OS X	132
2.11.8	Command Line Usage	134
	<i>help</i>	135
	<i>exportresourcestrings</i>	135
	<i>generate</i>	136
	<i>licenseserver</i>	138
	<i>setdeflang</i>	138
	<i>setfopath</i>	139
2.12	RaptorXML Server	141
2.12.1	RaptorXML Commands	143
2.12.2	Command Line Interface (CLI)	144
	<i>XML, DTD, XSD Validation Commands</i>	144
 <i>valxml-withdtd (xml)</i>	145
 <i>valxml-withxsd (xsi)</i>	146
 <i>valdtd (dtd)</i>	147
 <i>valxsd (xsd)</i>	148
 <i>valany</i>	149
	<i>Well-formedness Check Commands</i>	150
 <i>wfxml</i>	151
 <i>wfdtd</i>	152
 <i>wfany</i>	153
	<i>XBRL Validation Commands</i>	153
 <i>valxbrl (xbrl)</i>	154
 <i>valxbrltaxonomy (dts)</i>	156
 <i>valany</i>	157
	<i>XSLT Commands</i>	158
 <i>xslt</i>	159
 <i>valxslt</i>	160
	<i>XQuery Commands</i>	162
 <i>xquery</i>	162
 <i>valxquery</i>	163
	<i>Help and License Commands</i>	164
 <i>Help Command</i>	164
 <i>License Commands</i>	165
	<i>Localization Commands</i>	165
 <i>exportresourcestrings</i>	166
 <i>setdeflang</i>	166

<i>Options</i>	167
.....Catalogs.....	167
.....Errors.....	167
.....Global Resources.....	168
.....Help and Version.....	168
.....Messages.....	168
.....Processing.....	168
.....XBRL Evaluation.....	169
.....XBRL Schemas.....	169
.....XML Instance.....	170
.....XML Instance Validation.....	170
.....XML Schema Document (XSD).....	170
.....XQuery.....	172
.....XSLT.....	174
.....ZIP Files.....	175

3 FlowForce Tutorial 178

3.1 Deploying a MapForce mapping	180
3.2 Defining a job - triggers & execution steps	183
3.3 Defining a subjob	186
3.4 Directory polling - acting on a trigger file	189
3.5 Using parameters to query a database	192
3.6 Using a deployed mapping as a web service	195
3.7 Deploying a StyleVision transformation	199
3.8 Using RaptorXML Server to validate a document	203

4 User Guide 208

4.1 FlowForce Administration Interface	209
4.2 FlowForce concepts	210
4.3 Job configuration	212
4.3.1 Job input parameters	214
4.3.2 Execution steps	215
<i>Execution step</i>	215
<i>Choose step</i>	218
<i>For-each step</i>	220
<i>Error/success handling step</i>	220
<i>Step results</i>	222
.....Step expressions.....	222
4.3.3 Triggers	228
<i>Timer trigger types & common settings</i>	228
<i>Run Once</i>	229
<i>Run Daily</i>	230

	<i>Run On days of week</i>	230
	<i>Run On days of months</i>	230
	<i>Run On days in weeks of months</i>	231
	<i>File system trigger</i>	231
	<i>HTTP trigger</i>	232
4.3.4	Service	234
	<i>Technical details</i>	235
4.3.5	Queue settings	237
4.4	Credentials	238
4.5	Built-in functions	241
4.5.1	filesystem - File system functions	242
4.5.2	ftp - FTP client	244
4.5.3	mail - Sending E-mail	247
4.5.4	shell - Command line execution	248
4.5.5	compute - Evaluating expressions	249

Index

Chapter 1

Altova FlowForce Server®

1 Altova FlowForce Server®

FlowForce Server® is a new Altova product that allows you to automate and schedule the execution of MapForce mappings, StyleVision transformations and other tasks on dedicated high-speed servers. Windows and Linux operating systems are supported.

The FlowForce Server system consists of the following modules, which can be installed individually:

- FlowForce Server (including FlowForce Server Administration Interface)
- [License Server](#)
- [MapForce Server](#)
- [StyleVision Server](#)
- [RaptorXML Server](#)

For an overview please see: [FlowForce Server Architecture](#)

This documentation is in multiple parts:

- The [Administrator Guide](#) describes how to install, setup and maintain the server, as well as how to define the access control settings.
- The [Tutorial](#) shows you how to deploy a mapping from MapForce, define a scheduled job in FlowForce Server Administration Interface, and execute that job to produce output files.
- The [RaptorXML Server](#) page describes the various RaptorXML editions and how to use it from within FlowForce Server.
- The [User Guide](#) describes the browser application in more detail; the different trigger types and the various execution steps. It also describes how you can change the input/output files supplied by the deployed mapping when the job executes.

Note:

The FlowForce Server administration interface does not support SSL.

What's new in Altova FlowForce Server® 2013R2

- Integration with [RaptorXML Server](#) and RaptorXML Development editions
- Job [flow control](#) allowing the execution of job steps based on conditions
- Ability to [repeat execution](#) steps any number of times
- Definition of [step variables](#) allowing the results of one step to be used in following job steps
- An expanded set of [built-in steps](#) allowing mail notifications, FTP server interaction, and the ability to compute expressions

Chapter 2

Administrator Guide

2 Administrator Guide

The Administrator guide focuses on the specifics of FlowForce Server namely:

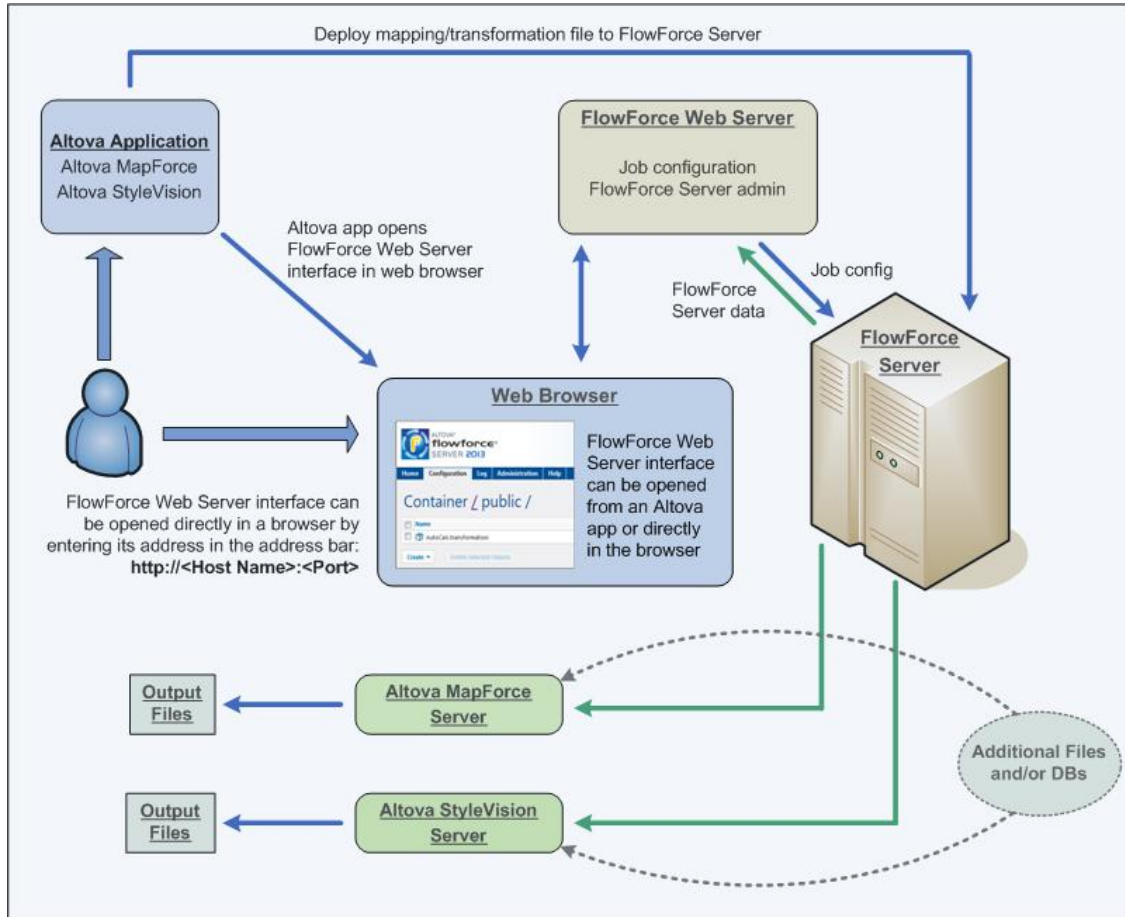
- [Getting started](#) with FlowForce Server, i.e. how to install and start FlowForce Server
- Data storage and the various configuration files of [FlowForce Server](#)
- How to define the [Access Control](#) settings
- The [architecture](#) of the FlowForce Server system
- The FlowForce Server [concepts](#)

Please note:

You must install LicenseServer to be able to work with FlowForce Server.

2.1 Architecture

A FlowForce Server installation consists of several server modules and other components, that can be selected during the installation process.



FlowForce Server Web Administration Interface

This is the module that acts as the front-end to FlowForce Server. It is a standalone web application that is installed on the same machine as FlowForce Server.

The user interface runs in an internet browser and allows administrators to configure access control settings as well as the specific server operations including jobs, triggers, etc.

FlowForce Server supports the current versions of Mozilla Firefox, Google Chrome, and Microsoft Internet Explorer 9 and 8. Note: When using Internet Explorer 9 as your browser, please disable the "Show friendly HTTP error messages" in the Advanced tab, to view the HTML form when using FlowForce jobs as web services.

FlowForce Server

[FlowForce Server](#) is the core of the FlowForce Server system and runs as a background service without a graphical user interface. FlowForce Server continuously checks for trigger conditions, starts and monitors job execution, and writes detailed logs. To execute job steps that use a deployed MapForce mapping, FlowForce Server sends an execution request to MapForce Server.

Altova LicenseServer

[LicenseServer](#) is a service that allows you to manage Altova licenses from one central location using a browser. The license server is installed on a server that all other servers and clients can access. Server licenses can be bound to specific machines and clients can be bound to specific servers.

Note:

LicenseServer must be installed to use FlowForce Server as well as **any** of the other server-based products (MapForce Server, Stylevision Server, or RaptorXML Server).

MapForce

The MapForce graphical mapping application has been enhanced with an integrated deployment feature. Once a mapping has been tested and debugged, MapForce lets you deploy it to FlowForce Server. The newly deployed mapping is then immediately available for use in any job on the server.

An administrator or developer runs MapForce on a personal Windows workstation to develop and deploy mappings onto the high-speed server.

MapForce Server

[MapForce Server](#) is an implementation of the MapForce Built-in execution engine that executes mappings previously deployed via the MapForce graphical environment. MapForce Server is always installed on the same machine as FlowForce Server.

Stylevision

StyleVision allows you to design reports and forms based on XML, SQL database, and XBRL inputs. Once a stylesheet has been tested and debugged it can be deployed to FlowForce Server. The deployed files are then available for use in any transformation job on the server.

Stylevision Server

[StyleVision Server](#) is an implementation of the stand-alone version of StyleVision that executes transformations previously deployed via the StyleVision graphical environment. StyleVision Server is always installed on the same machine as FlowForce Server.

RaptorXML (+XBRL) Server

Altova [RaptorXML Server](#) is Altova's third-generation, super-fast XML and XBRL processor and validates XML documents, checks the well-formedness of XML documents, and transforms XSLT and XQuery documents. RaptorXML Server is always installed on the same machine as FlowForce Server.

2.2 FlowForce Server concepts

Configuration

Configuration data in FlowForce Server's database are comprised of various objects that define the operation of FlowForce Server. This includes jobs, credentials, functions, triggers, and other objects.

Configuration objects are organized in a freely defined hierarchy of containers. Some configuration settings are edited together (e.g. jobs include triggers), and other settings can also be stored as standalone objects under their own name (e.g. credentials and functions).

Container

A container is similar to a folder in a commonly used file system. It is used to create a hierarchical structure for storing configuration objects and other containers. Containers can be assigned access permissions.

Two predefined containers exist in FlowForce Server: /system which contains system functions, e.g. copy, move, etc., and /public which is the default container when deploying a mapping to FlowForce Server from MapForce. Other containers can be created as needed, e.g. for departments or user groups.

Function

A FlowForce Server function performs a specific operation when used in a job execution step. It may have input parameters that need to be passed to it by the caller. Available functions include the [system functions](#) delivered with FlowForce Server, deployed MapForce mappings or StyleVision transformations, and the execution steps of other jobs.

Job

A Job consists of Triggers, Execution steps, input parameters, and other settings. Triggers define when a job will be executed, and the execution steps define what the job actually does when it executes. Multiple triggers and execution steps can be defined per job.

Trigger

Triggers define under which circumstances a job will be executed. Three types of triggers can currently be defined: [Timer triggers](#), [File system triggers](#), and [HTTP triggers](#). Multiple triggers can be defined per job.

Service

FlowForce Server permits exposing jobs as web services via the HTTP protocol. This allows interactive or automated access to these jobs.

Credential

Credentials are stored login data used to execute FlowForce Server jobs. Credentials can be defined as standalone "objects" and be assigned to various jobs, or they can be manually entered for a specific job.

Queue

The queue settings in a FlowForce Server job allow limiting the number of parallel job executions to control use of server resources.

Access Control

All important operations in FlowForce Server are linked to permissions or privileges which need to be assigned to the user to successfully execute them.

User

FlowForce Server users are persons that have been added to FlowForce Server by the FlowForce Server administrator with a login name and a password. Depending on the assigned rights and privileges, users can define FlowForce Server jobs, deploy mappings, or view logs.

Two special users are predefined by FlowForce Server: "root" is the initial administrator user, and "anonymous" is a special user account used for FlowForce Server services that should be available to users without explicit log in to FlowForce Server.

Role

Roles are used to manage privileges and object permissions for user groups as opposed to individual users.

Having defined users, you can assign them to a role thus creating user groups. The users become "members" assigned to the specific role.

Permission

Permissions control access to containers and configurations. Unlike privileges they can be redefined on every level of the container hierarchy, and are by default inherited from parent containers.

Permissions, like privileges, are inherited from all roles the user is a member of, as well as from permissions directly assigned to the user.

Privilege

Privileges control user rights globally. This means privilege settings cannot be overridden in the container hierarchy of FlowForce Server.

When a user logs into FlowForce Server, the set of effective privileges is determined by the user privileges and all role privileges the user is member of.

2.3 Getting started

This section deals with what the first-time user of FlowForce, the administrator, has to do to set up the software and configure it for multiple users. Note, it is not necessary to add new users, roles etc. when you work through the tutorial. The tutorial makes use of default users and roles, built-in to FlowForce.

Install the software

To install FlowForce, LicenseServer and other Altova Server products, see: [Installation Windows](#), [Installation Linux](#), or [Installation Max OS X](#)

Register FlowForce Server and other Altova Server products

To register FlowForce Server and all other Altova Server products with Altova LicenseServer, see [LicenseServer](#).

Start FlowForce Server

To start FlowForce Server and change your default password to something new, see: [Initial setup - Windows](#), [Initial setup - Linux](#), or [Initial setup - Mac OS X](#)

Request evaluation license

To request evaluation licenses for the Altova Server products, see: [Server Management \(LicenseServer\)](#)

Add new users

To add new users, see: [Users](#).

- Users are persons that are allowed to define and/or start jobs.
- Note that users inherit privileges from all their roles in addition to the privileges defined here, so it is better to define them in the roles page.

Add new roles and define the role privileges

To add new roles and define privileges, see: [Roles](#).

- The Role page lets you create new roles and define the role privileges.
- Roles are used to manage privileges and object permissions for user groups instead of individual users.
- This is the place that you define **role privileges**, as the privileges defined here are automatically inherited by users when you assign a role to a user.
- Having defined the users in the previous step, you can now assign users to a role, thus creating user groups assigned to the various roles. (The users become "members" assigned to the role).

To assign users to a role, see: [Assigning a user to a role](#).

Define the work environment (container structure) and the read/write/use permissions of your users

To add new permissions to the permission list see: [Permissions](#).

- Containers are used to organize jobs, deployed MapForce mappings and StyleVision transformations into a hierarchy similar to that of a file system composed of folders.
- Read/Write/Use permissions should generally only be assigned to **roles**, not to

individual users (although this is possible).

Define the necessary credentials, i.e. the login data needed for FlowForce Server to access your operating system user accounts

To add new credentials, see: [Credentials](#).

- Credentials are stored **login** data used to execute FlowForce Server jobs, and are stored in the FlowForce Server database as separate objects.
- Jobs are started automatically by FlowForce server, when the defined trigger conditions are met. FlowForce server then runs these jobs using a specific operating system **user account**, ensuring that execution steps do not access unauthorized data.
- Every job **MUST** have a credential assigned to it for the execution steps to be executed. This defines the **operating system** user account used to run the job execution steps. It is vital for the success of the job execution that the **operating system user** which is referenced by this credential has sufficient [access permissions](#).

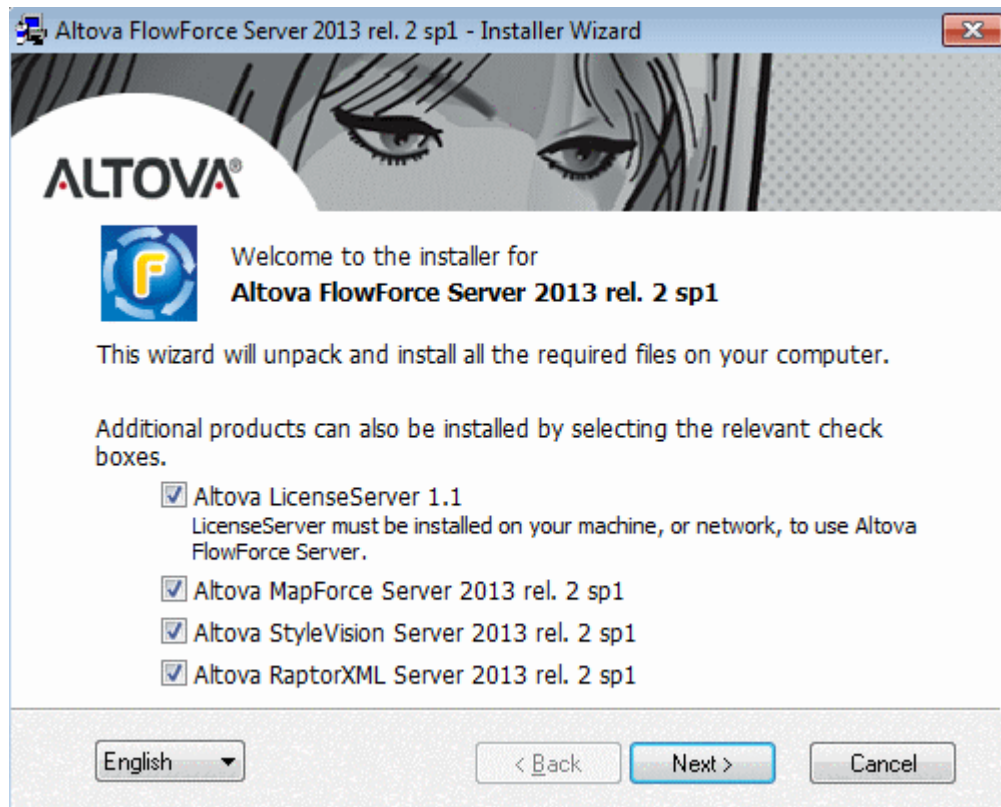
2.3.1 Installation Windows

Note:

All Altova Server products running under Windows have a minimum requirement: Windows XP with Service Pack 3.

Having downloaded FlowForce Server from the Altova website [download page](#), making sure to select the correct operating system:

1. Double click the installation file to start the installation process.
2. Select the extra servers that you also want to install: MapForce Server, StyleVision Server, or RaptorXML Server.



3. Make sure that you also install and start the [LicenseServer](#) licensing process, when installing FlowForce Server. This step is not necessary if LicenseServer is already running somewhere in your network.
4. Follow the wizard instructions to install the software.

Note:

You can select the installation language using the combo box in the bottom left of the wizard. The currently supported languages are: English, German, Spanish, and Japanese.

The language you select here also determines the language of the FlowForce Server user interface in the web browser.

File paths in Windows

File paths given in this documentation will not be the same for all operating systems. You

should note the following locations:

- FlowForce Server stores all data in the following locations:

Windows XP	C:\Documents and Settings\All Users\Application Data\Altova\FlowForceServer2013
Windows Vista, Windows 7/8	C:\ProgramData\Altova\FlowForceServer2013

- Application folder: The Application folder is the folder where your Altova application is located. The path to the Application folder is, by default, the following.

Windows XP	C:\Program Files\Altova
Windows Vista, Windows 7/8	C:\Program Files\Altova
32 bit Version on 64-bit OS	C:\Program Files (x86)\Altova

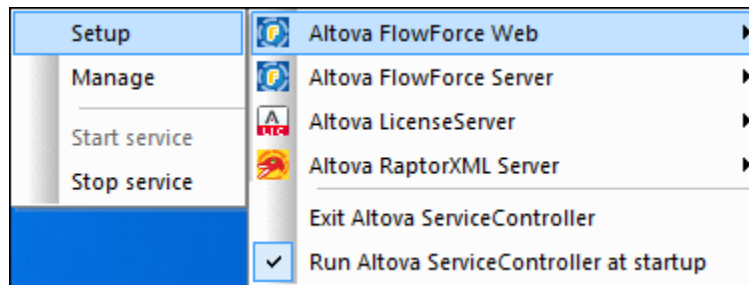
Next: [Starting FlowForce](#)

Initial setup - Windows

To register FlowForce Server with LicenseServer

- Click the Windows "Start" button and select **All Programs | Altova FlowForceServer | FlowForceServer Setup Page**.

You can also open the Setup page by clicking the Altova ServiceController icon in the system tray, mouse over **Altova FlowForce Web** in the menu that pops up (see *screenshot below*), and then select **Setup** from the submenu.



ALTOVA®
flowforce®
SERVER 2013

Home Help

Setup

Note: Changing the LicenseServer or any IP or port configuration, will only take effect after **restarting** the FlowForce services.

LicenseServer

127.0.0.1 [Browse] [Edit]

Register with LicenseServer

FlowForce Web Server

Bind address: Local only (127.0.0.1) 127.0.0.1 Port: 8082
Default time zone: Europe/Berlin


FlowForce Server

Bind address: Local only (127.0.0.1) 127.0.0.1 Port: 4646

Apply settings and restart FlowForce services

Altova FlowForce® 2013r2 - Copyright © 2011-2013, Altova GmbH

The Setup page appears in a browser window.

2. Click the browse button  of the LicenseServer group and select your LicenseServer from the list.
3. Click the "Register with LicenseServer" button to register with LicenseServer.
This opens the Altova Server Software License Agreement.
4. Click the "Accept" button of the agreement page if you agree to the license terms.
5. Click into the LicenseServer password field, enter the default password "default", and click the "Login" button.
This opens the Server Management tab of LicenseServer where you can assign a license to FlowForce server, please see [Assign licenses to registered products](#).

To configure network interfaces and ports

The default address and port will usually work fine, except if other services on the machine already use one of the ports, in which case you can change the ports used by FlowForce here.

1. Return to the FlowForce setup page.

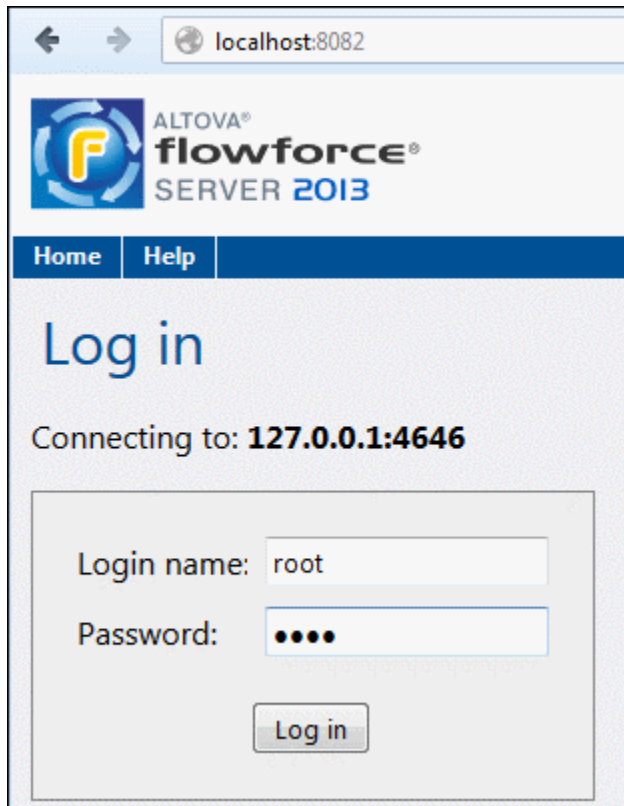
2. Configure the bind address and port for the FlowForce Web Server. By default, the web interface is available to users on all network interfaces on port 8082.
3. Set the default time zone to use in the web interface.
4. Configure the bind address and port for the FlowForce Server. The default setting for the server accepts only requests from the same machine (127.0.0.1). If you intend to start jobs as web services via HTTP from remote machines, select "All interfaces (0.0.0.0)" from the Bind address combo box.
5. Click "Apply settings and restart FlowForce services". The FlowForce services will restart, and your browser will be redirected to the Login page.

Note:

The FlowForce Server services are automatically started on every machine startup. Use the Windows control panel to disable the services if necessary. The "Services" management console can be found in "Administrative Tools", and can also be started using Start | Run | services.msc.

To log in to the FlowForce Web Administration Interface:

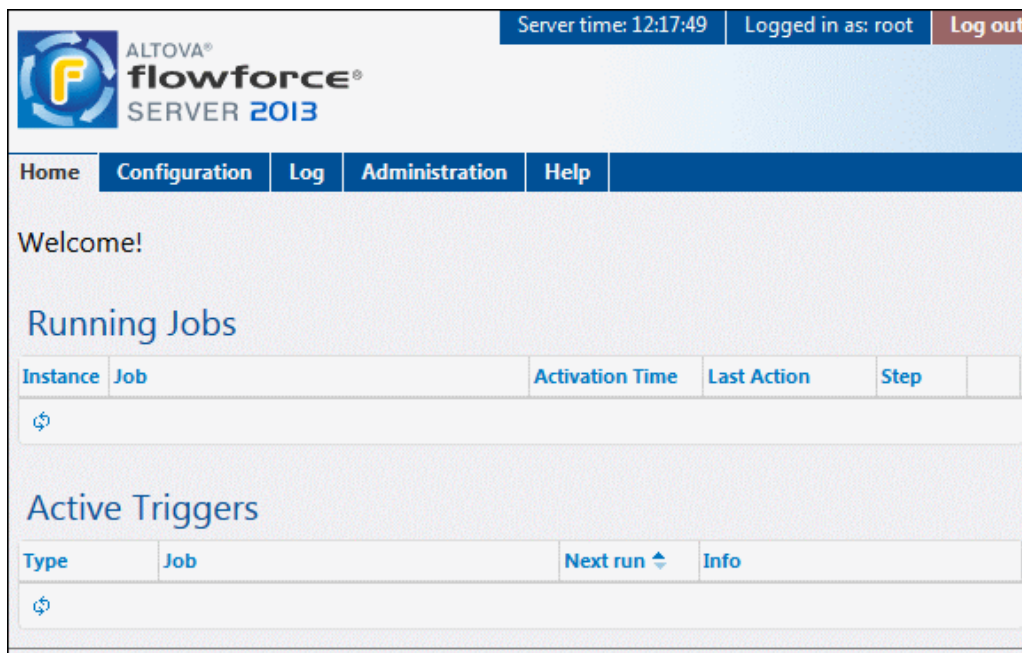
1. Start your browser and enter <http://localhost:8082>. If you changed the port on the FlowForce Server Configuration page, use the one you entered there. You are now connected to FlowForce Web Server and the Login page for FlowForce Server is opened.



Enter login name "**root**", as well as the password "**root**" if this is the first time that you have started FlowForce Server.

2. Click the "Log in" button to log in. You have now logged onto FlowForce Server.

Connection information, as well as any running jobs and active triggers are visible on the Home screen.



ALTOVA®
flowforce®
SERVER 2013

Server time: 12:17:49 | Logged in as: root | Log out

Home | Configuration | Log | Administration | Help

Welcome!

Running Jobs

Instance	Job	Activation Time	Last Action	Step
[Refresh]				

Active Triggers

Type	Job	Next run	Info
[Refresh]			

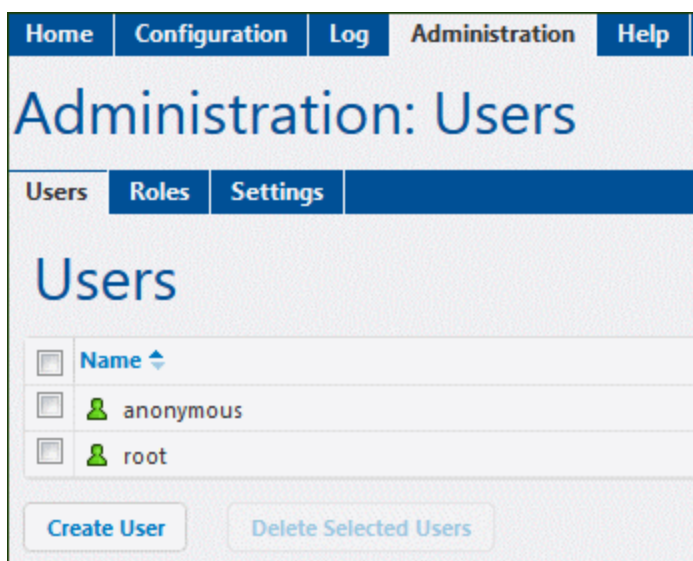
Logging out:

Click the "Log out" button at the far right of the browser window to log out.

To change your default password:

From the Home page shown above:

1. Click the "Administration" button, then the "Users" button.



Home | Configuration | Log | Administration | Help

Administration: Users

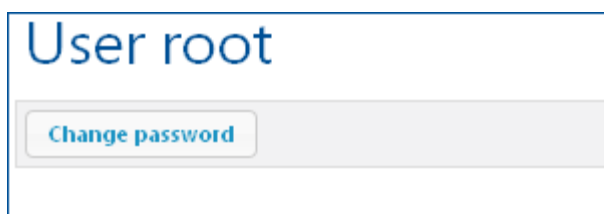
Users | Roles | Settings

Users

<input type="checkbox"/>	Name
<input type="checkbox"/>	anonymous
<input type="checkbox"/>	root

Create User | Delete Selected Users

2. Click the "root" user entry in the Users table.
3. Click the "Change password" button and enter your old and new passwords.

A screenshot of a web interface for a user named 'root'. The interface is enclosed in a blue border. At the top, the text 'User root' is displayed in a large, blue, sans-serif font. Below this, there is a horizontal bar with a light gray background. On the left side of this bar is a button with a rounded rectangle shape, containing the text 'Change password' in a blue, sans-serif font. The rest of the bar is empty.

4. Click Save to complete the process.

2.3.2 Installation Linux

Packages

Installation packages are available for:

Distribution	Package extension
Debian 6	.deb
Ubuntu 12.04	.deb
CentOS 6	.rpm

The components of FlowForce Server are provided in separate packages and can be downloaded from the [Altova website](#):

Package name	Description
flowforceserver	Required for all FlowForce Server installations. Contains the FlowForce Server engine and the FlowForce Web Administration Interface.
licenseserver	Required if you do not already have an Altova LicenseServer running in your network
mapforceserver	Required to run deployed MapForce mappings
stylevisionserver	Required to run deployed StyleVision transformations

Download the appropriate packages from the [Altova website](#) to your Linux computer and store them in any directory.

Installation

The installation must be done as the root user. If you are logged in as root, leave out the "sudo" prefix when typing the following commands.

Uninstalling old versions

On the Linux command line, you can check which Altova server products are installed with the following command:

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

If FlowForce Server is not installed, go ahead with the installation as documented in the next steps. If FlowForce Server is installed and you wish to install a newer version of FlowForce Server, uninstall the old version with the command:

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

If you need to uninstall other packages, use the same command as above and replace "flowforceserver" with the package name of the package you want to remove. For example:

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

Installing FlowForce Server

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

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

Install FlowForce Server with the following command:

```
[Debian]:    sudo dpkg --install flowforceserver-2013-debian.deb
[Ubuntu]:    sudo dpkg --install flowforceserver-2013-ubuntu.deb
[CentOS]:    sudo rpm -ivh flowforceserver-2013-1.x86_64.rpm
```

Installing Altova LicenseServer

In order for FlowForce Server to run, it must be licensed via an Altova LicenseServer on your network. Download Altova LicenseServer package from the [Altova website](#) to any directory on the Linux system. Install it just like you did FlowForce Server (*see previous step*).

```
[Debian]:    sudo dpkg --install licenseserver-1.0-debian.deb
[Ubuntu]:    sudo dpkg --install licenseserver-1.0-ubuntu.deb
[CentOS]:    sudo rpm -ivh licenseserver-1.0-1.x86_64.rpm
```

Installing MapForce Server and StyleVision Server

Install these packages just like you did FlowForce Server (*see previous step*).

Setting services to start automatically

On Ubuntu and CentOS, the services are set to start automatically by default. On Debian, use the `update-rc.d` command to configure the runlevels.

The next step is the [initial setup](#) to link FlowForce Server to LicenseServer, and to configure service interfaces and ports.

File paths in Linux

- Application path

Linux	/opt/Altova/FlowForceServer2013/bin
-------	-------------------------------------

- Data folder

Linux	/var/opt/Altova/FlowForceServer2013
-------	-------------------------------------

Initial setup - Linux

Starting LicenseServer as a service

If LicenseServer is not already running on a different server in your network, LicenseServer must be installed and running as a service on the same machine as FlowForce Server. Start LicenseServer as a service with the following command:

```
[Debian]: sudo /etc/init.d/licenseserver start
[Ubuntu]: sudo initctl start licenseserver
[CentOS]: sudo initctl start licenseserver
```

(If you need to stop LicenseServer, replace `start` with `stop` in the above command)

Starting FlowForce Web Administration Interface

The initial setup can be performed in a browser-based interface. Start FlowForce Web Server as a service with the following command:

```
[Debian]: sudo /etc/init.d/flowforcewebserver start
[Ubuntu]: sudo initctl start flowforcewebserver
[CentOS]: sudo initctl start flowforcewebserver
```

On first run and without any custom configuration files FlowForceWebServer will start on a random port and provide a setup page.

If your server machine has a GUI web browser, you can then open the setup page using the following URL:

```
file:///var/opt/Altova/FlowForceServer2013/flowforceweb.html
```

If your browser is running on a different machine:

On Debian, the URL to the setup page appears in the terminal window. On other distributions, you need to extract the URL to the setup page from the log file using the following command:

```
grep running /var/opt/Altova/FlowForceServer2013/data/ffweb.log
```

The output is similar to:

FlowForceWeb running on **http://127.0.0.1:34597/setup?key=52239315203**

Type this link into the address bar of your browser (and replace "127.0.0.1" with the host name of your server machine).

Firewall

If you use the setup page for your first time FlowForceServer configuration please make sure the random port address FlowForceWebServer was started on is not blocked from your firewall.

To register FlowForce Server with LicenseServer

Having followed the Linux installation procedure and started LicenseServer and FlowForce Web Server:

1. Open the setup page in your web browser as described above.

ALTOVA® flowforce® SERVER 2013

Home Help

Setup

Note: Changing the LicenseServer or any IP or port configuration, will only take effect after **restarting** the FlowForce services.

LicenseServer

127.0.0.1 [Browse] [Edit]

[Register with LicenseServer](#)

FlowForce Web Server

Bind address: Local only (127.0.0.1) 127.0.0.1 Port: 8082

Default time zone: Europe/Berlin


FlowForce Server

Bind address: Local only (127.0.0.1) 127.0.0.1 Port: 4646

[Apply settings and restart FlowForce services](#)

Altova FlowForce® 2013r2 - Copyright © 2011-2013, Altova GmbH

The Setup page appears in a browser window.

2. Click the browse button  of the LicenseServer group and select your LicenseServer from the list.
3. Click the "Register with LicenseServer" button to register with LicenseServer.
This opens the Altova Server Software License Agreement.
4. Click the "Accept" button of the agreement page if you agree to the license terms.
5. Click into the LicenseServer password field, enter the default password "default", and click the "Login" button.
This opens the Server Management tab of LicenseServer where you can assign a license to FlowForce server, please see [Assign licenses to registered products](#).

To configure network interfaces and ports

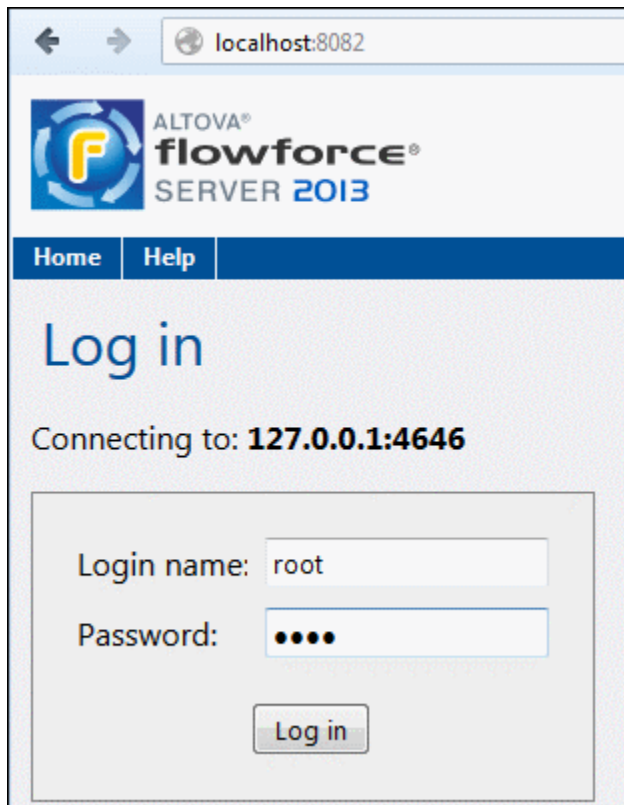
The default address and port will usually work fine, except if other services on the machine already use one of the ports, in which case you can change the ports used by FlowForce here.

1. Return to the FlowForce setup page.

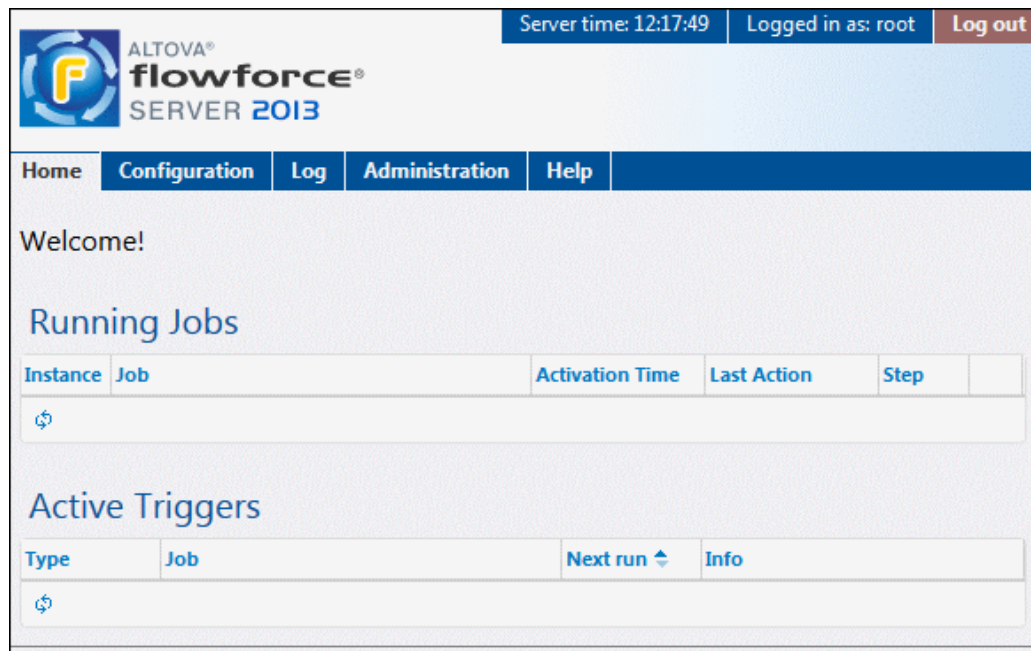
2. Configure the bind address and port for the FlowForce Web Server. By default, the web interface is available to users on all network interfaces on port 8082.
3. Set the default time zone to use in the web interface.
4. Configure the bind address and port for the FlowForce Server. The default setting for the server accepts only requests from the same machine (127.0.0.1). If you intend to start jobs as web services via HTTP from remote machines, select "All interfaces (0.0.0.0)" from the Bind address combo box.
5. Click "Apply settings and restart FlowForce services". The FlowForce services will restart, and your browser will be redirected to the Login page.

To log in to the FlowForce Web Administration Interface:

1. Start your browser and enter <http://localhost:8082>. If you changed the port on the FlowForce Server Configuration page, use the one you entered there. You are now connected to FlowForce Web Server and the Login page for FlowForce Server is opened.



- Enter login name "**root**", as well as the password "**root**" if this is the first time that you have started FlowForce Server.
2. Click the "Log in" button to log in. You have now logged onto FlowForce Server. Connection information, as well as any running jobs and active triggers are visible on the Home screen.

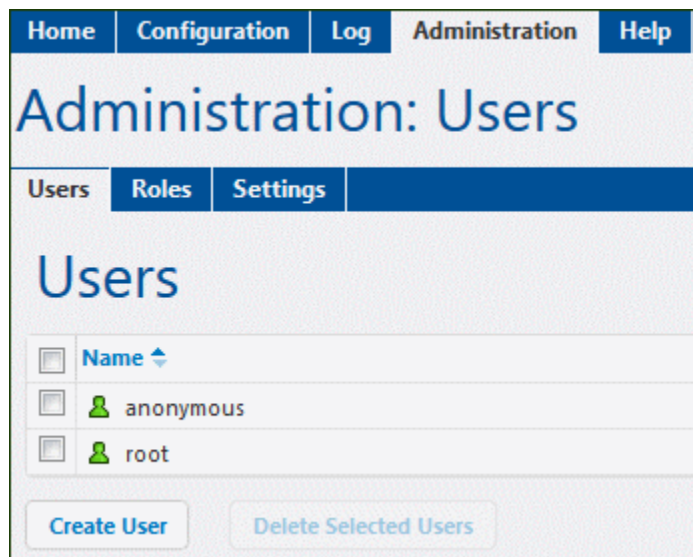
**Logging out:**

Click the "Log out" button at the far right of the browser window to log out.

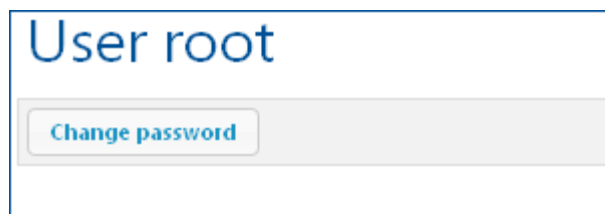
To change your default password:

From the Home page shown above:

1. Click the "Administration" button, then the "Users" button.



2. Click the "root" user entry in the Users table.
3. Click the "Change password" button and enter your old and new passwords.

A screenshot of a software interface showing a dialog box titled "User root". Inside the dialog, there is a button labeled "Change password". The dialog has a light gray background and a blue border.

4. Click Save to complete the process.

2.3.3 Installation Mac OS X

FlowForce Server can be installed on Mac OS X systems (version 10.7 or higher). Since you might need to uninstall a previous version, uninstalling is described first.

Uninstalling old versions of FlowForce Server and LicenseServer

Before uninstalling FlowForce Server, stop the service with the following command:

```
sudo launchctl unload /Library/LaunchDaemons/com.altova.FlowForce
Server.plist
```

To check whether the service has been stopped, open the Activity Monitor terminal and make sure that FlowForce Server is not in the list.

In the Applications terminal, right-click the FlowForce 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/FlowForceServer2013/
```

If you need to uninstall an old version of Altova LicenseServer, use the same procedure outlined above for FlowForce Server.

```
sudo launchctl unload /Library/LaunchDaemons/com.altova.LicenserServer11.
plist
```

Downloading the Mac OS X package

After downloading the Linux package from the [Altova website](#), copy the package to any directory on the Linux system.

Since you will need an [Altova LicenseServer](#) in order to run FlowForce Server, you may want to download LicenseServer from the [Altova website](#) at the same time as you download FlowForce Server, rather than download it at a later time. The Mac OS X installer file has a `.pkg` file extension.

Installing FlowForce Server

In a terminal window, switch to the directory where you have copied the installer file, and double-click it. 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.

The FlowForce Server package will be installed in the folder:

```
/usr/local/Altova/FlowForceServer2013/
```

Clicking the FlowForce Server icon in the Application terminal pops up the onscreen help ([this documentation](#)).

Installing Altova LicenseServer

For FlowForce Server to run, it must be licensed via an Altova LicenseServer on your network. On Mac OS X systems, [Altova LicenseServer](#) will need to be installed separately.

Download Altova LicenseServer from the [Altova website](#) and double-click the installer package to start the installation. Follow the on-screen instructions. You will need to accept the license agreement for installation to proceed.

The LicenseServer package will be installed in the folder:

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

For information about how to register FlowForce Server with [Altova LicenseServer](#) and license it, see the section, [Initial setup - Mac OS X](#).

Initial setup - Mac OS X

Licensing procedure

To license FlowForce Server on Mac OS X systems, do the following:

1. If LicenseServer is not already running as a service, **start** it as a service.
2. **Start** FlowForce Server as a service.
3. **Register** FlowForce Server with LicenseServer.
4. In the configuration page of LicenseServer, **assign** a license to FlowForce Server machine. How to do this is described in the [Altova LicenseServer documentation](#).

Note: You must have both FlowForce Server and [Altova LicenseServer](#) installed and running as services. See the section [Installation on Mac OS X](#) for information about installing these packages.

You must have administrator (root) privileges to be able to register FlowForce Server with LicenseServer.

Starting LicenseServer as a service

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

```
sudo launchctl load /Library/LaunchDaemons/com.altova.LicenseServer11.plist
```

If at any time you need to stop LicenseServer, use:

```
sudo launchctl unload /Library/LaunchDaemons/com.altova.LicenseServer11.plist
```

Starting FlowForce Server as a service

Start FlowForce Server as a service with the following command:

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

If at any time you need to stop FlowForce Server, use:

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

Registering FlowForce Server

Before assigning a license to FlowForce Server from LicenseServer, FlowForce Server must be

registered with LicenseServer. You can register FlowForce Server by using the [licenseserver](#) command of its CLI. Note that FlowForce Server must be started with root rights.

```
sudo /usr/local/Altova/FlowForceServer2013/bin/FlowForceServer  
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 FlowForce Server executable is:

```
/usr/local/Altova/FlowForceServer2013/bin
```

After successfully registering FlowForce Server, you can go to LicenseServer and assign a license to FlowForceServer. How to do this is described in the [Altova LicenseServer documentation](#).

2.4 Altova LicenseServer

Altova LicenseServer (hereafter also called LicenseServer) provides a central location for the management of licenses for Altova products. Altova applications running in a network can have licenses assigned to them from the LicenseServer, thus giving administrators the flexibility to manage and monitor licenses.

Licensing process with Altova LicenseServer

To assign an Altova server product a license using Altova LicenseServer, you need to do the following:

1. [Start LicenseServer](#).
2. Open the [LicenseServer Configuration page](#), which is the administrator's interface with LicenseServer, on [Windows](#) or [Linux](#).
3. [Upload the license/s](#) you have received from Altova to the license pool of your Altova LicenseServer. Do this in the [License Pool](#) tab of the LicenseServer Configuration page.
4. Register the Altova server product ([FlowForce Server](#), [MapForce Server](#), [StyleVision Server](#), [Register RaptorXML\(+XBRL\) Server with LicenseServer](#)) with LicenseServer. Depending on the product's type, the method of registering it with LicenseServer will be different: either via the product's GUI or its command line. See the documentation of your Altova server product for information about how to register it with LicenseServer.
5. In the [Server Management](#) tab of the LicenseServer Configuration page, [assign a license](#) to the Altova server product according to the number of cores on the product machine.

Licenses can thereafter be conveniently monitored and managed centrally with LicenseServer. See the [Configuration Page Reference](#) for available functionality.

About this documentation

This documentation is organized into the following parts:

- Introductory information about: [network requirements](#); installation on [Windows](#) and [Linux](#); and [Altova ServiceController](#).
- [How to Assign Licenses](#), which describes in a step-by-step way how to assign licenses with Altova LicenseServer.
- [Configuration Page Reference](#): A description of the administrator's interface with LicenseServer.

Note:

The LicenseServer administration interface does not support SSL.

2.4.1 Network Information

Altova LicenseServer must be installed on a server machine that is accessible by all clients running Altova products that require a license. Any firewall on both the client and server must allow the network traffic to and from the LicenseServer that is necessary for the LicenseServer to operate correctly.

On the LicenseServer, **port 35355** is used to distribute licenses, and therefore it must be open for network traffic with client machines.

The following are the default networking parameters and requirements of LicenseServer:

- *For LicenseServer license distribution:*
Either one or both of
IPv4 TCP connection on port 35355
IPv6 TCP connection on port 35355

For administrative tasks, The LicenseServer is accessed by a web interface that uses port 8088. The port used can be [configured to suit your requirements](#).

Connection to the Master Licensing Server at altova.com

The Altova LicenseServer needs to be able to communicate with the Master Licensing Server at altova.com to validate and authenticate license-related data and to ensure continuous compliance with the Altova license agreements. If the Altova LicenseServer is unable to connect with the altova.com Master Licensing Servers for a duration of more than 5 days (= 120 hours), then the Altova LicenseServer will no longer permit the usage of any Altova software products connected to the Altova LicenseServer.

Any such loss of connection with the altova.com master servers will be logged in the [Messages tab](#) of the [Configuration page of the Altova LicenseServer](#). In addition, the administrator can configure the Altova LicenseServer to automatically send an alert email when the connection to altova.com is lost. Alert Mail settings are available in the [Settings tab](#) of the [Configuration page](#).

2.4.2 Installation (Windows)

Altova LicenseServer can be installed on Windows systems in one of two ways:

- As an independent installation.
- As part of an Altova server product installation. (Altova server products are: Altova FlowForce Server, Altova MapForce Server, and Altova SyleVision Server.)

If LicenseServer is not installed on your system at the time an Altova server product is installed, the option to install LicenseServer is selected by default during installation setup. If LicenseServer is already installed, the option to install it is deselected by default. You can change the default option if you like.

Note: If you wish to re-install Altova LicenseServer, you must first de-install the older version.

For information about how to proceed with assigning licenses, see the section [How to Assign Licenses](#).

2.4.3 Installation (Linux)

Altova LicenseServer can be installed on Linux systems (Debian, Ubuntu, RedHat, CentOS).

Uninstalling old versions of LicenseServer

On the Linux command line interface (CLI), you can check whether LicenseServer is installed with the following command:

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

If LicenseServer is not installed, go ahead with the installation as documented in the next steps. If LicenseServer is installed and you wish to install a newer version of it, uninstall the old version with the command:

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

Installing Altova LicenseServer

On Linux systems, LicenseServer must be installed independently of other Altova server products. It is not included as part of the installation packages of Altova server products. Download Altova LicenseServer from the [Altova website](#) and copy the package to any directory on the Linux system.

Distribution	Installer extension
Debian	.deb
Ubuntu	.deb
CentOS	.rpm

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 LicenseServer with the following command:

```
[Debian]:          sudo dpkg --install licenseserver-1.1-debian.deb
[Ubuntu]:          sudo dpkg --install licenseserver-1.1-ubuntu.deb
[CentOS]:          sudo rpm -ivh licenseserver-1.1-1.x86_64.rpm
```

The LicenseServer package will be installed in:

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

For information about how to proceed with assigning licenses, see the section [How to Assign Licenses](#).

2.4.4 Installation (Mac OS X)

Altova LicenseServer can be installed on Mac OS X systems (version 10.7 or higher). Since you might need to uninstall a previous version, uninstalling is described first.

Uninstalling old versions of LicenseServer

Before uninstalling LicenseServer, stop the service 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 terminal and make sure that LicenseServer is not in the list.

In the Applications terminal, right-click the LicenseServer 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/LicenseServer
```

Installing Altova LicenseServer

Download Altova LicenseServer from the [Altova website](#) (the installer file has a `.pkg` file extension), and double-click the installer package to start the installation. Follow the on-screen instructions. You will need to accept the license agreement for installation to proceed.

The LicenseServer package will be installed in the folder:

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

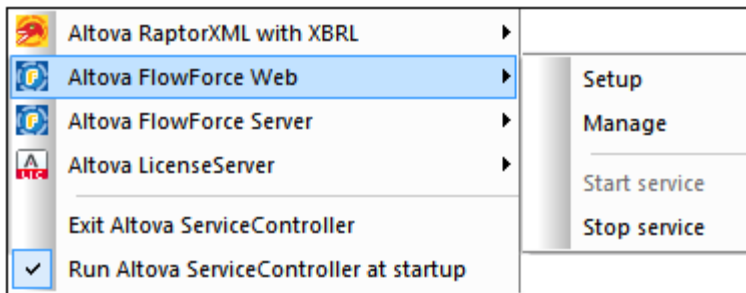
2.4.5 Altova ServiceController

The Altova ServiceController (hereafter also called ServiceController) is an application for conveniently starting, stopping and configuring Altova services **on Windows systems**. It is not available on Linux systems.

Altova ServiceController is installed with Altova LicenseServer, and can be started by clicking its command in the Altova LicenseServer folder of the **Start** menu. After the ServiceController has been started, it can be accessed via the system tray (*screenshot below*).



To run the ServiceController after logging in to the system, click the ServiceController icon in the system tray to pop up 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 pops up (*see screenshot below*), click **Exit Altova ServiceController**.



Starting and stopping Altova services

Each installed Altova service component will have an entry in the ServiceController menu (see *screenshot above*). An Altova service can be started or stopped via a command in its ServiceController sub-menu. Additionally, important administration tasks of individual services can be accessed via the ServiceController menu. In the screenshot above, for example, the Altova FlowForce Web service has a sub-menu in which you can choose to access its Setup page.

2.4.6 How to Assign Licenses

To assign an Altova server product a license using Altova LicenseServer, you need to do the following:

1. [Start LicenseServer](#).
2. Open the [LicenseServer Configuration page](#), which is the administrator's interface with LicenseServer, on [Windows](#) or [Linux](#).
3. [Upload the license/s](#) you have received from Altova to the license pool of your Altova LicenseServer. Do this in the [License Pool](#) tab of the LicenseServer Configuration page.
4. Register the Altova server product ([FlowForce Server](#), [MapForce Server](#), [StyleVision Server](#)) with LicenseServer. Depending on the product's type, the method of registering it with LicenseServer will be different: either via the product's GUI or its command line. See the documentation of your Altova server product for information about how to register it with LicenseServer.
5. In the [Server Management](#) tab of the [LicenseServer Configuration page](#), [assign a license](#) to the Altova server product according to the number of cores on the product machine.

Note on cores and licenses

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 on a particular server machine must be greater than or equal to the number of cores available on that server, whether it's a physical or virtual machine.

For example, if a server has eight cores (an octa-core processor), you must purchase at least an 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 an 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 license can be used for only one client machine at a time, even if it 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 the license cannot be used simultaneously for another client machine.

Start LicenseServer

This section:

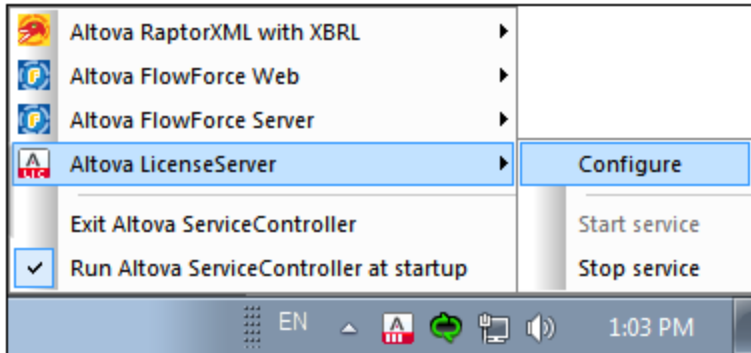
- How to start LicenseServer on [Windows systems](#)
- How to start LicenseServer on [Linux systems](#)
- How to start LicenseServer on [Mac OS X systems](#)
- Note about [Connection to altova.com](#)

Windows systems

You can start LicenseServer via the Altova ServiceController, which is available in the system

tray. (Click **Start | All Programs | Altova LicenseServer | Altova ServiceController** to start Altova ServiceController and display its icon in the system tray. If you select the *Run Altova ServiceController at Startup* option (see screenshot below), Altova ServiceController will start up on system start and its icon will be available in the system tray from then onwards.)

To start LicenseServer, click the Altova ServiceController icon in the system tray, mouse 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.



Linux systems

To start LicenseServer as a service on Linux systems, run the following command in a terminal window.

```
[Debian]:          sudo /etc/init.d/licenseserver start
[Ubuntu]:          sudo initctl start licenseserver
[CentOS]:          sudo initctl start licenseserver
```

(If you need to stop LicenseServer, replace **start** with **stop** in the above command)

Starting LicenseServer as a service

To start LicenseServer as a service on Mac OS X systems, run the following command in a terminal window:

```
sudo launchctl load /Library/LaunchDaemons/com.altova.LicenserServer11.plist
```

If at any time you need to stop LicenseServer, use:

```
sudo launchctl unload /Library/LaunchDaemons/com.altova.LicenserServer11.plist
```

Connection to the Master Licensing Server at altova.com

The Altova LicenseServer needs to be able to communicate with the Master Licensing Server at altova.com to validate and authenticate license-related data and to ensure continuous compliance with the Altova license agreements. If the Altova LicenseServer is

unable to connect with the `altova.com` Master Licensing Servers for a duration of more than 5 days (= 120 hours), then the Altova LicenseServer will no longer permit the usage of any Altova software products connected to the Altova LicenseServer.

Any such loss of connection with the `altova.com` master servers will be logged in the [Messages tab](#) of the [Configuration page of the Altova LicenseServer](#). In addition, the administrator can configure the Altova LicenseServer to automatically send an alert email when the connection to `altova.com` is lost. Alert Mail settings are available in the [Settings tab](#) of the [Configuration page](#).

Open LicenseServer Config Page (Windows)

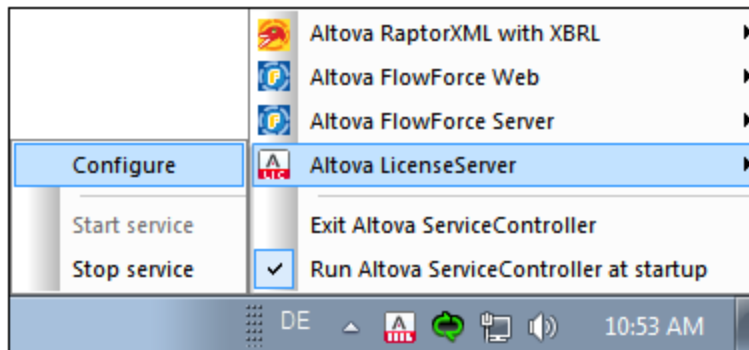
This section:

- [Opening the Configuration page if LicenseServer is on the same machine](#)
- [Opening the Configuration page if LicenseServer is on another machine](#)
- [Logging in with the initial password](#)
- [Setting a fixed port for the Configuration page](#)

Opening the Configuration page if LicenseServer is on the same machine

On Windows systems, if LicenseServer is on the same machine, you can open the [Configuration page](#) of LicenseServer in one of two ways:

- Click **Start | All Programs | Altova LicenseServer | LicenseServer Configuration Page**. The Configuration page opens in a new tab of your Internet browser.
- Click the Altova ServiceController icon in the system tray, mouse over **Altova LicenseServer** in the menu that pops up (see *screenshot below*), and then select **Configure** from the LicenseServer submenu.



The [Configuration page](#) opens in a new browser window, and its login mask is displayed (*screenshot below*).

Opening the Configuration page if LicenseServer is on another machine

To open the LicenseServer [Configuration page](#) from some other Windows machine on the local network (than that on which LicenseServer is installed), enter the URL of the LicenseServer [Configuration page](#) in the address bar of a browser and press **Enter**. By default, the URL of the Configuration page will be:

```
http://<serverIPAddressOrName>:8088/
```

The URL is present in the HTML code of the Configuration page itself, which is named `webui.html` and is located at:

```
C:/ProgramData/Altova/LicenseServer/WebUI.html
```

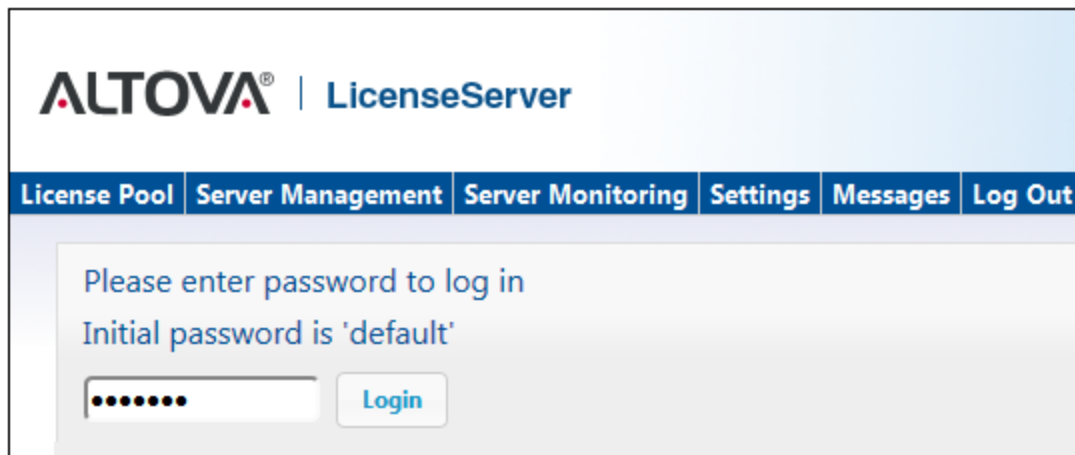
If you have [set the URL of the Configuration page](#) to be generated dynamically (in the Settings tab of the Configuration page), then a new URL is generated each time LicenseServer is started. You will need to check the current version of `webui.html` to find out the current URL of the [Configuration page](#).

The dynamically generated URL in `webui.html` will have a form something like:

`http://127.0.0.1:55541/optionally-an-additional-string`, and it is located in the function `checkIfServiceRunning()` in a script near the end of the `<head>` element. While the port number in the URL is dynamically assigned, the IP address part identifies the server on which LicenseServer has been installed. If you wish to access the LicenseServer [Configuration page](#) from another machine, make sure that the IP address part of the URL has the correct IP address or name of the server on which LicenseServer has been installed. For example, the URL could be something like: `http://MyServer:55541`.

Logging in with the initial password

After going through the steps above, the [Configuration page](#) is opened with the login mask displayed (*screenshot below*). You can log in with the initial password of `default`. After you have logged in, you can change your password in the [Settings](#) tab.



Setting a fixed or dynamic port for the Configuration page

The port of the Configuration page (and consequently its address) can be specified in the [Settings page](#). By default the port is 8088. You can set any other port you want for the LicenseServer [Configuration page](#) (see *screenshot below*). Alternatively, you allow the port to be selected dynamically each time LicenseServer starts up. In this case, you will need to find out the URL of the Configuration page from the file `webui.html` (see [Open LicenseServer Config Page \(Windows\)](#) and [Open LicenseServer Config Page \(Linux\)](#)).

Web UI

Configure the host addresses where the web UI is available to administrators.

☒ All interfaces and assigned IP addresses

☐ Local only (localhost)

☐ Only the following hostname or IP address:

Ensure this hostname or IP address exists or LicenseServer will fail to start!

Configure the port used for the web UI.

☐ Dynamically chosen by the operating system

☒ Fixed port

Ensure this port is available or LicenseServer will fail to start!

The advantage of a fixed port is that the page URL is known in advance and therefore can be accessed easily. If the port is assigned dynamically, the port part of the URL will have to be looked up in the file `webUI.html` each time LicenseServer is started afresh.

Open LicenseServer Config Page (Linux)

This section:

- [Opening the Configuration page for the first time with the returned URL](#)
- [URL of the LicenseServer Configuration page](#)
- [Logging in with the initial password](#)
- [Setting a fixed port for the Configuration page](#)

Opening the Configuration page for the first time with the returned URL

On Linux systems, when you register your Altova server product with LicenseServer via the CLI, the URL of the LicenseServer Configuration page is returned. On opening this URL in a browser, you are prompted to read and accept the license agreement. After accepting the license agreement, the Configuration page's login mask is displayed (*screenshot below*).

URL of the LicenseServer Configuration page

To open the LicenseServer [Configuration page](#) at any time, enter its URL in the address bar of a browser and press **Enter**. By default, the URL of the Configuration page will be:

```
http://<serverIPAddressOrName>:8088/
```

The URL is present in the HTML code of the Configuration page itself, which is named `webUI.html` and is located at:

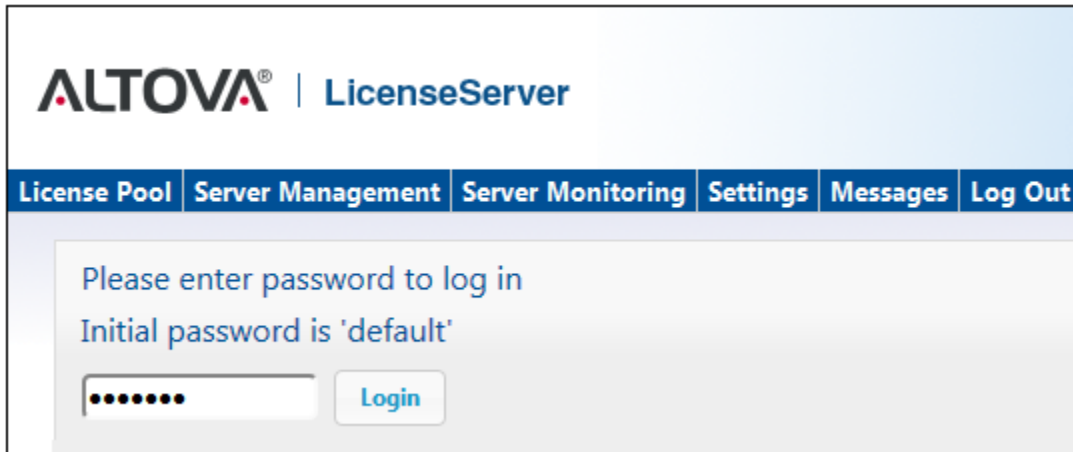
```
/var/opt/Altova/LicenseServer/webUI.html
```

If you have [set the URL of the Configuration page](#) to be generated dynamically (in the Settings tab of the Configuration page), then a new URL is generated each time LicenseServer is started. You will need to check the current version of `webUI.html` to find out the current URL of the [Configuration page](#).

The dynamically generated URL in `webUI.html` will have a form something like: `http://127.0.0.1:55541`, and it is located in the function `checkIfServiceRunning()` in a script near the end of the `<head>` element. While the port number in the URL is dynamically assigned, the IP address part identifies the server on which LicenseServer has been installed. If you wish to access the LicenseServer [Configuration page](#) from another machine, make sure that the IP address part of the URL has the correct IP address or name of the server on which LicenseServer has been installed. For example, the URL could be something like: `http://MyServer:55541`.

Logging in with the initial password

After going through the steps above, the [Configuration page](#) is opened with the login mask displayed (*screenshot below*). You can log in with the initial password of `default`. After you have logged in, you can change your password in the [Settings](#) tab.



Setting a fixed or dynamic port for the Configuration page

The port of the Configuration page (and consequently its address) can be specified in the [Settings page](#). By default the port is `8088`. You can set any other port you want for the LicenseServer [Configuration page](#) (see *screenshot below*). Alternatively, you allow the port to be selected dynamically each time LicenseServer starts up. In this case, you will need to find out the URL of the Configuration page from the file `WebUI.html` (see [Open LicenseServer Config Page \(Windows\)](#) and [Open LicenseServer Config Page \(Linux\)](#)).

Web UI

Configure the host addresses where the web UI is available to administrators.

☒ All interfaces and assigned IP addresses

☐ Local only (localhost)

☐ Only the following hostname or IP address:

Ensure this hostname or IP address exists or LicenseServer will fail to start!

Configure the port used for the web UI.

☐ Dynamically chosen by the operating system

☒ Fixed port

Ensure this port is available or LicenseServer will fail to start!

The advantage of a fixed port is that the page URL is known in advance and therefore can be accessed easily. If the port is assigned dynamically, the port part of the URL will have to be looked up in the file `webUI.html` each time LicenseServer is started afresh.

Open LicenseServer Config Page (Mac OS X)

This section:

- [Opening the Configuration page for the first time with the returned URL](#)
- [URL of the LicenseServer Configuration page](#)
- [Logging in with the initial password](#)
- [Setting a fixed port for the Configuration page](#)

Opening the Configuration page for the first time with the returned URL

On Mac OS X systems, when you register your Altova server product with LicenseServer via the CLI, the URL of the LicenseServer Configuration page is returned. On opening this URL in a browser, you are prompted to read and accept the license agreement. After accepting the license agreement, the Configuration page's login mask is displayed (*screenshot below*).

URL of the LicenseServer Configuration page

To open the LicenseServer [Configuration page](#) at any time, enter its URL in the address bar of a browser and press **Enter**. By default, the URL of the Configuration page will be:

```
http://<serverIPAddressOrName>:8088/
```

The URL is present in the HTML code of the Configuration page itself, which is named `webUI.html` and is located at:

```
/var/Altova/LicenseServer/webUI.html
```

If you have [set the URL of the Configuration page](#) to be generated dynamically (in the Settings tab of the Configuration page), then a new URL is generated each time LicenseServer is started. You will need to check the current version of `webUI.html` to find out the current URL of the [Configuration page](#).

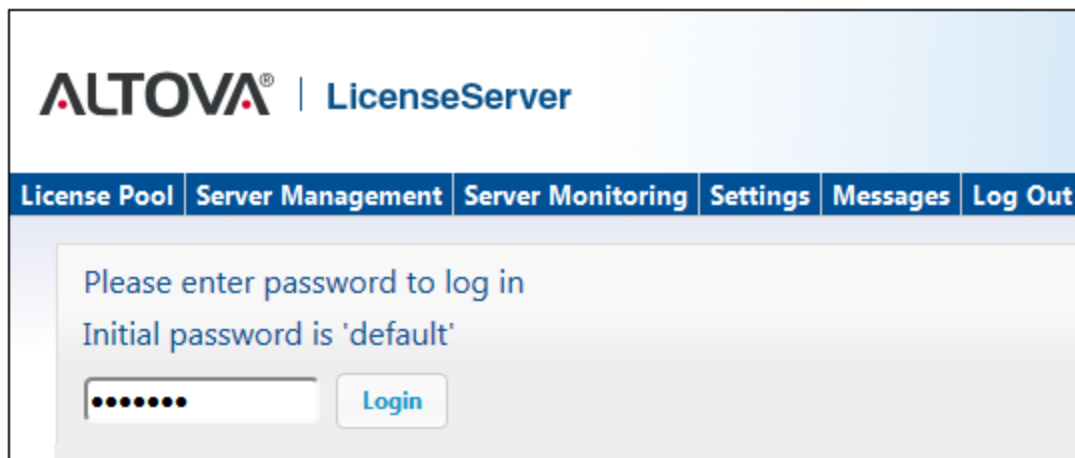
The dynamically generated URL in `webUI.html` will have a form something like:

`http://127.0.0.1:55541`, and it is located in the function `checkIfServiceRunning()` in a script near the end of the `<head>` element. While the port number in the URL is dynamically assigned, the IP address part identifies the server on which LicenseServer has been installed. If you wish to access the LicenseServer [Configuration page](#) from another machine, make sure that the IP address part of the URL has the correct IP address or name of the server on which LicenseServer has been installed. For example, the URL could be something like: `http://MyServer:55541`.

Note: The [Configuration page](#) can also be accessed directly via the **Finder | Applications | Altova License Server** icon.

Logging in with the initial password

After going through the steps above, the [Configuration page](#) is opened with the login mask displayed (*screenshot below*). You can log in with the initial password of `default`. After you have logged in, you can change your password in the [Settings](#) tab.



Setting a fixed or dynamic port for the Configuration page

The port of the Configuration page (and consequently its address) can be specified in the [Settings page](#). By default the port is `8088`. You can set any other port you want for the LicenseServer [Configuration page](#) (see *screenshot below*). Alternatively, you allow the port to be selected dynamically each time LicenseServer starts up. In this case, you will need to find out the URL of the Configuration page from the file `WebUI.html` (see [Open LicenseServer Config Page \(Windows\)](#) and [Open LicenseServer Config Page \(Linux\)](#)).

Web UI

Configure the host addresses where the web UI is available to administrators.

☒ All interfaces and assigned IP addresses

☐ Local only (localhost)

☐ Only the following hostname or IP address:

Ensure this hostname or IP address exists or LicenseServer will fail to start!

Configure the port used for the web UI.

☐ Dynamically chosen by the operating system

☒ Fixed port

Ensure this port is available or LicenseServer will fail to start!

The advantage of a fixed port is that the page URL is known in advance and therefore can be accessed easily. If the port is assigned dynamically, the port part of the URL will have to be looked up in the file `WebUI.html` each time LicenseServer is started afresh.

Upload Licenses to LicenseServer

This section:

- [Uploading a license file to the license pool of LicenseServer](#)
- [License status](#)
- [Activating the licenses you wish to use](#)
- [Next steps](#)

Uploading a license file to the license pool of LicenseServer

After you have obtained a license file from Altova, you must upload it to the Altova LicenseServer. (How to do this is described below.) Each license file can contain one or more licenses and depends on your purchase. When you upload a license file, all the licenses in it will be uploaded to the server and can be assigned to an Altova product that has been registered with that LicenseServer. All the uploaded licenses, from one or more license files and for all Altova products, are collected in a license pool on the LicenseServer. The license pool is displayed in the License Pool tab of the LicenseServer Configuration page (*screenshot below*).

License files are uploaded to the LicenseServer using the Upload function of the License Pool tab (*see screenshot below*).

Click the **Browse** button and select the license file you want. The license file will appear in the Upload License File text field and the **Upload** button will be enabled. Click the **Upload** button to upload the license file. All the licenses in the file are uploaded and displayed in the License Pool tab. The screenshot below shows multiple licenses, uploaded from multiple license files.

	Status	Name	Company	Product	Edition	Version	Key	Expires in day	SMP days left	CPU Cores	Running
<input type="checkbox"/>	Active	Mr. Nobody	Altova GmbH	Altova FlowForce Server		2013	CAWYXW8-	334	334	1	0
<input checked="" type="checkbox"/>	Active	Mr. Nobody	Altova GmbH	Altova FlowForce Server		2013	7CMJT18-	334	334	2	0
<input type="checkbox"/>	Active	Mr. Nobody	Altova GmbH	Altova MapForce Server		2013	MM5UC1U-	334	334	1	0
<input type="checkbox"/>	Active	Mr. Nobody	Altova GmbH	Altova RaptorXML+XBRL		2013	HC139LF-	334	334	1	0
<input type="checkbox"/>	Active	Mr. Nobody	Altova GmbH	Altova StyleVision Server		2013	3D78278-	334	334	1	0
<input type="checkbox"/>	Inactive	Mr. Nobody	Altova GmbH	Altova FlowForce Server		2013	966PPHM-	334	334	3	0
<input type="checkbox"/>	Inactive	Mr. Nobody	Altova GmbH	Altova StyleVision Server		2013	DA5T2WU-	334	334	4	0

License status

License status values are as follows:

- Activating:** When a license is uploaded into the license pool of LicenseServer, the server will transmit license-related data to the `altova.com` master licensing server to validate, authenticate, and activate the license that was supplied. This is necessary to

ensure compliance with the Altova license agreements. During this initial activation and authentication transaction—which typically lasts between 30 seconds and a couple of minutes, depending on your Internet connection, speed, and overall network traffic—the status of the license will be indicated as *Activating...*

- *Failed Verification*: If a connection with the `altova.com` master licensing server cannot be made, then the status of the license in the pool will be shown as *Failed Verification*. If this happens, check your Internet connection and firewall rules to ensure that LicenseServer is able to communicate with the `altova.com` master licensing server.
- *Active*: Once the license has been authenticated and activated, the status in the pool will change to *Active*.
- *Inactive*: If a license has been verified, but is present on another LicenseServer on the network, the status in the pool will be shown as *Inactive*. An *Inactive* status also results when a license is manually deactivated in the license pool by the administrator.
- *Blocked*: A license is shown in the license pool as *Blocked* if there was a problem authenticating the license and the `altova.com` master licensing server has not granted permission to the LicenseServer to use this license. This could be the result of a license agreement violation, over-usage of a license, or other compliance issues. Should you see a license showing up as *Blocked*, please contact Altova Support with your license information and any other relevant data.

These statuses are summarized in the table below:

Status	Meaning
<i>Activating...</i>	On upload, license information is sent to <code>altova.com</code> for verification. Refresh the browser to view the updated status. Verification and activation can take a few minutes.
<i>Failed Verification</i>	A connection to <code>altova.com</code> could not be made. After establishing a connection, either restart the service or activate the license (with the Activate button).
<i>Active</i>	Verification was successful, the license is active.
<i>Inactive</i>	Verification was successful, but the license is on another LicenseServer on the network. Licenses can be made inactive with the Deactivate button.
<i>Blocked</i>	Verification was not successful. License is invalid and is blocked. Contact Altova Support .

Note: After a license has been sent to `altova.com` for verification, the browser must be refreshed to see the updated status. Verification and activation can take a few minutes.

Note: If a connection to `altova.com` could not be made, the status will be *Failed Verification*. After establishing a connection, either restart the service or try activating the license with the **Activate** button.

Note: When a license is given a status of *Inactive* or *Blocked*, a message explaining the status is also added to the Messages log.

Only an active license can be assigned to a product installation. An inactive license can be activated or deleted from the license pool. If a license is deleted from the license pool, it can be

uploaded again to the pool by uploading the license file containing it. When a license file is updated, only those licenses in it that are not already in the pool will be uploaded to the pool. To activate, deactivate, or delete a license, select it and then click the **Activate**, **Deactivate**, or **Delete** button, respectively.

Activate the license/s you wish to use

Before you can assign a license to an Altova product, it must be active. So do ensure it is active. If it is inactive, select it and click **Activate**.

Next Steps

After you have uploaded the license file to the LicenseServer and checked that the license you want is active, do the following:

1. Register the Altova server product ([FlowForce Server](#), [MapForce Server](#), [StyleVision Server](#)) with LicenseServer. (If you have already done this prior to uploading the license file, you can now start assigning licenses.)
2. [Assign a license](#) to your Altova product that has been registered with the LicenseServer.

Register FlowForce Server with LicenseServer

This section:

- [Methods of registering FlowForce Server with LicenseServer](#)
 - [Accessing the FlowForce Server Setup page \(Windows\)](#)
 - [Accessing the FlowForce Server Setup page \(Linux\)](#)
 - [Registering FlowForce Server via the Setup page \(Windows and Linux\)](#)
 - [Registering FlowForce Server via the FlowForce CLI \(Windows\)](#)
 - [Registering FlowForce Server via the FlowForce CLI \(Linux\)](#)
 - [Next steps](#)
-

Methods of registering FlowForce Server

FlowForce Server can be registered with LicenseServer using any of the following methods:

- [Via the FlowForce Server Setup page \(Windows and Linux\)](#)
- [Via the FlowForce CLI \(Windows\)](#)
- [Via the FlowForce CLI \(Linux\)](#)

Accessing the FlowForce Server Setup page (Windows)

The FlowForce Server Setup page can be accessed in one of the following ways:

- Via the **Start** menu:
Start | Altova FlowForce Server 2013 | FlowForce Server Setup Page
- Via [Altova ServiceController](#): Click the ServiceController icon in the system tray. In the

menu that pops up, select *Altova FlowForce Web | Setup*.

This pops up the FlowForce Server Setup page (*screenshot above*).

Accessing the FlowForce Server Setup page (Linux)

After you have installed FlowForce Server on Linux (see the FlowForce Server user documentation for information about how to do this), start FlowForce Web Server as a service with the following command:

```
sudo /etc/init.d/flowforcewebserver start
```

A message containing the URL of the FlowForce Server Setup appears in the terminal window:

```
FlowForceWeb running on http://127.0.1.1:3459/setup?key=52239315203
```

Enter the URL into the address field of a browser and hit Enter to access FlowForce Server Setup page (*screenshot above*).

Registering FlowForce Server via the Setup page (Windows and Linux)



In the Setup page (*screenshot below*)—how to access it is described above—the LicenseServer field specifies the Altova LicenseServer to be used for registration.

ALTOVA®
flowforce®
SERVER 2013

Home Help

Setup

LicenseServer

Enter address here or search for LicenseServer  

[Register with LicenseServer](#)

FlowForce Web Server

Bind address: All interfaces (0.0.0.0) ▼ 127.0.0.1 Port: 8082

Default time zone: Europe/Berlin ▼

FlowForce Server



Bind address: All interfaces (0.0.0.0) ▼ 127.0.0.1 Port: 4646

[Apply settings and restart FlowForce services](#)

The LicenseServer can be specified in one of two ways.

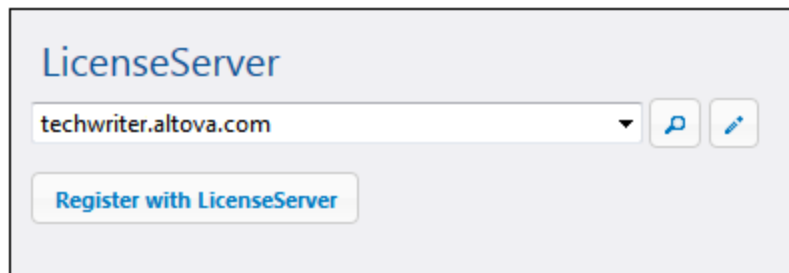
- You can search for Altova LicenseServers that are currently available on the network—that is, those that are currently running. Do this by clicking the **Search for Altova LicenseServers** button (*highlighted yellow in the screenshot below*).

LicenseServer

Enter address here or search for LicenseServer  

[Register with LicenseServer](#)

The search returns a list of available Altova LicenseServers on the network. One LicenseServer will be selected (*screenshot below*) and the others will be available in the dropdown list of the combo box. Select the LicenseServer on which your FlowForce license is stored.



LicenseServer

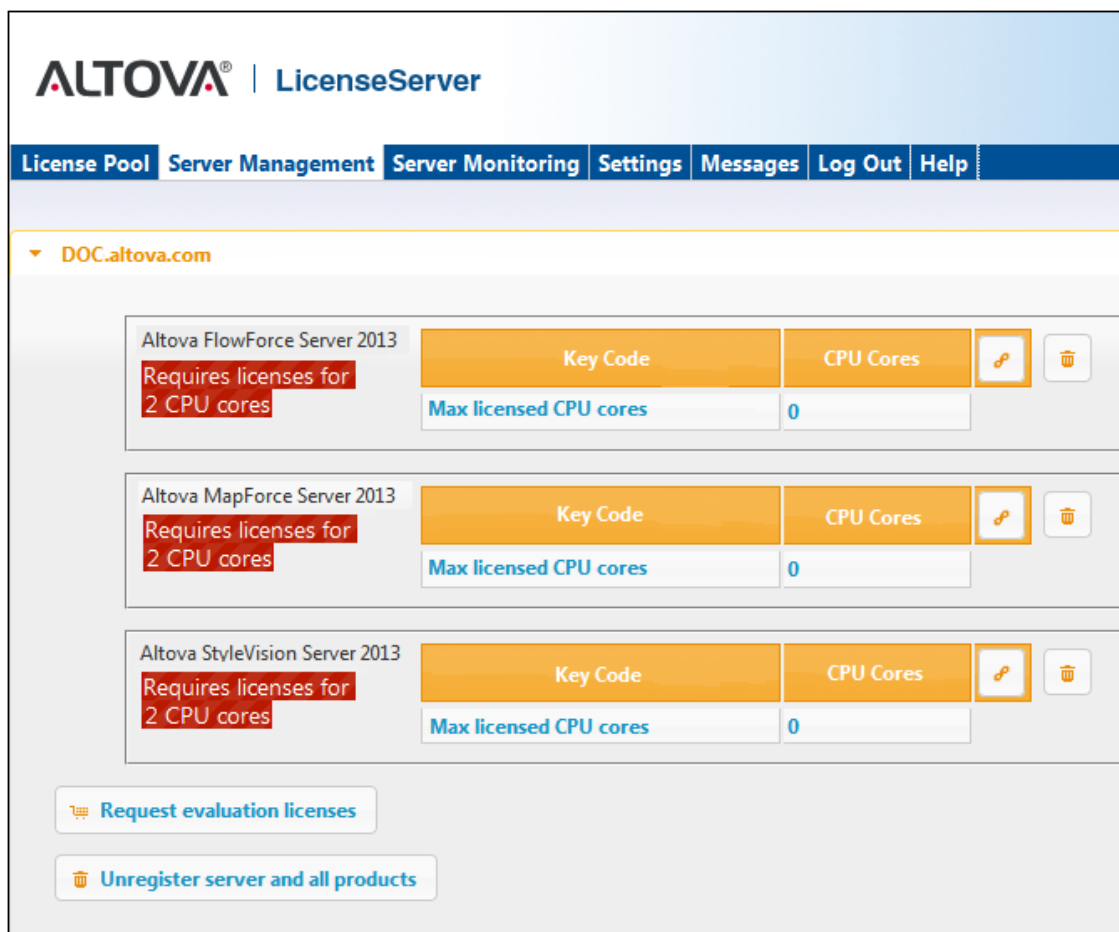
techwriter.altova.com

Register with LicenseServer

- Alternatively, you can enter the address of the LicenseServer in the LicenseServer field. If the currently running LicenseServers are available as a dropdown list, you must click the **Manually Enter Address** button to be able to enter an address in the LicenseServer field.

After you have specified the LicenseServer, click **Register with LicenseServer**. The Altova server application will be registered with the specified LicenseServer, and that LicenseServer's [Configuration page](#) will open in a browser with its Server Management tab active (*screenshot below*).

Note: You may need to allow pop-ups in order for the LicenseServer Configuration page to be displayed.



ALTOVA® | LicenseServer

License Pool Server Management Server Monitoring Settings Messages Log Out Help

▼ DOC.altova.com

	Key Code	CPU Cores		
Altova FlowForce Server 2013 Requires licenses for 2 CPU cores	Max licensed CPU cores	0		
Altova MapForce Server 2013 Requires licenses for 2 CPU cores	Max licensed CPU cores	0		
Altova StyleVision Server 2013 Requires licenses for 2 CPU cores	Max licensed CPU cores	0		

Request evaluation licenses

Unregister server and all products

In the screenshot below, three Altova products have been registered with the Altova LicenseServer at `DOC.altova.com`. How to assign licenses is described in the next section, [Assign Licenses to Registered Products](#).

Registering FlowForce Server via the FlowForce CLI (Windows)

On Windows machines, FlowForce Server can also be registered with an Altova LicenseServer on your network via the command line (CLI) by using the `licenseserver` command:

```
FlowForceServer licenseserver Server-Or-IP-Address
```

For more details of CLI commands, see the FlowForce CLI documentation (either in the CLI window itself, delivered with the installed product package, or on the [Altova website](#)).

If FlowForce Server was installed with other Altova server products as sub-packages, registering FlowForce Server will automatically also register the Altova server products. After successfully registering FlowForce Server, you can go to LicenseServer and assign a license to FlowForce Server. How to do this is described in the section [Assign Licenses to Registered Products](#).

Registering FlowForce Server via the FlowForce CLI (Linux)

On Linux machines, FlowForce Server can be registered with LicenseServer by using the `licenseserver` command of the FlowForce Server CLI. Note that FlowForce Server must be started with root rights.

```
sudo /opt/Altova/FlowForceServer2013/bin/flowforceserver 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 FlowForce Server executable is:

```
/opt/Altova/MapForceServer2013/bin
```

After successfully registering FlowForce Server, you can go to LicenseServer and assign a license to FlowForce Server. How to do this is described in the section [Assign Licenses to Registered Products](#).

Next Steps

After you have registered your Altova product with LicenseServer, do the following:

1. If you have not already uploaded your license file/s to the LicenseServer (see previous section, [Upload Licenses to LicenseServer](#)), upload the license file now and check that the license you want is active. If you have already done this, carry on to the next step, [Assign Licenses](#).
2. [Assign a license](#) to your Altova product that has been registered with the LicenseServer.

Register MapForce Server with LicenseServer

This section:

- [Registering MapForce Server from FlowForce Server \(Windows\)](#)
 - [Registering a standalone MapForce Server \(Windows\)](#)
 - [Registering MapForce Server \(Linux\)](#)
 - [Next steps](#)
-

MapForce Server can be installed as part of the FlowForce Server package or as a standalone server product. In either case, it must be registered with Altova LicenseServer. Only after it has been registered with LicenseServer can a [license be assigned](#) to it from LicenseServer. On Windows systems, if MapForce Server was installed as part of the FlowForce Server package, it will automatically be registered when FlowForce is registered. On Linux systems, only if MapForce Server is installed after FlowForce Server will it be registered automatically when FlowForce Server is registered subsequently.

Registering MapForce Server from FlowForce Server (Windows)

MapForce Server is packaged with FlowForce Server, so when FlowForce Server is registered with an Altova LicenseServer on your network, MapForce Server will automatically also be registered with LicenseServer. How to register FlowForce Server is described in the FlowForce Server documentation and in the section, [Register FlowForce Server with LicenseServer](#).

After the registration, you can go to LicenseServer and assign a MapForce Server license to MapForce Server. How to do this is described in the section, [Assign Licenses to Registered Products](#).

Registering a standalone MapForce Server (Windows)

If you have installed MapForce Server as a standalone package, you must register it with an Altova LicenseServer on your network and then license it from the Altova LicenseServer. You can register MapForce Server via its command line interface (CLI) by using the `licenseserver` command:

```
MapForceServer licenseserver Server-Or-IP-Address
```

For more details of CLI commands, see the MapForce Server CLI documentation (either in the CLI window itself, delivered with the installed product package, or on the [Altova website](#)).

After successfully registering MapForce Server, you can go to LicenseServer and assign a license to MapForce Server. How to do this is described in the section, [Assign Licenses to Registered Products](#).

Registering MapForce Server (Linux)

On Linux machines, MapForce Server can be registered with LicenseServer by using the `licenseserver` command of the MapForce Server CLI. Note that MapForce Server must be started with root rights.

```
sudo /opt/Altova/MapForceServer2013/bin/mapforceserver 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 MapForce Server executable is:

```
/opt/Altova/MapForceServer2013/bin
```

After successfully registering MapForce Server, you can go to LicenseServer and assign a license to MapForce Server. How to do this is described in the section [Assign Licenses to Registered Products](#).

Next Steps

After you have registered your Altova product with LicenseServer, do the following:

1. If you have not already uploaded your license file/s to the LicenseServer (see previous section, [Upload Licenses to LicenseServer](#)), upload the license file now and check that the license you want is active. If you have already done this, carry on to the next step, [Assign Licenses](#).
2. [Assign a license](#) to your Altova product that has been registered with the LicenseServer.

Register RaptorXML(+XBRL) Server with LicenseServer

This section:

- [Registering RaptorXML\(+XBRL\) Server \(Windows\)](#)
 - [Registering RaptorXML\(+XBRL\) Server \(Linux\)](#)
 - [Next steps](#)
-

RaptorXML(+XBRL) Server must be installed on the server machine and then be started as a service. It must then be registered with LicenseServer. Only after it has been registered with LicenseServer can a [license be assigned](#) to it from LicenseServer. This section describes how to register RaptorXML(+XBRL) Server with LicenseServer.

Registering RaptorXML(+XBRL) Server (Windows)

You can register RaptorXML(+XBRL) Server via its command line interface (CLI) by using the `licenseserver` command:

```
RaptorXML licenseserver Server-Or-IP-Address
```

For more details of CLI commands, see the RaptorXML(+XBRL) Server CLI documentation (either in the CLI window itself, delivered with the installed product package, or on the [Altova website](#)).

After successfully registering RaptorXML(+XBRL) Server, you can go to LicenseServer and assign a license to RaptorXML(+XBRL) Server. How to do this is described in the section [Assign Licenses to Registered Products](#).

Registering RaptorXML(+XBRL) Server (Linux)

On Linux machines, RaptorXML(+XBRL) Server can be registered with LicenseServer by using the `licenseserver` command of the RaptorXML(+XBRL) Server CLI. Note that RaptorXML(+XBRL) Server must be started with root rights.

```
sudo /opt/Altova/RaptorXMLServer2013/bin/raptorxmlserver 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 RaptorXML(+XBRL) Server executable is:

```
/opt/Altova/RaptorXMLServer2013/bin
```

After successfully registering RaptorXML(+XBRL) Server, you can go to LicenseServer and assign a license to RaptorXML(+XBRL) Server. How to do this is described in the section [Assign Licenses to Registered Products](#).

Next Steps

After you have registered your Altova product with LicenseServer, do the following:

1. If you have not already uploaded your license file/s to the LicenseServer (see previous section, [Upload Licenses to LicenseServer](#)), upload the license file now and check that the license you want is active. If you have already done this, carry on to the next step, [Assign Licenses](#).
2. [Assign a license](#) to your Altova product that has been registered with the LicenseServer.

Register StyleVision Server with LicenseServer

This section:

- [Registering StyleVision Server from FlowForce Server \(Windows\)](#)
- [Registering a standalone StyleVision Server \(Windows\)](#)
- [Registering StyleVision Server \(Linux\)](#)
- [Next steps](#)

StyleVision Server can be installed as part of the FlowForce Server package or as a standalone server product. In either case, it must be registered with Altova LicenseServer. Only after it has been registered with LicenseServer can a [license be assigned](#) to it from LicenseServer. On Windows systems, if StyleVision Server was installed as part of the FlowForce Server package, it will automatically be registered when FlowForce is registered. On Linux systems, only if StyleVision Server is installed after FlowForce Server will it be registered automatically when FlowForce Server is registered subsequently.

Registering StyleVision Server from FlowForce (Windows)

StyleVision Server is packaged with FlowForce Server, so when FlowForce Server is registered with an Altova LicenseServer on your network, StyleVision Server will automatically also be registered with LicenseServer. How to register FlowForce Server is described in the FlowForce Server documentation and in the section, [Register FlowForce Server with LicenseServer](#).

After the registration, you can go to LicenseServer and assign a StyleVision Server license to StyleVision Server. How to do this is described in the section [Assign Licenses to Registered](#)

[Products.](#)

Registering a standalone StyleVision Server (Windows)

If you have installed StyleVision Server as a standalone package on Windows, you must register it with an Altova LicenseServer on your network and then license it from the Altova LicenseServer. You can register StyleVision Server via its command line interface (CLI) by using the `licenseserver` command:

```
StyleVisionServer licenseserver Server-Or-IP-Address
```

For more details of CLI commands, see the StyleVision Server CLI documentation (either in the CLI window itself, delivered with the installed product package, or on the [Altova website](#)).

After successfully registering StyleVision Server, you can go to LicenseServer and assign a license to StyleVision Server. How to do this is described in the section [Assign Licenses to Registered Products](#).

Registering StyleVision Server (Linux)

On Linux machines, StyleVision Server can be registered with LicenseServer by using the `licenseserver` command of the StyleVision Server CLI. Note that StyleVision Server must be started with root rights.

```
sudo /opt/Altova/StyleVisionServer2013/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/StyleVisionServer2013/bin
```

After successfully registering StyleVision Server, you can go to LicenseServer and assign a license to StyleVision Server. How to do this is described in the section [Assign Licenses to Registered Products](#).

Next Steps

After you have registered your Altova product with LicenseServer, do the following:

1. If you have not already uploaded your license file/s to the LicenseServer (see previous section, [Upload Licenses to LicenseServer](#)), upload the license file now and check that the license you want is active. If you have already done this, carry on to the next step, [Assign Licenses](#).
2. [Assign a license](#) to your Altova product that has been registered with the LicenseServer.

Assign Licenses to Registered Products

This section:

- [Before assigning a license](#)
- [The Server Management tab](#)

- [Icons in the Server Management tab](#)
- [Note on cores and licenses](#)
- [Assigning a license](#)
- [Unregistering products from LicenseServer](#)

Before assigning a license

Before you assign a license to an Altova product, make sure that:

- The relevant license has been uploaded to the [license pool of LicenseServer](#) and that the license is active.
- Your Altova product has been registered with LicenseServer.

The Server Management tab

Licenses are assigned in the Server Management tab of the LicenseServer Configuration page (*screenshot below*). The screenshot shows that three Altova products have been registered with LicenseServer. (Since MapForce Server and StyleVision Server are bundled with FlowForce Server, registering FlowForce Server with LicenseServer automatically also registers MapForce Server and StyleVision Server. No additional registration of the latter two products are required if FlowForce Server is registered.)

The screenshot displays the 'Server Management' tab in the Altova LicenseServer interface. At the top, there is a navigation bar with tabs: 'License Pool', 'Server Management' (selected), 'Server Monitoring', 'Settings', 'Messages', 'Log Out', and 'Help'. Below the navigation bar, the user 'DOC.altova.com' is logged in. The main content area lists three registered products:

Product Name	Key Code	CPU Cores	Actions
Altova FlowForce Server 2013 Requires licenses for 2 CPU cores	Max licensed CPU cores	0	Assign License, Unregister
Altova MapForce Server 2013 Requires licenses for 2 CPU cores	Max licensed CPU cores	0	Assign License, Unregister
Altova StyleVision Server 2013 Requires licenses for 2 CPU cores	Max licensed CPU cores	0	Assign License, Unregister

At the bottom of the interface, there are two buttons: 'Request evaluation licenses' and 'Unregister server and all products'.

Note the following points about the Server Management tab:

- Each product is listed under the name of its client machine. In the screenshot above, one client machine, named `Doc.altova.com`, is listed. This client machine (`Doc.altova.com`) has three Altova products registered with the LicenseServer. If an Altova product on a different client machine is registered with this LicenseServer, then that client machine, with its registered products, will also be listed in the Server Management tab.
- Each registered Altova product on a client machine has its own *Key Code* entry, which takes the key code of a license. A registered product's key code is assigned by clicking its **Edit Assigned Licenses** button (see *icon list below*) and selecting the required license from those available for that product (for example, FlowForce Server) in the license pool. This procedure is explained in more detail below.
- Each product also has a line stating how many CPU cores need to be licensed to run that product on that client. If the number of licensed cores is less than the number required, then the information is marked in red (see *screenshot above*). (The number of CPU cores that need to be licensed is the number of CPU cores on that client and is obtained from the client machine by LicenseServer.)

Icons in the Server Management tab



Edit Assigned Licenses. Available with each product. Pops up the Manage Licenses dialog, in which new licenses can be assigned to the product and already assigned licenses can be edited.



Show Licenses. Appears with each license. Switches to the License Pool tab and highlights the selected license, so that license details can be read.



Unregister This Product. Available with each product. The selected product (on the selected client machine) will be unregistered from LicenseServer.

Note on cores and licenses

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 on a particular server machine must be greater than or equal to the number of cores available on that server, whether it's a physical or virtual machine.

For example, if a server has eight cores (an octa-core processor), you must purchase at least an 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 an 8-core license.

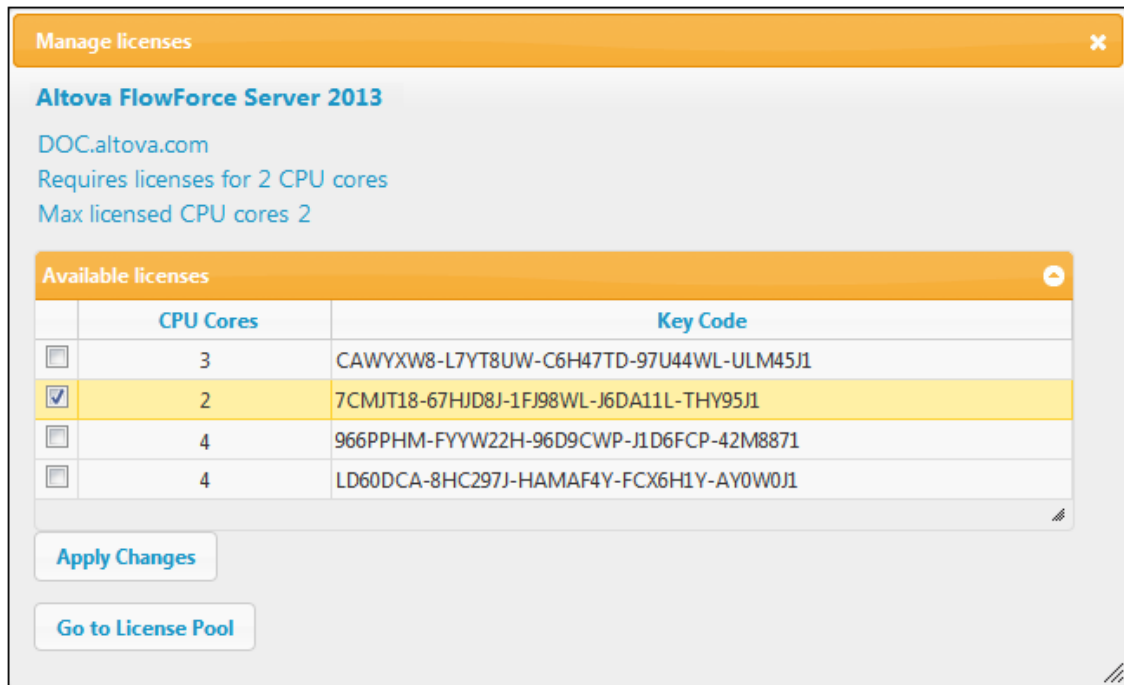
If you are using a computer server with a large number of CPU cores but only have a low volume to process, you may also create a virtual machine that is allocated a smaller number of cores, and purchase a license for that number. Such a deployment, of course, will have less processing speed than if all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

Note: Each license can be used for only one client machine at a time, even if it 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 the license cannot be used

simultaneously for another client machine.

Assigning a license

To assign a license to a registered product, click the **Edit Assigned Licenses** button of that product. This pops up the Manage Licenses dialog (*screenshot below*).






Note the following points about the licenses displayed in the Manage Licenses dialog:

- The product to be licensed is listed at the top left of the dialog. In the screenshot above the product is Altova FlowForce Server 2013.
- The dialog displays all the currently active licenses for that product in the license pool. In our screenshot, four currently active FlowForce Server licenses are in the license pool. LicenseServer will automatically detect from each license in the pool the product for which it has been issued.
- The licenses in the screenshot above have been licensed, respectively, for 3 CPU cores, 2 CPU cores, 4 CPU cores, and 4 CPU cores.
- You need to know the number of processor cores on the server on which the Altova server product has been installed. If the machine has a dual-core processor, you need a two-core (the CPU Cores count) license. This license could be, for example, the second license in the list shown in the screenshot above. You can also combine licenses. So, if the machine's processor is octa-core (eight-core), you can combine two 4-core licenses; for example, the third and fourth licenses in the list shown in the screenshot above.
- The Manage Licenses dialog will list only currently active licenses for that product. Licenses for other Altova products will not be listed.
- Licenses that have been assigned already—for example, to another installation of the product on the network—will have their check boxes checked. So only unchecked licenses may be selected.
- CPU cores indicates for how many CPU cores a license is valid.
- If you wish to make modifications to the license pool—for example, to upload, activate,

deactivate, or delete a license—click the **Go to License Pool** button.

Select the license you wish to assign. The license's check box will be checked. Also, the total number of CPU cores licensed for that product on that client is listed near the top left of the dialog as *Max licensed CPU cores* (see screenshot above). You can select more licenses if you wish to increase the number of licensed CPU cores for that product on that client. The *Max licensed CPU cores* in this case will be the sum of the CPU cores of all the selected licenses.

After selecting the license/s, click **Apply Changes**. The license/s will be assigned to that product and displayed in the Server Management tab (see screenshot below). The screenshot below shows that a 2-CPU-core license for Altova FlowForce Server has been assigned (to the client machine Doc.altova.com).

Altova FlowForce Server 2013 Requires licenses for 2 CPU cores	Key Code	CPU Cores		
	7CMJT18-67HJD8J-1FJ98WL-J6DA11L-THY95J1	2		
	Max licensed CPU cores	2		

Unregistering products

Each Altova product registered with LicenseServer is listed in the Server Management tab under its client machine name and has an **Unregister** icon to its right. Click this icon to unregister the product. If a license was assigned to the product, the assignment will be terminated when the product is unregistered. To unregister all products, click the **Unregister Server and All Products** button at the bottom of the Server Management tab (see first screenshot in this section).

To re-register a product, go to the product's pre-configuration page.

2.4.7 Configuration Page Reference

The LicenseServer Configuration page is the administrator's interface with LicenseServer. It allows the management of LicenseServer and the licensing of Altova products that have been registered with LicenseServer ([FlowForce Server](#), [MapForce Server](#), [StyleVision Server](#)).

The LicenseServer Configuration page is opened via a web browser. How to open the Configuration page is described in the sections, [Open LicenseServer Config Page \(Windows\)](#) and [Open LicenseServer Config Page \(Linux\)](#).

This section is a user's reference for the Configuration page and is organized by the tabs of the Configuration page:

- [License Pool](#)
- [Server Management](#)
- [Server Monitoring](#)
- [Server Monitoring](#)
- [Messages, Log Out](#)

For a step-by-step guide of how to assign licenses with LicenseServer, see the section [How to Assign Licenses](#).

License Pool

The **License Pool** tab displays all the licenses that are currently on the LicenseServer (see *screenshot below*). When a license file is uploaded to the LicenseServer with the **Upload** button on this page, all the licenses contained in the license file are placed in the license pool on the server and are displayed on the License Pool page.

The License Pool page displays information about all the licenses currently on the LicenseServer and thus provides a convenient overview of all Altova product licenses. On this page you can also activate, deactivate, and delete selected licenses.

Altova LicenseServer

ALTOVA® | LicenseServer

License Pool | Server Management | Server Monitoring | Settings | Messages | Log Out | Help

	Status	Name	Company	Product	Edition	Version	Key	Expires in day	SMP days left	CPU Cores	Running
<input type="checkbox"/>	Active	Mr. Nobody	Altova GmbH	Altova FlowForce Server		2013	CAWYXW8-	334	334	1	0
<input checked="" type="checkbox"/>	Active	Mr. Nobody	Altova GmbH	Altova FlowForce Server		2013	7CMJT18-	334	334	2	0
<input type="checkbox"/>	Active	Mr. Nobody	Altova GmbH	Altova MapForce Server		2013	MM5UC1U-	334	334	1	0
<input type="checkbox"/>	Active	Mr. Nobody	Altova GmbH	Altova RaptorXML+XBRL		2013	HC139LF-	334	334	1	0
<input type="checkbox"/>	Active	Mr. Nobody	Altova GmbH	Altova StyleVision Server		2013	3D78278-	334	334	1	0
<input type="checkbox"/>	Inactive	Mr. Nobody	Altova GmbH	Altova FlowForce Server		2013	966PPHM-	334	334	3	0
<input type="checkbox"/>	Inactive	Mr. Nobody	Altova GmbH	Altova StyleVision Server		2013	DA5T2WU-	334	334	4	0

Activate Deactivate Delete

Upload License File Browse... Upload

Uploading a license

To upload a license file (which you receive from Altova GmbH for your Altova product), click the **Browse** button, browse for the license file and select it. On clicking **Upload**, all the licenses contained in the license file are placed in the license pool and displayed on the License Pool page (screenshot above).

License status

License status values are as follows:

- *Activating:* When a license is uploaded into the license pool of LicenseServer, the server will transmit license-related data to the `altova.com` master licensing server to validate, authenticate, and activate the license that was supplied. This is necessary to ensure compliance with the Altova license agreements. During this initial activation and authentication transaction—which typically lasts between 30 seconds and a couple of minutes, depending on your Internet connection, speed, and overall network traffic—the status of the license will be indicated as *Activating....*
- *Failed Verification:* If a connection with the `altova.com` master licensing server cannot be made, then the status of the license in the pool will be shown as *Failed Verification*. If this happens, check your Internet connection and firewall rules to ensure that LicenseServer is able to communicate with the `altova.com` master licensing server.
- *Active:* Once the license has been authenticated and activated, the status in the pool will change to *Active*.
- *Inactive:* If a license has been verified, but is present on another LicenseServer on the network, the status in the pool will be shown as *Inactive*. An *Inactive* status also results when a license is manually deactivated in the license pool by the administrator.
- *Blocked:* A license is shown in the license pool as *Blocked* if there was a problem authenticating the license and the `altova.com` master licensing server has not granted permission to the LicenseServer to use this license. This could be the result of a license agreement violation, over-usage of a license, or other compliance issues. Should you see a license showing up as *Blocked*, please contact Altova Support with your license information and any other relevant data.

These statuses are summarized in the table below:

Status	Meaning
<i>Activating...</i>	On upload, license information is sent to <code>altova.com</code> for verification. Refresh the browser to view the updated status. Verification and activation can take a few minutes.
<i>Failed Verification</i>	A connection to <code>altova.com</code> could not be made. After establishing a connection, either restart the service or activate the license (with the Activate button).
<i>Active</i>	Verification was successful, the license is active.
<i>Inactive</i>	Verification was successful, but the license is on another LicenseServer on the network. Licenses can be made inactive with the Deactivate button.

<i>Blocked</i>	Verification was not successful. License is invalid and is blocked. Contact Altova Support .
----------------	---

Note: After a license has been sent to `altova.com` for verification, the browser must be refreshed to see the updated status. Verification and activation can take a few minutes.

Note: If a connection to `altova.com` could not be made, the status will be *Failed Verification*. After establishing a connection, either restart the service or try activating the license with the **Activate** button.

Note: When a license is given a status of *Inactive* or *Blocked*, a message explaining the status is also added to the Messages log.

Only an active license can be assigned to a product installation. An inactive license can be activated or deleted from the license pool. If a license is deleted from the license pool, it can be uploaded again to the pool by uploading the license file containing it. When a license file is updated, only those licenses in it that are not already in the pool will be uploaded to the pool. To activate, deactivate, or delete a license, select it and then click the **Activate**, **Deactivate**, or **Delete** button, respectively.

Connection to the Master Licensing Server at `altova.com`

The Altova LicenseServer needs to be able to communicate with the Master Licensing Server at `altova.com` to validate and authenticate license-related data and to ensure continuous compliance with the Altova license agreements. If the Altova LicenseServer is unable to connect with the `altova.com` Master Licensing Servers for a duration of more than 5 days (= 120 hours), then the Altova LicenseServer will no longer permit the usage of any Altova software products connected to the Altova LicenseServer.

Any such loss of connection with the `altova.com` master servers will be logged in the [Messages tab](#) of the [Configuration page of the Altova LicenseServer](#). In addition, the administrator can configure the Altova LicenseServer to automatically send an alert email when the connection to `altova.com` is lost. Alert Mail settings are available in the [Settings tab](#) of the [Configuration page](#).

Activating, deactivating, and deleting a license

An active license can be deactivated by selecting the license and clicking **Deactivate**. An inactive license can be activated (**Activate** button) or deleted (**Delete** button). When a license is deleted it is removed from the license pool. A deleted license can be added again to the license pool by uploading the license file containing it. If a license file is re-uploaded, only licenses that are not already in the license pool will be added to the license pool; licenses that are already in the pool will not be re-added.

License information

The following license information is displayed:

- *Status:* There are five values: *Failed Verification* | *Activating* | *Active* | *Inactive* | *Blocked*. See [License status](#) above.
- *Name, Company:* The name and company of the licensee. This information was

submitted at the time of purchase.

- *Product, Edition, Version*: The version and edition of the licensed products.
- *Key, Expires in days, SMP (days left)*: The license key to unlock the product, and the number of days left before the license expires. Each licensed purchase comes with a Support & Maintenance Package, which is valid for a certain number of days. The *SMP* column notes how many SMP days are still left.
- *CPU Cores*: The number of CPU cores that the license allows.
- *Running*: The number of CPU cores currently using the license.

Note on cores and licenses

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 on a particular server machine must be greater than or equal to the number of cores available on that server, whether it's a physical or virtual machine.

For example, if a server has eight cores (an octa-core processor), you must purchase at least an 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 an 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 license can be used for only one client machine at a time, even if it 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 the license cannot be used simultaneously for another client machine.

Server Management

In the **Server Management** tab (*screenshot below*), you can assign licenses to [registered products](#).

ALTOVA® | LicenseServer

License Pool | **Server Management** | Server Monitoring | Settings | Messages | Log Out | Help

▼ DOC.altova.com

Altova FlowForce Server 2013 Requires licenses for 2 CPU cores	Key Code Max licensed CPU cores	CPU Cores 0		
Altova MapForce Server 2013 Requires licenses for 2 CPU cores	Key Code Max licensed CPU cores	CPU Cores 0		
Altova StyleVision Server 2013 Requires licenses for 2 CPU cores	Key Code Max licensed CPU cores	CPU Cores 0		

Request evaluation licenses

Unregister server and all products

Note the following points about the Server Management tab:

- Each product is listed under the name of its client machine. In the screenshot above, one client machine, named `Doc.altova.com`, has three Altova products registered with the LicenseServer. If an Altova product on a different client machine is registered with this LicenseServer, then that client machine, with its registered products, will also be listed in the Server Management tab.
- Each registered Altova product on a client machine has its own *Key Code* entry, which takes the key code of a license. A registered product's key code is assigned by clicking its **Edit Assigned Licenses** button and selecting the required license from those available for that product (for example, FlowForce Server) in the license pool. This procedure is explained in more detail below.
- Each product also has a line stating how many CPU cores need to be licensed to run that product on that client. If the number of licensed cores is less than the number required, then the information is marked in red (see screenshot above). (The number of CPU cores that need to be licensed is the number of CPU cores on that client and is obtained from the client machine by LicenseServer.)

Icons in the Server Management tab



Edit Assigned Licenses. Available with each product. Pops up the Manage Licenses dialog, in which new licenses can be assigned to the product and already assigned licenses can be edited.



Show Licenses. Appears with each license. Switches to the License Pool tab and highlights the selected license, so that license details can be read.



Unregister This Product. Available with each product. The selected product (on the selected client machine) will be unregistered from LicenseServer.

Assigning a license

To assign a license to a registered product, click the **Edit Assigned Licenses** button of that product. This pops up the Manage Licenses dialog (*screenshot below*).

Available licenses	
CPU Cores	Key Code
<input type="checkbox"/>	3 CAWYXW8-L7YT8UW-C6H47TD-97U44WL-ULM45J1
<input checked="" type="checkbox"/>	2 7CMJT18-67HJD8J-1FJ98WL-J6DA11L-THY95J1
<input type="checkbox"/>	4 966PPHM-FYYW22H-96D9CWP-J1D6FCP-42M8871
<input type="checkbox"/>	4 LD60DCA-8HC297J-HAMAF4Y-FCX6H1Y-AY0W0J1

Apply Changes

Go to License Pool

Select the license you wish to assign. After selecting the license/s, click **Apply Changes**. The license/s will be assigned to that product and displayed in the Server Management tab (see *screenshot below*).

Altova FlowForce Server 2013 Requires licenses for 2 CPU cores	Key Code	CPU Cores		
	7CMJT18-67HJD8J-1FJ98WL-J6DA11L-THY95J1	2		
	Max licensed CPU cores	2		

Requesting an evaluation license

You can obtain a 30-day free evaluation license for each of a client's installed Altova products that have been registered with LicenseServer. Click the **Request Evaluation Licenses** button near the bottom of the Server Management tab. A dialog pops up containing a list of the Altova server products (on that client machine) which have been registered with LicenseServer. Make sure that the products for which you want an evaluation license are checked, then fill in the registration fields, and send the request. You will receive an e-mail from Altova containing the

30-day evaluation license/s. The number of cores for which the license will be valid per product will be exactly the number required by the product at the time the request is sent. Save the license/s to disk and [upload to the license pool](#).

Unregistering products

Each Altova product registered with LicenseServer is listed in the Server Management tab under its client machine name and has an **Unregister** icon to its right. Click this icon to unregister the product. If a license was assigned to the product, the assignment will be terminated when the product is unregistered. To unregister all products, click the **Unregister Server and All Products** button at the bottom of the Server Management tab (*see first screenshot in this section*).

To re-register a product, go to the product's Setup page.

For more information, see the section, [Assigning Licenses to Registered Products](#).

Server Monitoring

The **Server Monitoring** tab provides an overview of servers currently running licensed Altova products. It contains product information along with information about users and licenses.

Settings

The **Settings** tab is as shown below. You can set the following:

- The password for logging in to LicenseServer.
- Network settings for the web-based configuration page (Web UI), and for LicenseServer (License Service). These settings are described below the screenshot.
- Email server settings and the alert mail recipient. These are described below the screenshot.

License Pool	Server Management	Server Monitoring	Settings	Messages	Log Out	Help
--------------	-------------------	-------------------	----------	----------	---------	------

LicenseServer Password

Change Password

Web UI

Configure the host addresses where the web UI is available to administrators.

☒ All interfaces and assigned IP addresses

☐ Local only (localhost)

☐ Only the following hostname or IP address:

Ensure this hostname or IP address exists or LicenseServer will fail to start!

Configure the port used for the web UI.

☐ Dynamically chosen by the operating system

☒ Fixed port

Ensure this port is available or LicenseServer will fail to start!

License Service

Configure the host addresses where the LicenseServer service is available to clients.

☒ All interfaces and assigned IP addresses

☐ Local only (localhost)

☐ Only the following hostnames or IP addresses:

Ensure the hostnames or IP addresses exist or LicenseServer will fail to start!

Alert Mail

Configure email settings for communication with administrator.

SMTP Host

SMTP Port

User authentication

User password

From

To

Send Test Mail

Save

Network settings

Administrators can specify network access points to the LicenseServer configuration page and to LicenseServer:

- *Web UI:* Allowed IP addresses can vary from all interfaces and IP addresses on that machine to a fixed address, and ports can be either dynamically calculated or fixed. This allows a wide range of allowed IP-Address:Port settings. The default port setting is **8088**.
- *License Service:* IP addresses can vary from all interfaces and IP addresses on that machine to a fixed address. The port number is fixed at **35355**.

By default, these settings allow unrestricted access to LicenseServer and its configuration page from within the networks to which LicenseServer is connected. If you wish to restrict access to either LicenseServer or its configuration page, enter the appropriate settings and click **Save**.

Alert Mail settings

Altova LicenseServer needs to be connected to the `altova.com` server. If the connection is broken for more than 24*5 hours (5 days), LicenseServer will not allow licenses. As a result, work sessions with Altova products licensed by LicenseServer could be disrupted. In order to alert that administrator that a connection is broken, an alert mail can be sent to an email address. The Alert Mail pane of this page enables settings to be made for sending alert mails to an administrator's email address.

SMTP Host and *SMTP Port* are the settings of the email server from which the email alert will be sent. *User Authentication* and *User Password* are the user's credentials for accessing the email server. The *From* field takes a name by which email alerts can be recognized by the recipient, for example: `Altova LicenseServer`. The *To* field takes the recipient's email address.

Click **Save** when done. After saving the Alert Mail settings, email alerts will be sent to the address specified whenever a significant event occurs, such as when connection to `altova.com` is lost. These events are also recorded in the [Messages tab](#).

Messages, Log Out

The **Messages** tab displays all messages relevant to licenses in the license pool of the LicenseServer. Each message has a **Delete** button that allows you to delete that particular message.

The **Log Out** tab serves as the Log Out button. Clicking the tab logs you out immediately and then displays the Login mask.

2.5 FlowForce Server data storage

FlowForce Server is a background service/daemon that stores all configuration and other important data in a database. Configuration data can consist of: user profiles, roles, triggers and jobs. FlowForce Server has no user interface and is configured via FlowForce [Administration Interface](#).

FlowForce Server stores all data in the following locations:

Windows XP	C:\Documents and Settings\All Users\Application Data\Altova\FlowForceServer2013
Windows Vista, Windows 7/8	C:\ProgramData\Altova\FlowForceServer2013
Linux	/var/opt/Altova/FlowForceServer2013

Global settings

- **flowforceserver.ini**: global configuration settings for FlowForce Server (e.g. language)
- **flowforceweb.ini**: global configuration settings for FlowForce Web Server (e.g. language)
- **flowforceweb.html**: the Setup page page used to register FlowForce with LicenseServer and to start the servers. Note: This page is regenerated every time you start FlowForce Web Server.

Data directory

- **ffweb.log / flowforce.log**: log files for FlowForce Web Server and FlowForce Server
- **flowforce.db**: the main database file storing the FlowForce Server object system, user data, active jobs, roles, etc.
- **flowforceelog.db**: the database that stores all FlowForce Server logs
- **flowforce.ini**: the configuration file defining the port and listening interfaces of FlowForce Server
- **flowforceweb.ini**: the configuration file defining the port and listening interfaces of FlowForce Web Server
- **"files"** subdirectory: stores files associated with deployed functions
- **"logs"** subdirectory: contains captured output from job execution steps
- **"tmp"** subdirectory: stores temporary files
- **"tools"** subdirectory: can be used to override tool selection, usually empty

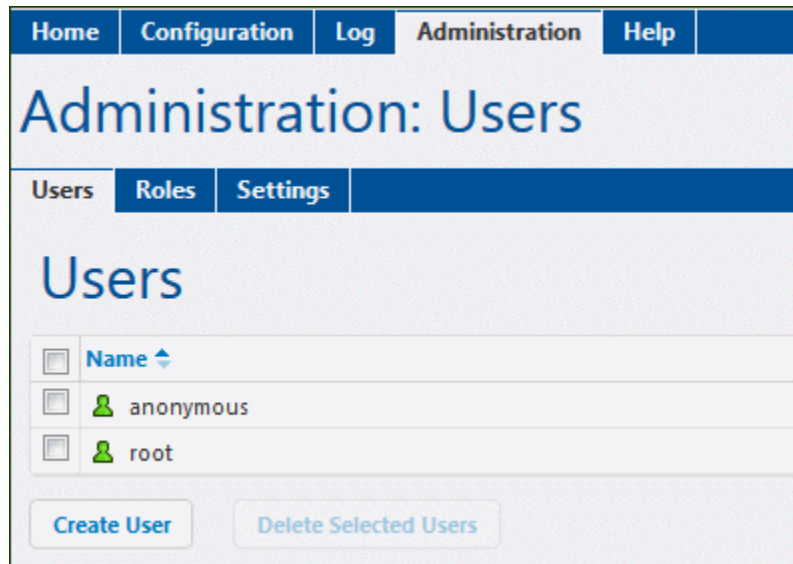
Both FlowForceServer and FlowForceWebServer use ini style configuration files. Administrators can edit the files with a text editor, or use the web-based Setup page.

2.6 FlowForce Administration Interface

FlowForce Server Administration Interface is the module that acts as the front-end of FlowForce Server. FlowForce Server Administration Interface can be accessed using an internet browser and allows you to configure the specific server actions such as: jobs, triggers, etc.

FlowForce Server supports the current versions of Mozilla Firefox, Google Chrome, and Microsoft Internet Explorer 9 and 8. Note: When using Internet Explorer 9 as your browser, please disable the "Show friendly HTTP error messages" in the Advanced tab, to view the HTML form.

There are several menu items available in the browser window:



Home

Displays the connection details as well as a list of currently running/aborted jobs and active triggers.

Configuration

Displays the contents of the container hierarchy, and allows you to access containers, jobs, credentials, and functions by clicking them in the list. New containers, jobs, and credentials, are added using the "Create " button. See the [User Guide](#) for more details.

Log

Displays the log table. Can be filtered by time, job name and severity.

Administration

Lets you view and edit/access server-wide configuration, such as user accounts, roles and other settings.

[Users](#)

Allows you to create, remove, and maintain users.

[Roles](#)

Allows you to create, remove, and maintain access control roles.

[Settings](#)

Allows you to define the global settings of FlowForce, i.e. the default time zone and the default mail server settings.

Help

Opens the FlowForce Server documentation in a separate browser tab or window.

2.7 FlowForce access control

FlowForce Server grants access control to containers and configurations. Logged-in users are granted privileges and permissions based on explicit assignments to their user account, as well as the roles the user is a member of.

Privileges are global rights, independent of the FlowForce Server container hierarchy, whereas permissions are inherited down the hierarchy and can be refined at each level.

Users and Roles:

Users are persons that have been added to FlowForce Server by the FlowForce Server administrator. Depending on the assigned rights and privileges, users can define or run FlowForce Server jobs.

Roles are used to manage privileges and object permissions for user groups as opposed to individual users.

Privileges:

Privileges control user rights globally and can be defined in the Users and Roles pages.

Permissions:

Permissions control access to containers and configurations. Unlike privileges they can be redefined on every level of the container hierarchy, and are by default inherited from parent containers.

Credentials:

Credentials are stored login data used to execute FlowForce jobs.

The same system used for Linux.

2.7.1 Users and Roles

A user account defines a log-in name and has a set of roles the user is a member of. A role can be a member of another, broader role, which makes all members of the narrower role also members of the broader role.

This enables the definition of hierarchical access rights, the role Director of Marketing may be a member of Marketing, which in turn may be a member of Employees. Assigning a user, e.g. Bob, the role Director of Marketing, automatically makes him a member of Marketing and of Employees as well. When Bob logs into FlowForce Server he will be granted all privileges and permissions granted to any of his roles.

Two special users are predefined by FlowForce:

- **root** is the initial administrator user. It is by default all-powerful and allows tasks such as adding other users and roles, as well as setting up privileges and permissions.
- **anonymous** is a special user account for users that do not explicitly log in. Anonymous access to the FlowForce Server Administration Interface is not possible, but you can enable anonymous access for certain services exposed via the [HTTP service interface](#).

Two special roles are also predefined by FlowForce:

- **authenticated** is the role automatically assigned to every user **except** anonymous. It therefore includes every user who is authenticated using an existing user name and password.
- **all** is the role automatically assigned to every user including anonymous.

While any users you create will be members of both all and authenticated, any roles you create are not by default members of any other role.

See also:

[Privileges](#)

[Permissions](#)

[Credentials](#)

[Defining restricted user rights](#)

How to add Users

This page allows Administrators to define users and assign them user roles. The user name and password needed to access the web administration interface, or to deploy MapForce mappings, are defined here.

To add a user to FlowForce:

1. Click the "Create User" button on the Users page (of the Administration tab).

	Name
<input type="checkbox"/>	anonymous
<input type="checkbox"/>	root

Create User Delete Selected Users

2. Enter the User name and password.

Create User

User name: Operator

Password: ...

Re-type Password: ...

3. Define the privileges of this user (on the same browser page) by activating the [Privileges](#) check boxes. Note users inherit all privileges of their assigned roles; we therefore recommend assigning privileges to roles only to simplify maintenance.

Privileges

☐ Maintain global settings

☒ Maintain users, roles and privileges

☐ Set own password

☐ Override security

☒ View unfiltered log

☒ Read users and roles

Save

4. Click Save to save the user.

Users and Roles

Roles make it easy to define user groups such as project teams, branch offices etc. To assign roles to a user on **this** page, roles must have been previously defined on the [Roles](#) page. Note that two default roles have already been assigned to this user, i.e. "all" and "authenticated".

User Operator

[Change password](#)

Privileges

- ☒ Maintain users, roles and privileges
- ☒ Set own password
- ☐ Override security
- ☒ View unfiltered log
- ☒ Read users and roles

[Save](#)

Assigned Roles

Roles available

<input type="checkbox"/>	Name
<input type="checkbox"/>	Name

[Assign >>](#)

[<< Remove](#)

Roles assigned to the user 'Operator'

<input type="checkbox"/>	Name
<input type="checkbox"/>	all
<input type="checkbox"/>	authenticated

You can however assign a user to a role on the [Roles page](#) if you want to.

To assign one or more roles to a user:

1. Click the role name check box(es) of the role(s) you want to assign (e.g. Deploy_mapping which is added in the following [Roles](#) topic).
2. Click the "Assign" button to assign the role(s) to this user.
Role assignments are saved immediately.

Assigned Roles

Roles available

<input type="checkbox"/>	Name
<input checked="" type="checkbox"/>	Deploy_mapping

[Assign >>](#)

[<< Remove](#)

Roles assigned to the user 'Operator'

<input type="checkbox"/>	Name
<input type="checkbox"/>	all
<input type="checkbox"/>	authenticated

The Deploy_mapping role has now been assigned to the user "Operator".

The interface is titled "Assigned Roles". It is divided into two main sections. The left section, "Roles available", contains a search box with the placeholder text "Name" and a dropdown arrow. Below it are two buttons: "Assign >>" and "<< Remove". The right section, "Roles assigned to the user 'Operator'", contains a list of roles. Each role entry has a checkbox on the left and the role name on the right. The roles listed are "Deploy_mapping", "all", and "authenticated".

Roles available	
<input type="checkbox"/>	Name

Assign >>

<< Remove

Roles assigned to the user 'Operator'	
<input type="checkbox"/>	Name
<input type="checkbox"/>	Deploy_mapping
<input type="checkbox"/>	all
<input type="checkbox"/>	authenticated

Note that the roles "all" and "authenticated" are default roles supplied with FlowForce.

See also: [How to add Roles](#)

How to add Roles

To add a role to FlowForce:

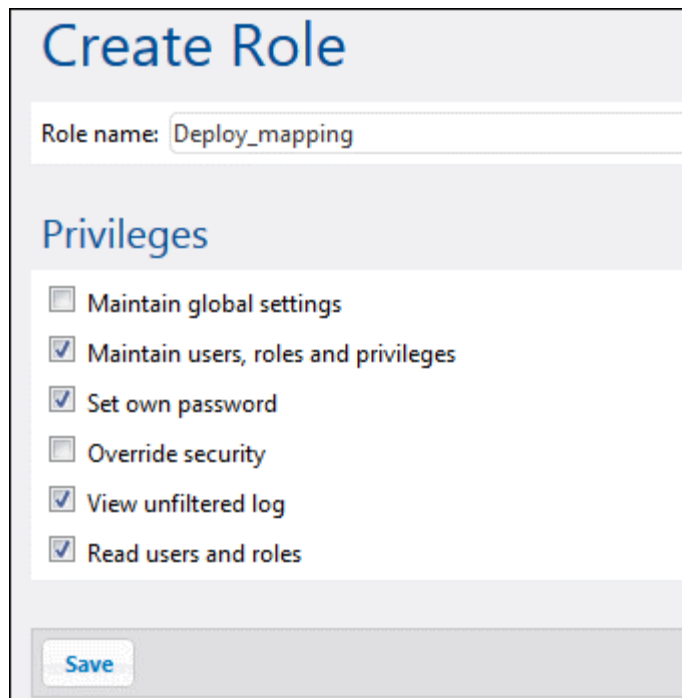
1. Click the "Create Role" button on the Roles page.

The interface shows the "Administration: Roles" page. It has a top navigation bar with "Home", "Configuration", "Log", "Administration", and "Help". Below this is a sub-navigation bar with "Users", "Roles", and "Settings". The main content area is titled "Roles" and contains a list of roles. Each role entry has a checkbox on the left and the role name on the right. The roles listed are "all" and "authenticated". At the bottom of the list are two buttons: "Create Role" and "Delete Selected Roles".

Administration: Roles	
<input type="checkbox"/>	Name
<input type="checkbox"/>	all
<input type="checkbox"/>	authenticated

Create Role Delete Selected Roles

2. Enter the Role name (e.g. Deploy_mapping) and define the privileges the members of this role should have, by activating the [Privileges](#) check boxes, then click Save to save the role.



The 'Create Role' form has a title bar 'Create Role'. Below it is a text input field for 'Role name' containing 'Deploy_mapping'. A section titled 'Privileges' contains a list of checkboxes: 'Maintain global settings' (unchecked), 'Maintain users, roles and privileges' (checked), 'Set own password' (checked), 'Override security' (unchecked), 'View unfiltered log' (checked), and 'Read users and roles' (checked). At the bottom is a 'Save' button.

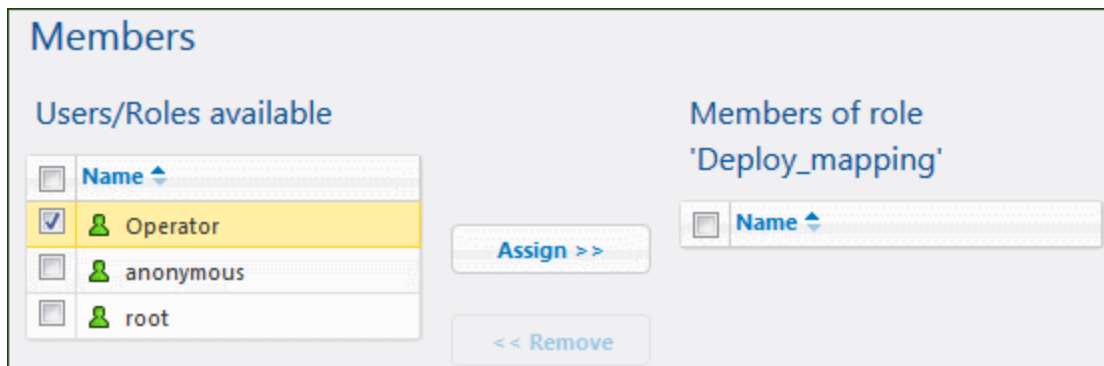
Note: Users and roles cannot have the same names in FlowForce.

Roles and Users

To assign a user to a role on **this** page (Role Deploy_mapping), users must have been previously defined on the [Users](#) page.

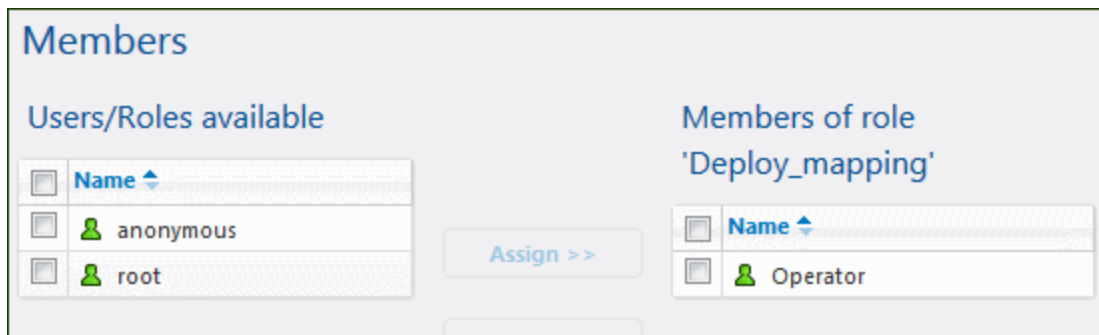
To assign a user to a role:

1. Click the user (or role) check box that you want to assign the role to, in the "Users/Roles available" table.
2. Click the "Assign" button to assign the user(s) to this role.



The 'Members' page has a title bar 'Members'. It is divided into two main sections. The left section, 'Users/Roles available', contains a table with a header 'Name' and a dropdown arrow. The table has three rows: 'Operator' (checked), 'anonymous' (unchecked), and 'root' (unchecked). The right section, 'Members of role 'Deploy_mapping'', contains a table with a header 'Name' and a dropdown arrow. Between the two sections are two buttons: 'Assign >>' and '<< Remove'.

The user "Operator" has now been added to the Deploy_mapping role.

**Assigning roles to roles**

FlowForce Server makes it possible for you to assign roles to roles. What this does is allow indirect inheritance of permissions/privileges.

- If user A is a member of role B, and role B is a member of role C, then user A is also an indirect member of C.

See also: [Defining restricted user rights](#)

Defining restricted user rights

To restrict access to the FlowForce Server /public container, it is recommended to perform the following steps:

- Take away permissions and privileges from the authenticated role.
- Group users requiring extra permissions in a new role.
- Assign extra permissions to the new role.

To take away permissions and privileges from the authenticated role:

1. Go to the Configuration page.
2. Click the "Permissions" button to the right of the /public container.
3. Click the "Change" button for authenticated.
4. Define the permissions that should apply to all users.
5. Click "Save Changes" to commit your changes.

To group users requiring extra permissions in a new role:

1. Go to the Administration page.
2. Click Roles to go to see the roles list.
3. Click the "Create Role" button to create a new role.
4. Enter a descriptive role name, e.g. Operators.
5. Click "Save" to save your changes.
6. Select the users you want to assign to the role from the list on the left (Users/Roles available).
7. Click "Assign" to assign them to the new role.

To assign extra permissions to the new role:

1. Go to Configuration page.
2. Click the "Permissions" button next to the /public container.
3. Click "Add Permissions".
4. Select the role from the combo box.
5. Define the extra permissions for this role.
6. Click "Save Changes" to commit your changes.

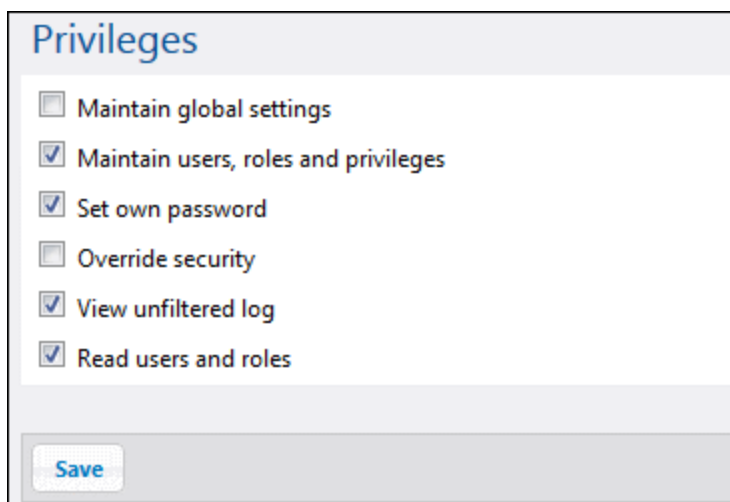
See also:
[Privileges](#)

2.7.2 Privileges

Privileges control several user rights **globally**. This means privilege settings cannot be overridden in the container hierarchy of FlowForce. When a user logs into FlowForce, the set of effective privileges is determined by the user privileges and all role privileges the user is member of.

It is recommended to assign privileges only to roles, and have the assignment to users take place via role membership.

Privileges are assigned on the user and role pages of the FlowForce Server Administration Interface.

A screenshot of a web interface titled "Privileges". It contains a list of six checkboxes with corresponding labels: "Maintain global settings", "Maintain users, roles and privileges", "Set own password", "Override security", "View unfiltered log", and "Read users and roles". The checkboxes for "Maintain users, roles and privileges", "Set own password", "View unfiltered log", and "Read users and roles" are checked. At the bottom left of the form is a "Save" button.

Maintain global settings

A user having this privilege can change the global settings of FlowForce Server that are to be found on the Administration | Settings page. These are the time zone and SMTP settings set up by the administrator.

Maintain users, roles and privileges

Any user having this privilege can create, delete and edit users and roles, their privilege assignments and passwords.

This is an administrative privilege and should only be assigned to FlowForce Server administrators.

By default, only the user "root" possesses this privilege.

Set own password

Any user having this privilege can change his own password. Users who do not have this privilege need to have their password set by a FlowForce Server administrator.

By default the "authenticated" role, and hence every user account except "anonymous", possesses this privilege.

Override security

Any user having this privilege can change permissions in the container hierarchy without needing "write" security permission. This allows FlowForce Server administrators to regain access to resources accidentally rendered inaccessible.

This is an administrative privilege and should only be assigned to FlowForce Server administrators.

By default, only "root" possesses this privilege.

View unfiltered log

By default users can only see log entries related to Configurations they have "read" access to. By granting this privilege a user can read all log entries, including those not associated with a specific configuration.

By default, only "root" possesses this privilege.

Read users and roles

By default users will only see their own user account and any roles they are member of. By granting this privilege a user can read all defined users and roles.

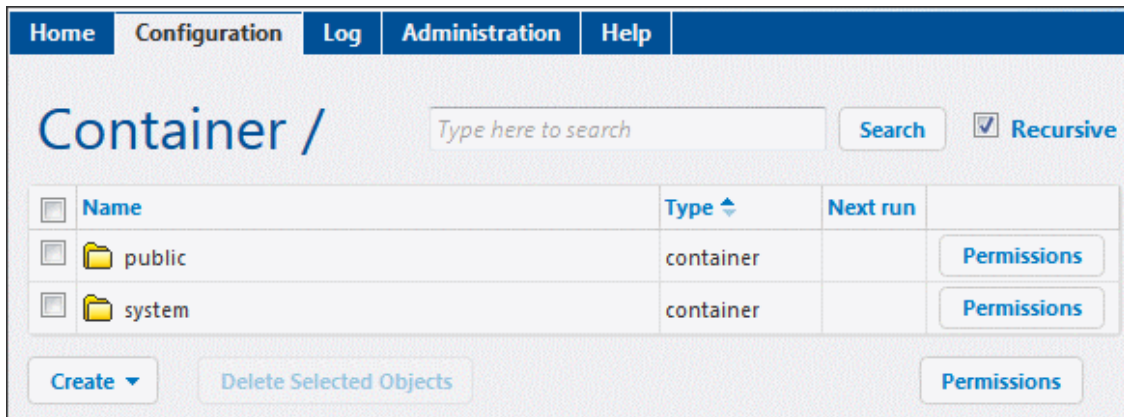
By default, only "root" possesses this privilege.

See also:

[Permissions](#)

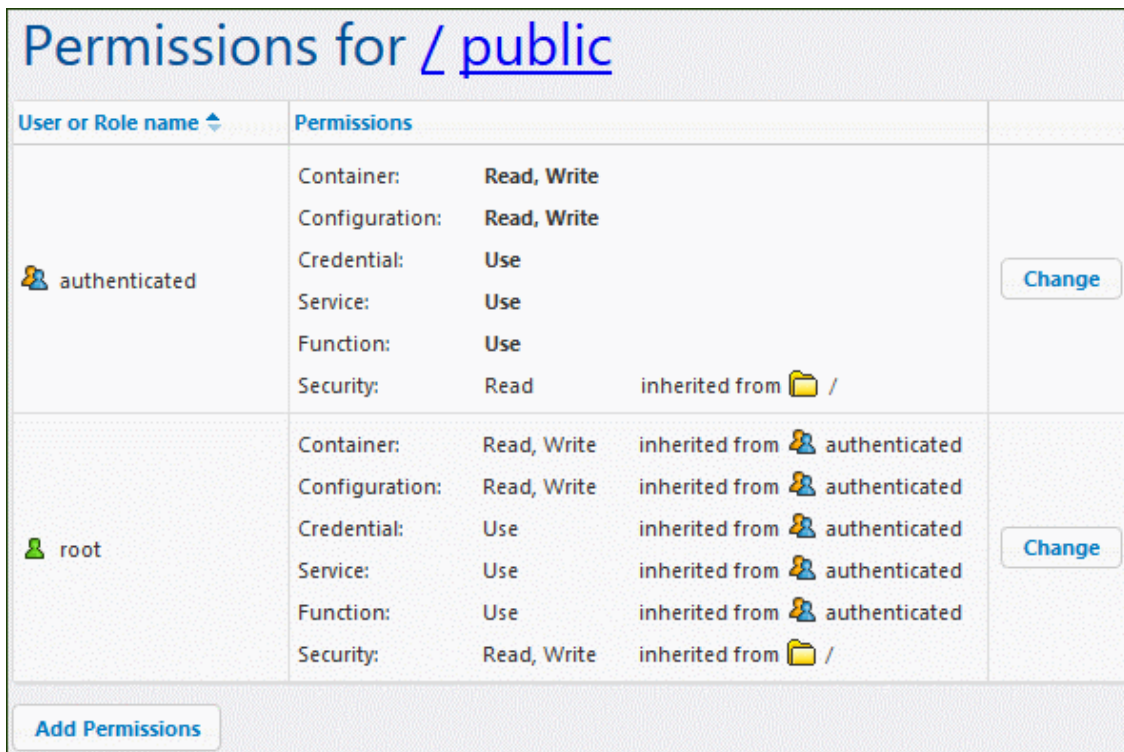
2.7.3 Permissions

Permissions control access to containers and configurations. Unlike privileges they can be redefined on every level of the container hierarchy, and are by default inherited from parent containers. Click the Configuration tab and the Permissions button (of that row, e.g. public) to see the container permissions.



Permissions, like privileges, are inherited from all roles the user is a member of, as well as from permissions directly assigned to the user.

This inheritance takes precedence over container hierarchy inheritance. If a permission is redefined for any role the user is a member of, container hierarchy inheritance for this particular permission is overridden.



Permission checks are performed on every user interaction. A user can only successfully edit a Configuration when all required permissions are granted. Permissions are not evaluated upon

job execution, therefore any permission changes will not retroactively apply to already previously jobs.

FlowForce Server assigns permissions in several groups:

Container

The container permissions define what a user can do with objects in a container.

If the container has "read" access, the user can list the contents and find an object in the container.

If the container has "read"/"write" access, the user can additionally create new (and delete existing) objects within the container, depending on other permissions that may apply.

Configuration

The configuration permissions define what a user can do with a configuration (job, credential).

If the configuration has "read" access, the user can look at the details of that configuration, such as the defined execution steps or the defined triggers.

If the configuration has "read"/"write" access, the user can additionally modify that configuration. To successfully create a new configuration, or delete an existing one, the user must be permitted "write" access for the container and the configuration.

Service

The service permission defines access to a job via the HTTP request interface.

If the service has "use" access, the user is permitted to access the service and thus execute the job via the request interface.

Note that service permission checks skip any container hierarchy checks. If a user is permitted to use a service he may do so without having "read" access to the container the corresponding job is defined in.

Also note that by granting service use to "anonymous", it becomes possible to use that service without any authentication.

Credential

The credential permission defines what a user can do with a credential. This makes it possible to supply credentials for reuse.

If the credential has "use" access, the user is permitted to refer to this credential from another configuration.

Function

The function permission defines whether a user can invoke a function as an execution step in another function.

If the function has "use" access, it is permitted to call this function from another function.

Security

The security permission controls access to the container's child permission lists.

If security has "read" access, the user is permitted to read the permission list of any child of the container.

If security has "read"/"write" access, the user can additionally change the permission list of any

child of the container.

By default users are permitted to read only permissions applicable to them. That means any permissions assigned to themselves or any role they are a member of. If the "Read Users and Roles" privilege is granted, users can read all permission entries.

See also:

[How to add Permissions](#)

How to add Permissions










Permissions can be assigned to both a user and a role. Read/Write/Use permissions should generally only be assigned to **roles** and not to individual users, to simplify maintenance.

The screenshot displays the 'Container / public /' interface. At the top, there is a search bar with the placeholder text 'Type here to search', a 'Search' button, and a checked 'Recursive' checkbox. Below this is a table with the following columns: 'Name', 'Type', and 'Next run'. The table contains five entries:

Name	Type	Next run
ChainedPersonList	function	
ChainedPersonListJob	job	2012-08-02 11:00:00
DB_PhoneList	function	
DB_PhoneListJob	job	2012-08-02 11:00:00
InputParams	credential	

Each entry has a checkbox on the left and a 'View log' button on the right. At the bottom of the interface, there are three buttons: 'Create', 'Delete Selected Objects', and 'Permissions'.

Permissions are accessed by clicking the "Permissions" button at the bottom of the right of the page, or next to the container that you want to define the permissions for. The screenshot below shows the permissions overview for the /public container.

Permissions for <u>/ public</u>			
User or Role name	Permissions		
 authenticated	Container:	Read, Write	
	Configuration:	Read, Write	
	Credential:	Use	
	Service:	Use	
	Function:	Use	
	Security:	Read	inherited from  /
 root	Container:	Read, Write	inherited from  authenticated
	Configuration:	Read, Write	inherited from  authenticated
	Credential:	Use	inherited from  authenticated
	Service:	Use	inherited from  authenticated
	Function:	Use	inherited from  authenticated
	Security:	Read, Write	inherited from  /
<div>Add Permissions</div>			

Permissions can be inherited from containers above the current container, if "Inherit" is selected (shown as the forward slash character / for the root container) on the Permissions overview page shown above.

To add a new Permission to a user or role:

Having clicked the "Permissions" button on the Container page:

1. Click the "Add Permissions" button on the Permission overview page of the container you want the permission to be added to, e.g. Container /public.
2. Select a previously defined role (or user) from the "User or Role" combo box e.g. "Deploy_mapping (role)".

Edit Permissions

User or Role:
Deploy_mapping (role)

Container:
Inherit

Configuration:
Inherit

Service:
Inherit

Credential:
Inherit

Function:
Inherit

Security:
Inherit

Set for all:

Inherit











No access


Save Changes

Discard Changes

3. Change the "Configuration" permission to "Read/Write" and leave the other permissions as they are currently defined (Inherit for all), then click "Save Changes".

Permissions for Container / public

User or Role name	Permissions	
 <u>Deploy_mapping</u>	Configuration: Read, Write	Change
 <u>authenticated</u>	Container: Read, Write Configuration: Read, Write Credential: Use Service: Use Function: Use Security: Read inherited from  <u>/</u>	Change
 <u>root</u>	Container: Read, Write inherited from  <u>authenticated</u> Configuration: Read, Write inherited from  <u>authenticated</u> Credential: Use inherited from  <u>authenticated</u> Service: Use inherited from  <u>authenticated</u> Function: Use inherited from  <u>authenticated</u> Security: Read, Write inherited from  <u>/</u>	Change



[Add Permissions](#)

The Deploy_mapping permission list has now been added to the container "/public".

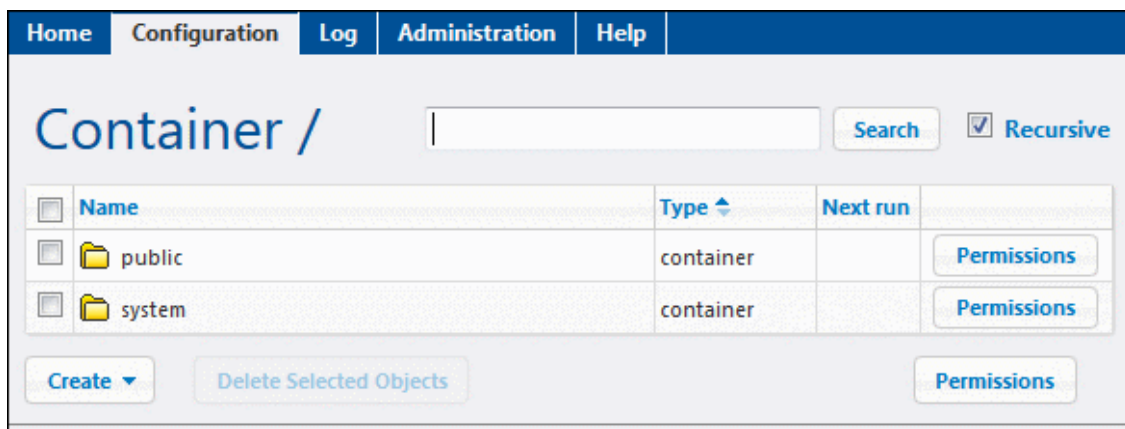
2.7.4 Credentials

Credentials are stored login data used to execute FlowForce Server jobs. Credentials can be defined as standalone "objects" and be assigned to various jobs, or they can be manually entered for a specific job.

Jobs are started automatically by FlowForce Server when the defined trigger conditions are met. FlowForce Server then runs these jobs using a specific operating system user account, ensuring that job steps do not access unauthorized data. Note that [file watch triggers](#) are also assigned credentials.

Credentials can be created, or deleted, on the Configuration (Container) page. Note that job credentials, i.e. username and password, can also be entered for individual jobs on the Job page.

Any user that has "write" access to the "Configuration" permission, can edit or remove credentials.



To add a credential to FlowForce Server:

1. Click the container you want to create the new credential in, e.g. public.
2. Click the "Create" button and select the "Create Credential" entry.



3. Enter the name of the credential as well as the **operating system** user name and password. To specify a user name in a Windows domain, please use the form **username@domain**.

The screenshot shows a web form titled "Create credential in / public /". It contains two input fields: "Credential name:" with the value "Cred_production" and "Credential description:" which is empty. Below these is a section titled "Credential" containing "User name:" with the value "production" and "Password:" with four dots. A "Save" button is at the bottom left.

4. Click Save.
The new credential "Cred_production" has been saved in the /public container.
5. Click the "Configuration" button to return to the Container page.

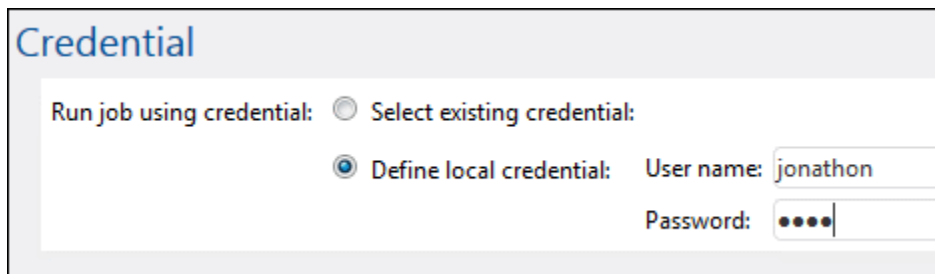
The screenshot shows a web page titled "Credential Cred_production in / public /". It has a "Referenced by" tab at the top. Below it is a "Credential description:" field. The "Credential" section shows "User name:" as "production" and "Password:" with a "Change password" button. At the bottom are "Save" and "Delete" buttons.

Please see [Permissions](#) for information on the container permissions that can be defined.

Credentials and jobs

Every job MUST have a credential assigned to it for the job steps to be executed. This defines the **operating system** user account used to run the job steps.

A predefined credential can be selected using the "existing credential" combo box, or the local credential can be manually entered in the "User name" and "Password" fields. This is done on the "Create job..." page.



Note:

If you **manually** enter the user name and password, as shown above, you will have to update them for those specific jobs, whenever your server credentials are changed.

Credentials:

- Credentials can be created in any container a user has access to.
- The credential password may be an empty string.
- As the clear text password needs to be sent to the operating system's login function, passwords are stored in a reversible encrypted form in the FlowForce Server database. The administrator should make sure to restrict access to the FlowForce Server database file.

FTP credentials

When using FTP system functions, e.g. ftp/retrieve, in the "Execute function" field you must also enter your FTP login credentials for the FTP server. You can select an existing credential or enter a local one.

Required Permissions for a user to execute a job

Since a job is defined to run with a certain credential, it is vital for the success of the job execution that the **operating system user** which is referenced by this credential has sufficient access permissions.

The operating system user needs the following **file system** permissions:

- Execution permission for the used server executables and all referred DLLs (implicitly set).
- Read permission for all paths used in the input files of the job.
- Write permission for all paths used in the output files of the job.
- Read and/or Write permission for the working directory, depending on the specific job.

2.8 Settings

This page contains the system-wide global settings of FlowForce Server.

Administration: Settings

Users Roles Settings

Settings

Input format

Default time zone: Europe/Berlin ▼

System function [/system/mail/send](#) parameters

SMTP server: localhost

SMTP port: 25

User authentication: +

Default sender: +

Save

Altova FlowForce® 201

Input format

lets you select your local time zone.

System function [/system/mail/send](#) parameters

lets you define your SMTP server settings and enter the user authentication for the SMTP server used by the built-in function [/system/mail/send](#).

2.9 Command Line Usage

This section describes the command line interface of FlowForce Server.

Default location on Windows

The FlowForce Server executable (`FlowForceServer.exe`) is located by default at:

```
<ProgramFilesFolder>\Altova\FlowForceServer\bin\FlowForceServer.exe
```

Default location on Linux

On Linux systems, the FlowForce Server executable (`flowforceserver`) is located by default at

```
/opt/Altova/FlowForceServer2013/bin/flowforceserver
```

where the first forward slash indicates the root directory.

Usage

General command line syntax for `FlowForceServer` is:

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

The options and arguments in square brackets are optional.

where

<code>--h --help</code>	Displays the help text.
<code>--version</code>	Displays the version of FlowForce Server.

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

<code>help COMMAND</code>	Displays help for a specific command. For example: <code>help run</code>
<code>createdb</code>	Creates a new FlowForce database.
<code>debug</code>	Starts the application in debug mode.
<code>exportresourcestrings</code>	Exports all application resource strings to an XML file
<code>foreground</code>	Starts the application in foreground mode.
<code>initdb</code>	Creates or updates the FlowForce database.
<code>install</code>	Installs the application as a Windows service.
<code>licenseserver</code>	Register FlowForce Server with the Altova LicenseServer on the local network.

repair	Starts the application in repair mode.
setdeflang sdl	Sets the default language.
start	Starts the application as a service.
uninstall	Uninstalls the Windows service.
upgradedb	Upgrades the FlowForce database with the latest version

2.9.1 help

The `help` command takes a single argument: 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.

```
FlowForceServer help Command
```

Note: On Linux systems, use an all-lowercase `flowforceserver` to call the executable.

Example

Here's an example of the `help` command:

```
FlowForceServer help creatdb
```

The command above contains one argument: the command `creatdb`, for which help is required. When this command is executed, it will display information about the `creatdb` command.

The `--help` option

Help information about a command is also available by using the `--help` option with that command. For example, using the `--help` option with the `run` command, as follows:

```
FlowForceServer run --help
```

achieves the same result as does using the `help` command with an argument of `generate`:

```
FlowForceServer help run
```

In both cases, help information about the `run` command is displayed.

2.9.2 createdb

The `createdb` command creates a new database. If the database already exists then the command will fail.

```
FlowForceServer createdb [options]
```

Note: On Linux systems, use an all-lowercase `flowforceserver` to call the executable.

The default database is created at installation time, so it is usually not necessary to use this command.

Options

The single option is shown below.

	<code>--datadir=VALUE</code>	VALUE is the path of the data directory.
--	------------------------------	--

2.9.3 debug

The `debug` command starts FlowForce Server in debug mode, i.e. not as a service. To stop this mode press CTRL+C.

```
FlowForceServer debug [options]
```

Note: On Linux systems, use an all-lowercase `flowforceserver` to call the executable.

Not for general use.

Options

The single option is shown below.

	<code>--datadir=VALUE</code>	VALUE is the path of the data directory.
--	------------------------------	--

2.9.4 exportresourcestrings

The **exportresourcestrings** command outputs an XML file containing the resource strings of the FlowForceServer application. It takes two arguments: (i) the language of the resource strings in the output XML file, and (ii) the path and name of the output XML file. Allowed export languages (with their language codes in parentheses) are: English (*en*), German, (*de*), Spanish (*es*), and Japanese (*ja*).

```
FlowForceServer exportresourcestrings Language XMLOutput
```

Note: On Linux systems, use an all-lowercase **flowforceserver** to call the executable.

Arguments

The **exportresourcestrings** command takes the following arguments:

Language	Specifies the language of resource strings in the exported XML file. Allowed languages are: <i>en</i> , <i>de</i> , <i>es</i> , <i>ja</i>
XMLOutput	Specifies the location and name of the exported XML file.

Example

Here's an example of the **exportresourcestrings** command:

```
FlowForceServer exportresourcestrings en c:\Strings.xml
```

This command creates a file called *Strings.xml* at *c:* that contains all the resource strings of the FlowForce Server application in English.

Creating localized versions of FlowForceServer

Create a localized version of FlowForce Server as follows:

1. Generate an XML file containing the resource strings by using the **exportresourcestrings** command (see *command syntax above*).
2. Translate the resource strings 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 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\FlowForceServer2013\bin` folder. Your DLL file will have a name of the form `FlowForceServer2013_lc.dll`. The *lc* part of the name contains the language code. For example, in `FlowForceServer2013_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 FlowForce Server app to use. Use the language code that is part of the DLL name as the argument of the **setdeflang** command.

Note: Altova FlowForce Server is delivered with support for four languages: English, German, Spanish, and Japanese. So you do not need to create a localized version of these languages. To set any of these four languages as the default language, use FlowForce

Server's `setdeflang` command.

2.9.5 foreground

The `foreground` command starts Altova FlowForce Server in the foreground. It is used internally by the startup scripts for Linux and should not be specified by a user.

2.9.6 initdb

The `initdb` command creates a new database, or updates an existing one to the latest version.

```
FlowForceServer initdb [options]
```

Note: On Linux systems, use an all-lowercase `flowforceserver` to call the executable.

The default database is created at installation time, so it is usually not necessary to use this command.

Options

The single option is shown below.

	<code>--datadir=VALUE</code>	VALUE is the path of the database directory.
--	------------------------------	--

2.9.7 install

The `install` command installs Altova FlowForce Server as a Windows service.

```
FlowForceServer install [options]
```

The service is installed at installation time, so it is usually not necessary to use this command.

Options

The single option is shown below.

	<code>--datadir=VALUE</code>	VALUE is the path of the database directory.
--	------------------------------	--

2.9.8 licenseserver

The `licenseserver` command specifies the name of the machine on the network running the Altova LicenseServer. Alternatively, you can specify the machine's IP address.

The `licenseserver` command registers FlowForceServer with LicenseServer. Once this is done, you can, on LicenseServer, assign a license to FlowForceServer. How to do this is described in the LicenseServer documentation.

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

Note: On Linux systems, use an all-lowercase `flowforceserver` to call the executable.

You must have Administrator privileges (root) to be able to register FlowForce Server with LicenseServer!

Example

Here's an example of the `licenseserver` command in its simplest form:

```
FlowForceServer licenseserver DOC.altova.com
```

The command above specifies that the machine named `DOC.altova.com` is the machine running Altova LicenseServer. If the LicenseServer is running on the user's machine, the following commands would also be valid:

```
FlowForceServer licenseserver localhost
FlowForceServer licenseserver 127.0.0.1
```

Options

The options are listed below, in their short forms (first column) and long forms (second column), together with their descriptions. On the command line, one or two dashes can be used for both short and long forms.

<code>--j</code>	<code>--json</code>	Prints the result of the registration attempt as a machine-parsable JSON object. Form: <code>--json=true/false</code>
------------------	---------------------	--

2.9.9 repair

The **repair** command starts Altova FlowForce Server with all triggers and job execution processes disabled, to enable troubleshooting.

```
FlowForceServer repair [options]
```

Note: On Linux systems, use an all-lowercase **flowforceserver** to call the executable.

Example

Here's an example of the **repair** command:

```
FlowForceServer repair  
--datadir=C:\ProgramData\Altova\FlowForceServer2013\data
```

Options

The single option is shown below.

	--datadir=VALUE	VALUE is the path of the database directory.
--	-----------------	--

2.9.10 setdeflang (sdl)

The `setdeflang` command (short form is `sdl`) sets the default language:

```
FlowForceServer setdeflang | sdl LanguageCode
```

Note: On Linux systems, use an all-lowercase `flowforceserver` to call the executable.

Example

Here's an example of the `setdeflang` command:

```
FlowForceServer setdeflang de
```

The command above sets the default language for messages to German.

Supported languages

The table below lists the languages currently supported together with their language codes.

en	English
de	German
es	Spanish
ja	Japanese

2.9.11 start

The **start** command starts Altova FlowForce Server as a service.

```
FlowForceServer start [options]
```

Note: On Linux systems, use an all-lowercase **flowforceserver** to call the executable.

This command is used internally by the startup scripts or by the Windows service installation. It should not be used directly.

Options

The single option is shown below.

	<code>--datadir=VALUE</code>	VALUE is the path of the database directory.
--	------------------------------	--

2.9.12 uninstall

The `uninstall` command uninstalls the service Altova FlowForce Server.

```
FlowForceServer uninstall
```

Note: On Linux systems, use an all-lowercase `flowforceserver` to call the executable.

Example

Here's an example of the `uninstall` command:

```
FlowForceServer uninstall
```


2.9.13 upgradedb

The `upgradedb` command upgrades the database to the latest version.

```
FlowForceServer upgradedb [options]
```

Note: On Linux systems, use an all-lowercase `flowforceserver` to call the executable.

Example

Here's an example of the `upgradedb` command:

```
FlowForceServer upgradedb  
--datadir=C:\ProgramData\Altova\FlowForceServer2013\data
```

The default database is upgraded automatically at installation time, so it is usually not necessary to run this command manually.

Options

The single option is shown below.

	<code>--datadir=VALUE</code>	VALUE is the path of the database directory.
--	------------------------------	--

2.10 Altova MapForce Server

MapForce Server is an enterprise server product that runs on high-speed servers running MS Windows, Linux and Mac OS X operating systems. It operates as a module of FlowForce Server, and is also available as a standalone server product.

MapForce Server is supported on the following operating systems:

- Windows Server 2003, 2008 R2, or newer
- Windows XP with Service Pack 3, Windows 7, Windows 8, or newer
- Linux (CentOS 6, Debian 6, and Ubuntu 12.10, or newer)
- Mac OS X

MapForce Server is available for both 32-bit and 64-bit Windows versions.

Limitations:

- XML Signatures are not supported
- Global resources are not supported via the COM interface
- ODBC and ADO database connections are only supported by Windows. Other operating systems automatically connect via JDBC.

MapForce Server is available as part of the [FlowForce Server](#) installation package and interfaces with FlowForce server allowing you to define jobs, triggers, users, etc.

MapForce Server standalone:

MapForce Server is also available as a standalone product which can be downloaded from the [Altova download](#) page.

The Windows, Linux, and Mac OS X editions of the MapForce Server products also include Altova LicenseServer, which is needed to manage Altova server product licensing.

2.10.1 Installation on Windows

Note:

MapForce Server running under Windows has a minimum requirement: Windows XP with Service Pack 3.

Installing MapForce Server as part of FlowForce Server

The [FlowForce Server](#) installer allows you to install FlowForce Server, LicenseServer, as well as optionally MapForce Server and StyleVision Server. When installing FlowForce without having an Altova LicenseServer installed before make sure that you also select Altova LicenseServer.

MapForce Server standalone

The [MapForce Server standalone](#) installer contains a command-line binary which does not have a graphical user interface as well as Altova LicenseServer.

You can generate a MapForce Server Execution file (.mfx) from the MapForce command line using the /COMPILE command or via the menu item "Files->Compile to MapForce Server Execution File". Such a file can be directly executed on MapForce Server using the 'run' command without needing to use FlowForce Server. This allows you to execute mappings by running MapForce Server Execution Files (*.mfx) on any of the supported operating systems.

Note:

If FlowForce Server is installed at any time, then MapForce Server standalone will integrate itself into FlowForce Server. You will then be able to access MapForce Server through the FlowForce GUI.

Installing Altova LicenseServer

In order for MapForce Server to run, it must be licensed via an Altova LicenseServer on your network. (This applies only for non-Development Editions of MapForce Server.)

Altova LicenseServer will be downloaded and installed by default when you install either FlowForce Server or MapForce Server on Windows systems. If an Altova LicenseServer is already installed on your network, you do not need to install another one. In this case, during the installation process of FlowForce Server or MapForce Server, uncheck the option for installing Altova LicenseServer.

See the sections, [Licensing on Windows](#), [Licensing on Linux](#), or [Initial setup - Mac OS X](#) for more information about how to register and license MapForce Server with Altova LicenseServer.

Users and credentials - FlowForce

Users and user credentials are defined in FlowForce if you downloaded and installed MapForce Server as part of the FlowForce package, please see [Access Control](#).

Users and credentials - MapForce Server standalone

If you installed the MapForce Server standalone installer, which only allows command line access, please ask your Administrator to set up the correct access permissions for you.

File paths in Windows

File paths given in this documentation will not be the same for all operating systems. You should note the following locations:

- MapForce Server stores its configuration file containing the default language setting in the following locations:

Windows XP	C:\Documents and Settings\All Users\Application Data\Altova\MapForceServer2013
Windows Vista, Windows 7/8	C:\ProgramData\Altova\MapForceServer2013

- Application folder: The Application folder is the folder where your Altova application is located. The path to the Application folder is, by default, the following.

Windows XP	C:\Program Files\Altova
Windows Vista, Windows 7/8	C:\Program Files\Altova
32 bit Version on 64-bit OS	C:\Program Files (x86)\Altova

Licensing on Windows

To license MapForce Server on Windows systems, you need to do the following:

- If LicenseServer is not already running as a service, start it as a service. How to do this is described in the [Altova LicenseServer documentation](#).
- Register MapForce Server with LicenseServer. Depending on whether you have installed MapForce Server (i) as part of FlowForce, or (ii) as a standalone product, the method of registering MapForce Server with LicenseServer will be different. Both methods are described below.
- In the configuration page of LicenseServer, assign a license to MapForce Server according to the number of cores on the MapForce Server machine. How to do this is described in the [Altova LicenseServer documentation](#).

Registering MapForce Server from FlowForce

MapForce Server is packaged with FlowForce Server, so when FlowForce Server is registered with an Altova LicenseServer on your network, MapForce Server will automatically also be registered with LicenseServer. How to register FlowForce Server is described in the FlowForce Server documentation.

After the registration, you can go to LicenseServer and assign a license to MapForce Server. How to do this is described in the [Altova LicenseServer documentation](#).

Registering a standalone MapForce Server

If you have installed MapForce Server as a standalone package, you must register it with an Altova LicenseServer on your network and then license it from the Altova LicenseServer. You can register MapForce Server via its command line interface (CLI) by using the [licenseserver](#) command:

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

After successfully registering MapForce Server, you can go to LicenseServer and assign a license to MapForce Server. How to do this is described in the [Altova LicenseServer documentation](#).

Note on cores and licenses

MapForce Server licensing is based on the number of cores available on the MapForce Server machine. The number of cores licensed must be greater than or equal to the number of cores available on the server, whether it's a physical or virtual machine. For example, if a server has eight cores, you must purchase at least an eight-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.

2.10.2 Installation on Linux

Packages

Installation packages are available for:

Distribution	Package extension
Debian 6	.deb
Ubuntu 12.04	.deb
CentOS 6	.rpm

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

Installation

The installation must be done as the root user. If you are logged in as root, leave out the "sudo" prefix when typing the following commands.

Uninstalling old versions

On the Linux command line, you can check which Altova server products are installed with the following command:

```
[Debian Ubuntu]: dpkg --get-selections | grep Altova
[CentOS]:        rpm -qa | grep server
```

If MapForce Server is not installed, go ahead with the installation as documented in the next steps. If MapForce Server is installed and you wish to install a newer version of MapForce Server, uninstall the old version with the command:

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

If you need to uninstall the LicenseServer use:

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

Installing MapForce Server

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

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

Install MapForce Server with the following command:

```
[Debian]: sudo dpkg --install mapforceserver-2013-debian.deb
[Ubuntu]: sudo dpkg --install mapforceserver-2013-ubuntu.deb
```

```
[CentOS]: sudo rpm -ivh mapforceserver-2013-1.x86_64.rpm
```

Installing Altova LicenseServer

In order for MapForce Server to run, it must be licensed via an Altova LicenseServer on your network. Download Altova LicenseServer package from the [Altova website](#) to any directory on the Linux system. Install it just like you did MapForce Server (*see previous step*).

```
[Debian]: sudo dpkg --install licenseserver-1.0-debian.deb
```

```
[Ubuntu]: sudo dpkg --install licenseserver-1.0-ubuntu.deb
```

```
[CentOS]: sudo rpm -ivh licenseserver-1.0-1.x86_64.rpm
```

For information about how to register MapForce Server with Altova LicenseServer and license it, see the section, [Licensing](#).

File paths in Linux

- Application path

Linux	/opt/Altova/MapForceServer2013/bin
-------	------------------------------------

Licensing on Linux

To license MapForce Server on Linux systems (Debian 6, Ubuntu 12.04, CentOS 6), do the following:

1. If LicenseServer is not already running as a service, start it as a service.
2. Register MapForce Server with LicenseServer.
3. In the configuration page of LicenseServer, assign a license to MapForce Server according to the number of cores on the MapForce Server machine. How to do this is described in the [Altova LicenseServer documentation](#).

Note: You must have both Altova MapForce Server and Altova LicenseServer installed. See the section [Installation on Linux](#) for information about installing these packages.

You must have Administrator privileges (root) to be able to register MapForce Server with LicenseServer!

Starting LicenseServer as a service

If LicenseServer is not already running on a different server in your network, LicenseServer must be installed and running as a service. Start LicenseServer as a service with the following command:

```
[Debian]: sudo /etc/init.d/licenseserver start
```

```
[Ubuntu]: sudo initctl start licenseserver
```

```
[CentOS]: sudo initctl start licenseserver
```

(If you need to stop LicenseServer, replace **start** with **stop** in the above command)

Registering MapForce Server

Before licensing MapForce Server from LicenseServer, MapForce Server must be registered with LicenseServer. You can register MapForce Server by using the [licenseserver](#) command of its CLI.

```
sudo /opt/Altova/MapForceServer2013/bin/mapforceserver licenseserver  
[options] Server-Or-IP-Address
```

In the command above, *Server-Or-IP-Address* is the name of the server on which LicenseServer is installed. Notice also that the location of the MapForceServer executable is:

```
/opt/Altova/MapForceServer2013/bin/mapforceserver
```

After successfully registering MapForce Server, you can go to LicenseServer and assign a license to MapForce Server. How to do this is described in the [Altova LicenseServer documentation](#).

Note on cores and licenses

MapForce Server licensing is based on the number of cores available on the MapForce Server machine. The number of cores licensed must be greater than or equal to the number of cores available on the server, whether it's a physical or virtual machine. For example, if a server has eight cores, you must purchase at least an eight-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.

2.10.3 Installation Mac OS X

MapForce Server can be installed on Mac OS X systems (version 10.7 or higher). Since you might need to uninstall a previous version, uninstalling is described first.

Uninstalling old versions of MapForce Server and LicenseServer

In the Applications terminal, right-click the MapForce 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/MapForceServer2013/
```

If you need to uninstall an old version of Altova LicenseServer, first stop the service with the command:

```
sudo launchctl unload /Library/LaunchDaemons/com.altova.LicenserServer11.plist
```

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

Downloading the Mac OS X package

After downloading the Mac OS X package from the [Altova website](#), copy the package to any directory on the Mac OS X system.

Since you will need an [Altova LicenseServer](#) in order to run MapForce Server, you may want to download LicenseServer from the [Altova website](#) at the same time as you download MapForce Server, rather than download it at a later time. The Mac OS X installer file has a `.pkg` file extension.

Installing MapForce Server

In a terminal window, switch to the directory where you have copied the installer file, and double-click it. 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.

The MapForce Server package will be installed in the folder:

```
/usr/local/Altova/MapForceServer2013/
```

Clicking the MapForce Server icon in the Application terminal pops up this onscreen help.

Installing Altova LicenseServer

For MapForce Server to run, it must be licensed via an Altova LicenseServer on your network. On Mac OS X systems, [Altova LicenseServer](#) will need to be installed separately.

Download Altova LicenseServer from the [Altova website](#) and double-click the installer package to start the installation. Follow the on-screen instructions. You will need to accept the license agreement for installation to proceed.

The LicenseServer package will be installed in the folder:

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

For information about how to register MapForce Server with [Altova LicenseServer](#) and license it, see the section, [Initial setup - Mac OS X](#).

Initial setup - Mac OS X

Licensing procedure

To license MapForce Server on Mac OS X systems, do the following:

1. If LicenseServer is not already running as a service, **start** it as a service.
2. **Register** MapForce Server with LicenseServer.
3. In the configuration page of LicenseServer, **assign** a license to MapForce Server machine. How to do this is described in the [Altova LicenseServer documentation](#).

Note: You must have both MapForce Server and [Altova LicenseServer](#) installed and running as services. See the section [Installation on Mac OS X](#) for information about installing these packages.

You must have administrator (root) privileges to be able to register MapForce Server with LicenseServer.

Starting LicenseServer as a service

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

```
sudo launchctl load /Library/LaunchDaemons/com.altova.LicenserServer11.plist
```

If at any time you need to stop LicenseServer, use:

```
sudo launchctl unload /Library/LaunchDaemons/com.altova.LicenserServer11.plist
```

Registering MapForce Server

Before assigning a license to MapForce Server from LicenseServer, MapForce Server must be registered with LicenseServer. You can register MapForce Server by using the [licenseserver](#) command of its CLI.

```
sudo /usr/local/Altova/MapForceServer2013/bin/MapForceServer 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 MapForce Server executable is:

```
/usr/local/Altova/MapForceServer2013/bin
```

After successfully registering MapForce Server, you can go to LicenseServer and assign a license to MapForceServer. How to do this is described in the [Altova LicenseServer documentation](#).

2.10.4 Deploying mappings to Servers

A new command has been introduced in MapForce that lets you deploy a MapForce mapping to FlowForce Server, from where it can then be executed by MapForce Server.

This allows you to define jobs and job triggers, which define what a specific job does, and when it will execute.

Deploying a MapForce mapping to FlowForce:

This is done by opening a MapForce design file in MapForce and selecting the menu option **File | Deploy to FlowForce Server**.

Please see [Deploying mappings to Servers](#) for an example on how this is done.

Generating MapForce Server Execution files:

It is also possible to generate a mapping output file that can be directly executed on MapForce Server without needing to use FlowForce Server.

You can generate MapForce Executable Files in two ways:

- Using the MapForce command line using the `/COMPILE` command
Please see "Command line parameters" of the MapForce documentation for more information.
- Using the menu command **File | Compile to MapForce Server Execution File**.

To run a mapping and to generate its output is easily done by compiling the mapping with MapForce to a MapForce Server Execution file (.mfx) and to run this file with MapForce Server edition.

Run the MapForce Server Execution file:

The created MapForce Server Execution file can be run by MapForce Server using the command line command [run](#).

2.10.5 Command Line Usage

This section describes the command line interface of MapForce Server.

Default location on Windows

The MapForce Server executable (`MapForceServer.exe`) is located by default at:

```
<ProgramFilesFolder>\Altova\MapForceServer2013\bin\MapForceServer.exe
```

Default location on Linux

On Linux systems, the MapForce Server executable (`mapforceserver`) is located by default at

```
/opt/Altova/MapForceServer2013/bin/mapforceserver
```

where the first forward slash indicates the Root directory.

Usage

General command line syntax for `MapForceServer` is:

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

where

<code>--h --help</code>	Displays the help text.
<code>--version</code>	Displays the version of MapForce Server.

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

run	Executes a MapForce Server execution file (.mfx)
licenseserver	Register MapForce Server with the Altova LicenseServer on the local network.
setdeflang <code>sdl</code>	Sets the default language.
help <i>COMMAND</i>	Displays help for a specific command. For example: <code>help run</code>
exportresourcestrings	Exports all application resource strings to an XML file

run

The `run` command executes a MapForce Server Execution file (.mfx). It requires an input mfx file as its argument.

```
MapForceServer run [options] MfxFile [> logfile.log]
```

Note: On Linux systems, use an all-lowercase `mapforceserver` to call the executable.

Example

Here's an example of the `run` command in its simplest form:

```
MapForceServer run ProdTest.mfx
```

To create a **log file** when running MapForce Server, use the redirect operator ">" at the end of the run command, e.g.:

```
MapForceServer run ProdTest.mfx > Mylog.log
```

Options

The options are listed below, in their short forms (first column) and long forms (second column), together with their descriptions. On the command line, one or two dashes can be used for both short and long forms.

<code>--gc</code>	<code>--globalresourceconfig</code>	The name of the global resource configuration. Form: <code>--gc=VALUE</code>
<code>--gr</code>	<code>--globalresourcefile</code>	The path of the global resource definition file. Form: <code>--gr=FILE.</code>
<code>--l</code>	<code>--lang</code>	The language used for displaying messages. Form: <code>--lang=VALUE (en,de,ja,es)</code>
<code>--p</code>	<code>--param</code>	The input parameter you want to pass to the mapping, can be used multiple times. The <code>--param</code> switch must be used before each parameter. Form: <code>--p=NAME:VALUE</code> NAME specifies the name of the input parameter and VALUE its value, e. g. <code>--p=file:out.xml</code> . Note: Use quotes if a name or a value contain space characters, e. g. <code>--p="output file":out.xml</code> <code>--p=company:"Nanonull Inc"</code>

Note for Windows:

Avoid using the end backslash and closing quote on the command line `\` e.g. `"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 (`"c:\My Directory\\"`), or try to avoid using spaces and therefore quotes at all e.g. `c:\MyDirectory`.

licenseserver

The `licenseserver` command specifies the name of the machine on the network running the Altova LicenseServer. Alternatively, you can specify the machine's IP address.

The `licenseserver` command registers MapForceServer with LicenseServer. Once this is done, you can, on LicenseServer, assign a license to MapForceServer. How to do this is described in the LicenseServer documentation.

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

Note: On Linux systems, use an all-lowercase `mapforceserver` to call the executable.

Example

Here's an example of the `licenseserver` command in its simplest form:

```
MapForceServer licenseserver DOC.altova.com
```

The command above specifies that the machine named `DOC.altova.com` is the machine running Altova LicenseServer. If the LicenseServer is running on the user's machine, the following commands would also be valid:

```
MapForceServer licenseserver localhost
MapForceServer licenseserver 127.0.0.1
```

Options

The options are listed below, in their short forms (first column) and long forms (second column), together with their descriptions. On the command line, one or two dashes can be used for both short and long forms.

<code>--j</code>	<code>--json</code>	Prints the result of the registration attempt as a machine-parsable JSON object. Form: <code>--json=true/false</code>
------------------	---------------------	--

setdeflang

The `setdeflang` command (short form is `sdl`) takes:

```
MapForceServer setdeflang | sdl LanguageCode
```

Note: On Linux systems, use an all-lowercase `mapforceserver` to call the executable.

Example

Here's an example of the `setdeflang` command:

```
MapForceServer setdeflang de
```

The command above sets the default language for messages to German.

Supported languages

The table below lists the languages currently supported together with their language codes.

en	English
de	German
es	Spanish
ja	Japanese

help

The `help` command takes a single argument: 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.

```
MapForceServer help Command
```

Note: On Linux systems, use an all-lowercase `mapforceserver` to call the executable.

Example

Here's an example of the `help` command:

```
MapForceServer help run
```

The command above contains one argument: the command `run`, for which help is required. When this command is executed, it will display information about the `run` command.

The --help option

Help information about a command is also available by using the `--help` option with that command. For example, using the `--help` option with the `run` command, as follows:

```
MapForceServer run --help
```

achieves the same result as does using the `help` command with an argument of `generate`:

```
MapForceServer help run
```

In both cases, help information about the `run` command is displayed.

exportresourcestrings

The `exportresourcestrings` command outputs an XML file containing the resource strings of the MapForce Server application. It takes two arguments: (i) the language of the resource strings in the output XML file, and (ii) the path and name of the output XML file. Allowed export languages (with their language codes in parentheses) are: English (`en`), German, (`de`), Spanish (`es`), and Japanese (`ja`).

MapForceServer exportresourcestrings Language XMLOutput

Note: On Linux systems, use an all-lowercase `mapforceserver` to call the executable.

Arguments

The `exportresourcestrings` command takes the following arguments:

Language	Specifies the language of resource strings in the exported XML file. Allowed languages are: en, de, es, ja
XMLOutput	Specifies the location and name of the exported XML file.

Example

Here's an example of the `exportresourcestrings` command:

```
MapForceServer exportresourcestrings en c:\Strings.xml
```

This command creates a file called `Strings.xml` at `c:\` that contains all the resource strings of the MapForce Server application in English.

Creating localized versions of MapForceServer

Create a localized version of MapForce Server as follows:

1. Generate an XML file containing the resource strings by using the `exportresourcestrings` command (see *command syntax above*).
2. Translate the resource strings 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 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\MapForceServer2013\bin` folder. Your DLL file will have a name of the form `MapForceServer2013_lc.dll`. The `_lc` part of the name contains the language code. For example, in `MapForceServer2013_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 MapForce Server app to use. Use the language code that is part of the DLL name as the argument of the `setdeflang` command.

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

2.11 Altova StyleVision Server

StyleVision Server is an implementation of the [StyleVision](#) built-in execution engine. It operates as a module of [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.

Installation and licensing

StyleVision Server is supported on the following operating systems:

- Windows Server 2003, 2008 R2, or newer
- Windows XP with Service Pack 3, Windows 7, Windows 8, or newer
- Linux (CentOS 6, Debian 6, and Ubuntu 12.10, or newer)
- Mac OS X 10.7 or newer

StyleVision Server is available for both 32-bit and 64-bit Windows machines.

2.11.1 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 StyleVision.

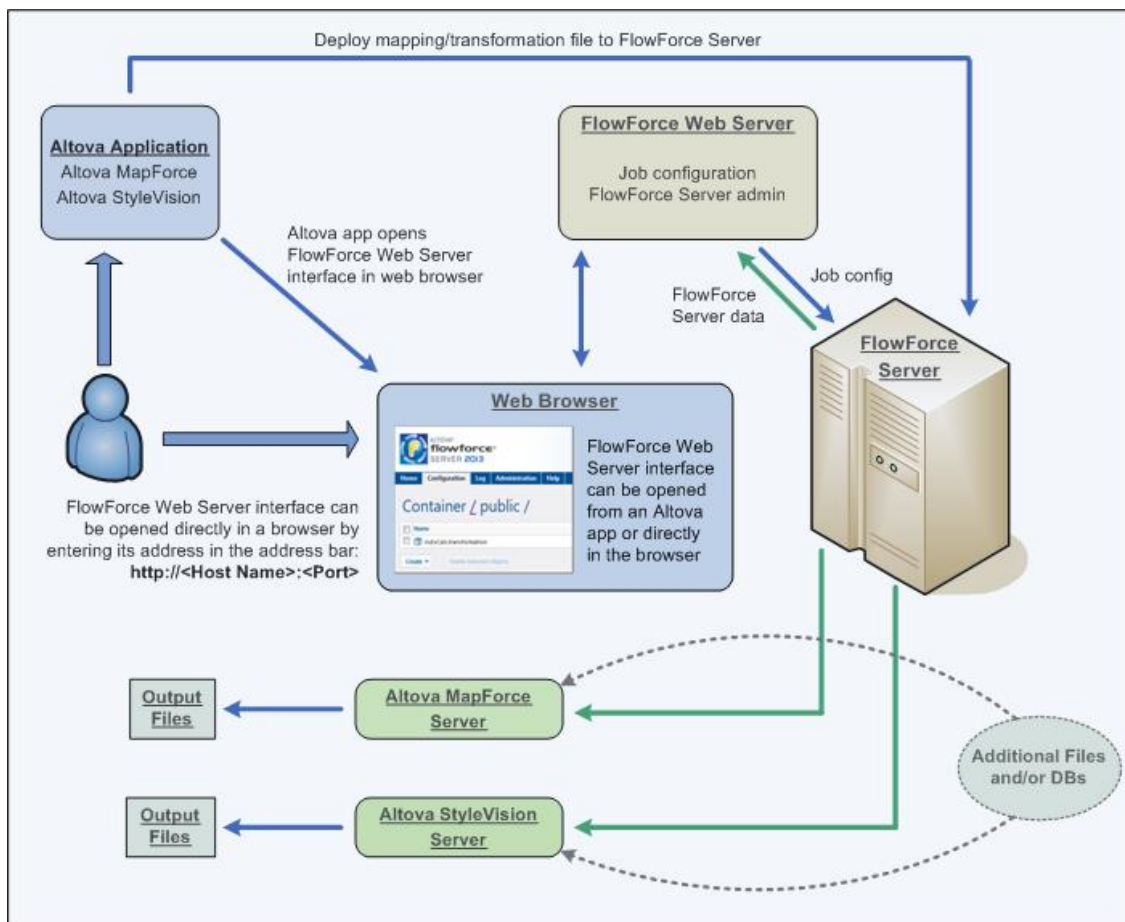
StyleVision Server can be used in two ways:

- As part of the [Altova FlowForce](#) workflow
- As a standalone server product that is accessed via the command line

StyleVision Server 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](#).

StyleVision Server as a standalone server product

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

2.11.2 Installation on Windows

StyleVision Server can be installed on Windows systems in one of two ways:

- As part of the [FlowForce Server](#) installation package. Installing [FlowForce Server](#) (optionally) installs StyleVision Server as a component, but in a separate folder.
- As a separate standalone server product called StyleVision Server.

Both products, [FlowForce Server](#) and StyleVision Server, can be downloaded from the [Altova website](#).

Installing Altova LicenseServer

In order for StyleVision Server to run, it must be licensed via an [Altova LicenseServer](#) on your network.

[Altova LicenseServer](#) will be downloaded and installed by default when you install either FlowForce Server or StyleVision Server on Windows systems. If an [Altova LicenseServer](#) is already installed on your network, you do not need to install another one. In this case, during the installation process of FlowForce Server or StyleVision Server, uncheck the option for installing [Altova LicenseServer](#).

See the sections, [Licensing on Windows](#) and [Licensing on Linux](#), for more information about how to register and license StyleVision Server with [Altova LicenseServer](#).

2.11.3 Installation on Linux

StyleVision Server can be installed on Linux systems (Debian, Ubuntu, CentOS).

Uninstalling old versions of StyleVision Server and LicenseServer

On the Linux command line interface (CLI), you can check whether StyleVision Server or LicenseServer is installed with the following command:

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

If StyleVision Server is not installed, go ahead with the installation as documented in the next steps. 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]:          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]:          sudo rpm -e licenseserver
```

Copying the Linux package

After downloading the Linux package from the [Altova website](#), copy the package 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.

Distribution	Installer extension
Debian	.deb
Ubuntu	.deb
CentOS	.rpm

Installing 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-2013-debian.deb
```

```
[Ubuntu]:  sudo dpkg --install stylevisionserver-2013-ubuntu.deb
[CentOS]:  sudo rpm -ivh stylevisionserver-2013-1.x86_64.rpm
```

The StyleVision Server package will be installed in the folder:

```
/opt/Altova/StyleVisionServer2013
```

Installing Altova LicenseServer

In order for StyleVision Server to run, it must be licensed via an Altova LicenseServer on your network. On Linux systems, [Altova LicenseServer](#) will need to be installed separately. Download Altova LicenseServer from the [Altova website](#) and copy the package to any directory on the Linux system. Install it just like you did StyleVision Server (*see previous step*).

```
[Debian]:  sudo dpkg --install licenseserver-1.0-debian.deb
[Ubuntu]:  sudo dpkg --install licenseserver-1.0-ubuntu.deb
[CentOS]:  sudo rpm -ivh licenseserver-1.0-1.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](#).

2.11.4 Installation on Mac OS X

StyleVision Server can be installed on Mac OS X systems (version 10.7 or higher). Since you might need to uninstall a previous version, uninstalling is described first.

Uninstalling old versions of StyleVision Server and LicenseServer

In the Applications terminal, 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/StyleVisionServer2013/
```

If you need to uninstall an old version of Altova LicenseServer, first stop the service with the following command:

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

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

Downloading the Mac OS X package

After downloading the MacOS X package from the [Altova website](#), copy the package to any directory on the Mac OS X 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. The Mac OS X installer file has a `.pkg` file extension.

Installing StyleVision Server

In a terminal window, switch to the directory where you have copied the installer file, and double-click it. 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.

The StyleVision Server package will be installed in the folder:

```
/usr/local/Altova/StyleVisionServer2013/
```

Clicking the StyleVision Server icon in the Application terminal pops up the onscreen help ([this documentation](#)).

Installing Altova LicenseServer

In order for StyleVision Server to run, it must be licensed via an Altova LicenseServer on your network. On Mac OS X systems, [Altova LicenseServer](#) will need to be installed separately.

Download Altova LicenseServer from the [Altova website](#) and double-click the installer package to start the installation. Follow the onscreen instructions. You will need to accept the license agreement for installation to proceed.

The LicenseServer package will be installed in the folder:

`/usr/local/Altova/LicenseServer`

For information about how to registerStyleVision Server with [Altova LicenseServer](#) and license it, see the section, [Licensing](#).

2.11.5 Licensing on Windows

To license StyleVision Server on Windows systems, you need to do the following:

1. If LicenseServer is not already running as a service, start it as a service. How to do this is described in the [Altova LicenseServer documentation](#).
2. Register StyleVision Server with LicenseServer. Depending on whether you have installed StyleVision Server (i) as part of FlowForce, or (ii) as a standalone product, the method of registering StyleVision Server with LicenseServer will be different. Both methods are described below.
3. In the configuration page of LicenseServer, assign a license to StyleVision Server according to the number of cores on the StyleVision Server machine. How to do this is described in the [Altova LicenseServer documentation](#).

Note: Altova LicenseServer will be installed by default when you install either the Altova FlowForce Server product or Altova StyleVision Server product.

Registering StyleVision Server from FlowForce

StyleVision Server is packaged with [FlowForce Server](#), so when [FlowForce Server](#) is registered with an Altova LicenseServer on your network, StyleVision Server will automatically also be registered with LicenseServer. How to register FlowForce Server is described in the FlowForce Server documentation.

After the registration, you can go to LicenseServer and assign a license to StyleVision Server. How to do this is described in the [Altova LicenseServer documentation](#).

Registering a standalone StyleVision Server

If you have installed StyleVision Server as a standalone package, you must register it with an Altova LicenseServer on your network and then license it from the Altova LicenseServer. You can register StyleVision Server via its command line interface (CLI) by using the [licenseserver command](#):

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

After successfully registering StyleVision Server, you can go to LicenseServer and assign a license to StyleVision Server. How to do this is described in the [Altova LicenseServer documentation](#).

Note on cores and licenses

StyleVision Server licensing is based on the number of cores available on the StyleVision Server machine. The number of cores licensed must be greater than or equal to the number of cores available on the server, whether it's a physical or virtual machine. For example, if a server has eight cores, you must purchase at least an eight-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.

2.11.6 Licensing on Linux

To license StyleVision Server on Linux systems (Debian, Ubuntu, CentOS), do the following:

1. If LicenseServer is not already running as a service, start it as a service.
2. Register StyleVision Server with LicenseServer.
3. In the configuration page of LicenseServer, assign a license to StyleVision Server according to the number of cores on the StyleVision Server machine. How to do this is described in the [Altova LicenseServer documentation](#).

Note: You must have both Altova StyleVision Server and [Altova LicenseServer](#) installed. See the section [Installation on Linux](#) for information about installing these packages.

Note: You must have administrator (root) privileges to be able to register StyleVision Server with LicenseServer.

Starting LicenseServer as a service

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

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

If at any time you need to stop LicenseServer, use:

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

Registering StyleVision Server

Before licensing StyleVision Server from LicenseServer, StyleVision Server must be registered with LicenseServer. You can register StyleVision Server by using the [licenseserver command](#) of its CLI.

```
sudo /opt/Altova/StyleVisionServer2013/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/StyleVisionServer2013/bin
```

After successfully registering StyleVision Server, you can go to LicenseServer and assign a license to StyleVision Server. How to do this is described in the [Altova LicenseServer documentation](#).

Note on cores and licenses

StyleVision Server licensing is based on the number of cores available on the StyleVision Server machine. The number of cores licensed must be greater than or equal to the number of cores available on the server, whether it's a physical or virtual machine. For example, if a server has eight cores, you must purchase at least an eight-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.

2.11.7 Licensing on Mac OS X

To license StyleVision Server on Mac OS X systems, do the following:

1. If LicenseServer is not already running as a service, [start LicenseServer as a service](#).
2. [Register StyleVision Server](#) with LicenseServer.
3. In the configuration page of LicenseServer, assign a license to StyleVision Server according to the number of cores on the StyleVision Server machine. How to do this is described in the [Altova LicenseServer documentation](#).

Note: You must have both StyleVision Server and [Altova LicenseServer](#) installed and running as services. See the section [Installation on Mac OS X](#) for information about installing these packages.

Note: You must have administrator (root) privileges to be able to register StyleVision Server with LicenseServer.

Starting LicenseServer as a service

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

```
sudo launchctl load /Library/LaunchDaemons/com.altova.LicenseServer11.plist
```

If at any time you need to stop LicenseServer, use:

```
sudo launchctl unload /Library/LaunchDaemons/com.altova.LicenseServer11.plist
```

Registering StyleVision Server

Before assigning a license to StyleVision Server from LicenseServer, StyleVision Server must be registered with LicenseServer. You can register StyleVision Server by using the [licenseserver](#) command of its CLI.

```
sudo /usr/local/Altova/StyleVisionServer2013/bin/StyleVisionServer  
licenseserver localhost
```

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

```
/usr/local/Altova/StyleVisionServer2013/bin
```

After successfully registering StyleVision Server, you can go to LicenseServer and assign a license to StyleVision Server. How to do this is described in the [Altova LicenseServer documentation](#).

Note on cores and licenses

StyleVision Server licensing is based on the number of cores available on the StyleVision

Server machine. The number of cores licensed must be greater than or equal to the number of cores available on the server, whether it's a physical or virtual machine. For example, if a server has eight cores, you must purchase at least an eight-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.

2.11.8 Command Line Usage

This section describes the command line interface of StyleVision Server.

Default location on Windows

On Windows systems, the StyleVision Server executable (`StyleVisionServer.exe`) is located by default at:

```
<ProgramFilesFolder>\Altova\StyleVisionServer2013\bin\StyleVisionServer.exe
```

Default location on Linux

On Linux systems, the StyleVision Server executable (`stylevisionserver`) is located by default at

```
/opt/Altova/StyleVisionServer2013/bin/stylevisionserver
```

where the first forward slash indicates the Root directory.

Usage

General command line syntax for `StyleVisionServer` is:

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

where

<code>--h --help</code>	Displays the help text.
<code>--version</code>	Displays the version of StyleVision Server.

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

<code>help COMMAND</code>	Displays help for a specific command. For example: <code>help generate</code>
<code>exportresourcestrings</code>	Exports all application resource strings to an XML file.
<code>generate gen</code>	Generates one or several documents.
<code>licenseserver</code>	Register StyleVision Server with the Altova LicenseServer on the local network.
<code>setdeflang sdl</code>	Sets the default language.
<code>setfopath sfp</code>	Selects an FO processor for subsequent PDF generation.

help

The `help` command takes a single argument: 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.

```
StyleVisionServer help Command
```

Note: On Linux systems, use an all-lowercase `stylevisionserver` to call the executable.

Example

Here's an example of the `help` command:

```
StyleVisionServer help generate
```

The command above contains one argument: the command `generate`, for which help is required. When the example command above is executed, it will display information about the `generate` command.

The --help option

Help information about a command is also available by using the `--help` option with that command. For example, using the `--help` option with the `generate` command, as follows:

```
StyleVisionServer generate --help
```

achieves the same result as does using the `help` command with an argument of `generate`:

```
StyleVisionServer help generate
```

In both cases, help information about the `generate` command is displayed.

exportresourcestrings

The `exportresourcestrings` command outputs an XML file containing the resource strings of the StyleVision Server application. It takes two arguments: (i) the language of the resource strings in the output XML file, and (ii) the path and name of the output XML file. Allowed export languages (with their language codes in parentheses) are: English (`en`), German, (`de`), Spanish (`es`), and Japanese (`ja`).

```
StyleVisionServer exportresourcestrings LanguageCode XMLOutputFile
```

Note: On Linux systems, use an all-lowercase `stylevisionserver` to call the executable.

Arguments

The `exportresourcestrings` command takes the following arguments:

LanguageCode	Specifies the language of resource strings in the exported XML file. Supported languages are: en, de, es, ja
XMLOutputFile	Specifies the location and name of the exported XML file..

Example

Here's an example of the `exportresourcestrings` command:

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

This command creates a file called `Strings.xml` at `c:\` that contains all the resource strings of the StyleVision Server application in German.

Creating localized versions of StyleVision Server

You can create a localized version of StyleVision Server for any language of your choice. Four localized versions (English, German, Spanish, and Japanese) are already available in the `c:\Program Files (x86)\Altova\StyleVisionServer2013\bin` folder, and therefore do not to be created.

Create a localized version as follows:

1. Generate an XML file containing the resource strings by using the `exportresourcestrings` command (*see command syntax above*). The resource strings in this XML file will be one of the four supported languages: English (`en`), German (`de`), Spanish (`es`), or Japanese (`ja`), according to the 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\StyleVisionServer2013\bin` folder. Your DLL file will have a name of the form `StyleVisionServer2013_lc.dll`. The `_lc` part of the name contains the language code. For example, in `StyleVisionServer2013_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 four languages: English, German, Spanish, and Japanese. So you do not need to create a localized version of these languages. To set any of these four languages as the default language, use StyleVision Server's `setdeflang` command.

generate

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

- an input XML file as a mandatory option, and
- an input PXF file as its argument.

It generates one or more output files (HTML, PDF, RTF, and/or DOCX) by transforming the input XML file using the XSLT document contained in the input PXF file. At least one output-creation option must be supplied.

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

Note: On Linux systems, use an all-lowercase `stylevisionserver` to call the executable.

Example

Here's an example of the `generate` command in its simplest form:

```
StyleVisionServer generate --inputxml=Test.xml --html=Test.html Test.pxf
```

The command above contains the mandatory `--inputxml` option, the `InputPXF` argument (`Test.pxf`), and a minimum of one output-creation option (`--html`). If the output-creation option `--html` takes a relative path, as in the example, the output file's location will be relative to the folder in which the PXF file is.

Options

The options are listed below, in their short forms (first column) and long forms (second column), together with their descriptions. On the command line, one or two dashes can be used for both short and long forms.

<code>--xml</code>	<code>--inputxml</code>	The XML file to process. This option is mandatory. Form: <code>--inputxml=<i>Filename</i></code>
<code>--p</code>	<code>--param</code>	Assigns a value to a parameter defined in the PXF file. Form: <code>--param=<i>ParamName</i>:<i>ParamValue</i></code> . The <code>--param</code> switch must be used before each parameter.
<code>--html</code>	<code>--outhtml</code>	The output HTML file to create. Form: <code>--outhtml=<i>Filename</i></code>
<code>--pdf</code>	<code>--outpdf</code>	The output PDF file to create. Form: <code>--outpdf=<i>Filename</i></code>
<code>--rtf</code>	<code>--outrtf</code>	The output RTF file to create. Form: <code>--outrtf=<i>Filename</i></code>
<code>--docx</code>	<code>--outdocx</code>	The output DOCX file to create. Form: <code>--outdocx=<i>Filename</i></code>
<code>--v</code>	<code>--verbose</code>	The display of all messages can be turned on or off, respectively, with <code>--verbose=true</code> or <code>--verbose=false</code> . The default value is <code>false</code> if the option is not provided, but <code>true</code> if the option is provided without a value.
<code>--l</code>	<code>--lang</code>	The language used for displaying messages. Form: <code>--lang=<i>languagecode</i></code> Languages supported on installation: en, de, es, ja
<code>--h</code>	<code>--help</code>	Displays the help text for the <code>generate</code> command.

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

licenseserver

The `licenseserver` command specifies the name of the machine on the network running the Altova LicenseServer. Alternatively, you can specify the machine's IP address.

The `licenseserver` command registers StyleVision Server with LicenseServer. To do this successfully, the two servers must be connected on the network and LicenseServer must be running. 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, on LicenseServer, assign a license to StyleVision Server. How to do this is described in the [LicenseServer documentation](#).

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

Note: On Linux systems, use an all-lowercase `stylevisionserver` to call the executable.

Note: You must have administrator (root) privileges to be able to register StyleVision Server with LicenseServer.

Example

Here's an example of the `licenseserver` command in its simplest form:

```
StyleVisionServer licenseserver DOC.altova.com
```

The command above specifies that the machine named `DOC.altova.com` is the machine running Altova LicenseServer. If the LicenseServer is running on the user's machine, the following commands would also be valid:

```
StyleVisionServer licenseserver localhost
StyleVisionServer licenseserver 127.0.0.1
```

Options

The options are listed below, in their short forms (first column) and long forms (second column), together with their descriptions. On the command line, one or two dashes can be used for both short and long forms.

<code>--j</code>	<code>--json</code>	Prints the result of the registration attempt as a machine-parsable JSON object. Form: <code>--json=true/false</code>
------------------	---------------------	---

setdeflang

The `setdeflang` command (short form is `sd1`) sets the default language of StyleVision Server. It takes a mandatory `LanguageCode` argument.

```
StyleVisionServer setdeflang | sd1 LanguageCode
```

Note: On Linux systems, use an all-lowercase `stylevisionserver` to call the executable.

Example

Here's an example of the `setdeflang` command:

```
StyleVisionServer setdeflang DE
```

The command above sets the default language for messages to German.

Supported languages

The table below lists the languages currently supported together with their language codes.

EN	English
DE	German
ES	Spanish
JA	Japanese

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. If you wish to use another Apache FOP processor instance, 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 `<ProgramFilesFolder>\Altova\`; on Linux systems in a descendant folder of `<ProgramFilesFolder>\Altova\StyleVisionServer2013`.

```
StyleVisionServer setfopath | sfp [options] --path=Path | --pa=Path
```

Note: On Linux systems, use an all-lowercase `stylevisionserver` to call the executable.

Example

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 --path=C:\FOP\FOP.exe  
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.exe` 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 and using transformation files contained in the PXF file. The FO processor specified in the previous command is used for generating the PDF.

2.12 RaptorXML Server

Altova RaptorXML Server (hereafter also called RaptorXML for short) is Altova's third-generation, super-fast XML and XBRL processor. It has been built to be optimized for the latest standards and parallel computing environments. Designed to be highly cross-platform capable, the engine takes advantage of today's ubiquitous multi-core computers to deliver lightning fast processing of XML and XBRL data.

Note: XBRL processing is available only in RaptorXML+XBRL Server, not in RaptorXML Server or RaptorXML Development Edition.

RaptorXML is available in three editions:

- RaptorXML Server is a very fast XML processing engine with support for XML, XML Schema, XSLT, XPath, XQuery and is integrated into the FlowForce Server package.
- RaptorXML+XBRL Server supports all the features of RaptorXML Server with the additional capability of processing and validating the XBRL family of standards. This installer needs to be installed from the Altova website.
- RaptorXML **Development Edition** is the limited edition of RaptorXML Server. Like RaptorXML(+XBRL) Server, it can be used for XML, XSLT, and XQuery validations and transformations, but it has limited functionality in comparison with RaptorXML(+XBRL) Server. This installer needs to be installed from the Altova website.

The Development Edition limitations are as follows:

- Only one instance of its binary is allowed to run at a given time on a given machine.
- Supported only on Windows workstations (not on Windows Server, Linux, or Mac OS operating systems).
- Can only be run from the command line (also DoTransform.bat)
- No support for multiple file processing.
- No XBRL or charts support.

RaptorXML Server limitations:

- XML Signatures are not supported
- Global resources are not supported via the COM interface
- ODBC and ADO database connections are only supported by Windows. Other operating systems automatically connect via JDBC

Downloading

RaptorXML Server is part of the FlowForce Server installation package, please see [Getting started](#) for more information. All other editions need to be downloaded from the [Altova website](#).

Licensing

Please see the [Getting started](#) page on how to register and license RaptorXML Server.

Usage

Please see [Using RaptorXML Server to validate a document](#) in the tutorial section on how to use RaptorXML with FlowForce Server.

Please see [RaptorXML Commands](#) for an explanation of all the RaptorXML functions available to a FlowForce execution step.

2.12.1 RaptorXML Commands

The following RaptorXML functions are available in FlowForce Server to be used as functions in a job execution step.

The XBRL functions are only available (from the RaptorXMLXBRL container) if you have downloaded and installed/registered RaptorXML (+XBRL) Server from the Altova website.

The parameter names of the FlowForce functions approximate the command line parameter names of the RaptorXML command line interface. Explanations of each of the parameters is available by clicking the supplied links.

RaptorXML functions:

[valany](#)

[valdtd](#)

[valxml-withdtd](#)

[valxml-withxsd](#)

[valxquery](#)

[valxsd](#)

[valxslt](#)

[wfany](#)

[wfdtd](#)

[wfxml](#)

[xquery](#)

[xslt](#)

RaptorXML (+XBRL) functions, only if RaptorXML (+XBRL) edition has been installed:

[valxbrl](#)

[valxbrltaxonomy](#)

2.12.2 Command Line Interface (CLI)

The RaptorXML executable for use with the command line interface (CLI) is located by default at:

Usage

The command line syntax is:

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

RaptorXML	Calls the application.
--h --help	Displays the help text.
--version	Displays the application's version number.
<command>	The command to execute. See list below. Each command is described in detail, with its options and arguments, in sub-sections of this section.
[options]	The options of a command. They are listed with their respective commands and are described in detail in the Options section.
[arguments]	The argument/s of a command. They are listed and described with their respective commands.

CLI commands

The available CLI commands are listed below, organized by functionality. They are explained in detail in the sub-sections of this section. (Note that some validation commands appear in more than one group in the list below.)

XML, DTD, XSD Validation Commands

The XML validation commands can be used to validate the following types of document:

- **XML:** Validates XML instance documents against a DTD ([valxml-withdtd | xml](#)) or an XML Schema 1.0/1.1 ([valxml-withxsd | xsi](#)).
- **DTD:** Checks that a DTD is well-formed and contains no error ([valdtd | dtd](#)).
- **XSD:** Validates a W3C XML Schema (XSD) document according to rules of the XML Schema specification ([valxsd | xsd](#)).

XML validation commands are described in detail in the sub-sections of this section:

valxml-withdtd xml	Validates an XML instance document against a DTD.
valxml-withxsd xsi	Validates an XML instance document against an XML Schema.
valdtd dtd	Validates a DTD document.
valxsd xsd	Validates a W3C XML Schema (XSD) document.

valany	Validates any one XML, DTD or XSD document. Note that this command is also used to validate XBRL (instance or taxonomy), XSLT , or XQuery , documents; the type of document submitted is detected automatically.
------------------------	--

Note: XBRL instance, XBRL taxonomy, XSLT and XQuery documents can also be validated. These validation commands are described in their respective sections: [XBRL Validation Commands](#), [XSLT Commands](#), and [XQuery Commands](#).

valxml-withdtd (xml)

The `valxml-withdtd` | `xml` command validates against a DTD.

```
valxml-withdtd | xml [options] InputFile
```

The *InputFile* argument is the XML document to validate. If a reference to a DTD exists in the XML document, the `--dtd` option is not required.

Examples

- `valxml-withdtd --dtd=c:\MyDTD.dtd c:\Test.xml`
- `xml c:\Test.xml`
- `xml --verbose=true c:\Test.xml`

Options

To go to an option's description, click the option or its group header.

- Boolean option values are set to `true` if the option is specified without a value.
- The values of all options can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required.

[Options specific to the 'valxml-withdtd' command](#)

[--dtd=FILE](#)

[--namespaces=true|false](#)

[Catalog options](#)

[--catalog=FILE](#)

[--user-catalog=FILE](#)

[Global resource options](#)

[--enable-globalresources=true|false](#)

[--gr, --globalresourcefile=FILE](#)

[--qc, --globalresourceconfig=VALUE](#)

[Error options](#)

[--error-limit=N|unlimited](#)

[--error-format=text|shortxml|longxml](#)

Message options

[--log-output=FILE](#)

[--verbose=true|false](#)

Help and version options

[--h, --help](#)

[--version](#)

valxml-withxsd (xsi)

The `valxml-withxsd | xsi` command validates according to the W3C XML Schema Definition Language (XSD) 1.0 and 1.1 specifications.

```
valxml-withxsd | xsi [options] InputFile
```

The *InputFile* argument is the XML document to validate. The [--schemalocation-hints=true|false](#) indicates whether the XSD reference in the XML document is to be used or not, with the default being `true` (the location is used). The [--xsd=FILE](#) option specifies the schema/s to use.

Note: If using the `--script` option to run Python scripts, make sure to also specify `--streaming=false`.

Examples

- `valxml-withxsd --schemalocation-hints=false --xsd=c:\MyXSD.xsd c:\HasNoXSDDef.xml`
 - `xsi c:\HasXSDDef.xml`
-

Options

To go to an option's description, click the option or its group header.

- Boolean option values are set to `true` if the option is specified without a value.
- The values of all options can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required.

Options specific to the 'valxml-withxsd' command

[--assessment-mode=skip|lax|strict](#)

XML Schema options

[--schemalocation-hints=load-by-schemalocation|load-by-namespace|load-combining-both|ignore](#)

```
--schema-imports=load-by-schemalocation|  
load-preferring-schemalocation|  
load-by-namespace|  
load-combining-both|  
license-namespace-only  
--schema-mapping=prefer-schemalocation|prefer-namespace  
--xsd=FILE  
--xsd-version=1.0|1.1|detect
```

XML instance options

```
--xinclude=true|false  
--xml-mode=wf|id|valid
```

Catalog options

```
--catalog=FILE  
--user-catalog=FILE
```

Global resource options

```
--enable-globalresources=true|false  
--gr, --globalresourcefile=FILE  
--gc, --globalresourceconfig=VALUE
```

Error options

```
--error-limit=N|unlimited  
--error-format=text|shortxml|longxml
```

Message options

```
--log-output=FILE  
--verbose=true|false
```

Help and version options

```
--h, --help  
--version
```

valtdtd (dtd)

The `valtdtd` | `dtd` command validates according to the XML 1.0 or XML 1.1 specification.

```
valtdtd | dtd [options] InputFile
```

The *InputFile* argument is the DTD document to validate.

Examples

- `valtdtd c:\Test.dtd`
- `dtd --verbose=true c:\Test.dtd`

Options

To go to an option's description, click the option or its group header.

- Boolean option values are set to `true` if the option is specified without a value.
- The values of all options can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required.

Catalog options

[`--catalog=FILE`](#)
[`--user-catalog=FILE`](#)

Global resource options

[`--enable-globalresources=true|false`](#)
[`--gr, --globalresourcefile=FILE`](#)
[`--gc, --globalresourceconfig=VALUE`](#)

Error options

[`--error-limit=N|unlimited`](#)
[`--error-format=text|shortxml|longxml`](#)

Message options

[`--log-output=FILE`](#)
[`--verbose=true|false`](#)

Help and version options

[`--h, --help`](#)
[`--version`](#)

`valxsd (xsd)`

The `valxsd` | `xsd` command validates according to the W3C XML Schema Definition Language (XSD) 1.0 or 1.1 specification. Note that it is the schema itself that is validated against the XML Schema specification, not an XML instance document against an XML Schema.

```
valxsd | xsd [options] InputFile
```

The *InputFile* argument is the XML Schema document to validate. The [`--xsd-version=1.0|1.1|detect`](#) option specifies the XSD version to validate against, with the default being `1.0`.

Examples

- `valxsd c:\Test.xsd`
- `xsd --verbose=true c:\Test.xsd`

Options

To go to an option's description, click the option or its group header.

- Boolean option values are set to `true` if the option is specified without a value.

- The values of all options can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required.

XML Schema options

[--schemalocation-hints=load-by-schemalocation|
load-by-namespace|
load-combining-both|
ignore](#)
[--schema-imports=load-by-schemalocation|
load-prefering-schemalocation|
load-by-namespace|
load-combining-both|
license-namespace-only](#)
[--schema-mapping=prefer-schemalocation|prefer-namespace](#)
[--xsd-version=1.0|1.1|detect](#)

Catalog options

[--catalog=FILE](#)
[--user-catalog=FILE](#)

Global resource options

[--enable-globalresources=true|false](#)
[--gr, --globalresourcefile=FILE](#)
[--gc, --globalresourceconfig=VALUE](#)

XML instance options

[--xinclude=true|false](#)
[--xml-mode=wf|id|valid](#)

Error options

[--error-limit=N|unlimited](#)
[--error-format=text|shortxml|longxml](#)

Message options

[--log-output=FILE](#)
[--verbose=true|false](#)

Help and version options

[--h, --help](#)
[--version](#)

valany

The **valany** command validates an XML, DTD, or XML Schema document according to the respective specification/s. The type of document is detected automatically.

valany [**options**] *InputFile*

The *InputFile* argument is the document to validate. Note that only one document can be submitted as the argument of the command. The type of the submitted document is detected automatically.

Examples

- `valany c:\Test.xml`
 - `valany --errorformat=text c:\Test.xml`
-

Options

To go to an option's description, click the option or its group header.

- Boolean option values are set to `true` if the option is specified without a value.
- The values of all options can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required.

XML Schema options

[`--schemalocation-hints=load-by-schemalocation|load-by-namespace|load-combining-both|ignore`](#)
[`--schema-imports=load-by-schemalocation|load-preferred-schemalocation|load-by-namespace|load-combining-both|license-namespace-only`](#)
[`--schema-mapping=prefer-schemalocation|prefer-namespace`](#)

Catalog options

[`--catalog=FILE`](#)
[`--user-catalog=FILE`](#)

Global resource options

[`--enable-globalresources=true|false`](#)
[`--gr, --globalresourcefile=FILE`](#)
[`--gc, --globalresourceconfig=VALUE`](#)

Error options

[`--error-limit=N|unlimited`](#)
[`--error-format=text|shortxml|longxml`](#)

Message options

[`--log-output=FILE`](#)
[`--verbose=true|false`](#)

Help and version options

[`--h, --help`](#)
[`--version`](#)

Well-formedness Check Commands

The well-formedness check commands can be used to check the well-formedness of XML documents and DTDs. These commands are listed below and described in detail in the

sub-sections of this section:

wfxml	Checks the well-formedness of XML documents.
wfdtd	Checks the well-formedness of DTDs.
wfany	Checks the well-formedness of an XML document or DTD. Type is detected automatically.

wfxml

The **wfxml** command checks for well-formedness according to the XML 1.0 or XML 1.1 specification.

```
wfxml [options] InputFile
```

The *InputFile* argument is the XML document to check for well-formedness.

Examples

- `wfxml c:\Test.xml`
- `wfxml --verbose=true c:\Test.xml`

Options

To go to an option's description, click the option or its group header.

- Boolean option values are set to `true` if the option is specified without a value.
- The values of all options can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required.

[Catalog options](#)

[--catalog=FILE](#)

[--user-catalog=FILE](#)

[Global resource options](#)

[--enable-globalresources=true|false](#)

[--gr, --globalresourcefile=FILE](#)

[--gc, --globalresourceconfig=VALUE](#)

[Error options](#)

[--error-limit=N|unlimited](#)

[--error-format=text|shortxml|longxml](#)

[Message options](#)

[--log-output=FILE](#)

[--verbose=true|false](#)

[Help and version options](#)

[--h, --help](#)

[--version](#)

wfdtd

The **wfdtd** command checks for well-formedness according to the XML 1.0 or XML 1.1 specification.

```
wfdtd [options] InputFile
```

The *InputFile* argument is the DTD document to check for well-formedness.

Examples

- `wfdtd c:\Test.dtd`
- `wfdtd --verbose=true c:\Test.dtd`

Options

To go to an option's description, click the option or its group header.

- Boolean option values are set to `true` if the option is specified without a value.
- The values of all options can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required.

Catalog options

[--catalog=FILE](#)

[--user-catalog=FILE](#)

Global resource options

[--enable-globalresources=true|false](#)

[--gr, --globalresourcefile=FILE](#)

[--gc, --globalresourceconfig=VALUE](#)

Error options

[--error-limit=N|unlimited](#)

[--error-format=text|shortxml|longxml](#)

Message options

[--log-output=FILE](#)

[--verbose=true|false](#)

Help and version options

[--h, --help](#)

[--version](#)

wfany

The **wfany** command checks an XML or DTD document for well-formedness according to the respective specification/s. The type of document is detected automatically.

wfany [*options*] *InputFile*

The *InputFile* argument is the document to check for well-formedness. Note that only one document can be submitted as the argument of the command. The type of the submitted document is detected automatically.

Examples

- `wfany c:\Test.xml`
- `wfany --errorformat=text c:\Test.xml`

Options

To go to an option's description, click the option or its group header.

- Boolean option values are set to `true` if the option is specified without a value.
- The values of all options can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required.

Catalog options

[--catalog=FILE](#)

[--user-catalog=FILE](#)

Global resource options

[--enable-globalresources=true|false](#)

[--gr, --globalresourcefile=FILE](#)

[--gc, --globalresourceconfig=VALUE](#)

Error options

[--error-limit=N|unlimited](#)

[--error-format=text|shortxml|longxml](#)

Message options

[--log-output=FILE](#)

[--verbose=true|false](#)

Help and version options

[--h, --help](#)

[--version](#)

XBRL Validation Commands

The XBRL validation commands can be used to validate XBRL instance documents and XBRL taxonomies according to the XBRL 2.1, Dimensions 1.0 and Formula 1.0 specifications. The

available commands are listed below and described in detail in the sub-sections of this section:

valxbrl xbrl	Validates an XBRL instance document (.xbrl extension).
valxbrltaxonomy dts	Validates an XBRL taxonomy (schema) document (.xsd extension).
valany	Validates any one XBRL (instance or taxonomy) document. Note that this command is also used to validate XML, DTD, XSD, XSLT, or XQuery documents; the type of document submitted is detected automatically.

valxbrl (xbrl)

The **valxbrl | xbrl** command validates one or more XBRL instance documents according to the XBRL 2.1, Dimensions 1.0 and Formula 1.0 specifications.

```
raptorxmlxbrl valxbrl | xbrl [options] InputFile
```

The *InputFile* argument is the XBRL instance document to validate. To validate multiple documents, either: (i) list the files to be validated on the CLI, with each file separated from the next by a space; or (ii) list the files to be validated in a text file (.txt file), with one filename per line, and supply this text file as the *InputFile* argument together with the [--listfile](#) option set to `true` (see the *Options list below*).

Note: The XBRL instance document must not be nested in another XML document and must have the `xbrl` element as its root element.

```
<xbrl xmlns="http://www.xbrl.org/2003/instance"> ... </xbrl>
```

Examples

- **raptorxmlxbrl** valxbrl c:\Test.xbrl
- **raptorxmlxbrl** xbrl --formula-execution=true --formula-output=c:\FormulaOutput.xml c:\Test.xbrl
- **raptorxmlxbrl** xbrl --formula-execution --assertions-output=c:\AssertionsOutput.xml c:\Test.xbrl
- **raptorxmlxbrl** xbrl --formula-execution --formula-output=c:\FormulaOutput.xml --assertions-output=c:\AssertionsOutput.xml c:\Test.xbrl

Options

To go to an option's description, click the option or its group header.

- Boolean option values are set to `true` if the option is specified without a value.
- The values of all options can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required.

[XBRL schema options](#)

[--dimensions=true|false](#)
[--formula=true|false](#)
[--preload-formula-schemas=true|false](#)
[--preload-xbrl-schemas=true|false](#)
[--schemalocation-hints=true|false](#)

XBRL evaluation options

[--assertions-output-format=json|xml](#)
[--assertions-output=FILE](#)
[--formula-execution=true|false](#)
[--formula-output=FILE](#)
[--formula-parameters-file=FILE](#)
[--formula-parameters=JSON-ARRAY](#)
[--validate-dts-only=true|false](#)
[--xinclud=true|false](#)

Processing options

[--listfile=true|false](#)
[--script=FILE](#)

XML Schema options

[--schemalocation-hints=load-by-schemalocation|](#)
[load-by-namespace|](#)
[load-combining-both|](#)
[ignore](#)
[--schema-imports=load-by-schemalocation|](#)
[load-preferring-schemalocation|](#)
[load-by-namespace|](#)
[load-combining-both|](#)
[license-namespace-only](#)
[--schema-mapping=prefer-schemalocation|prefer-namespace](#)

Catalog options

[--catalog=FILE](#)
[--user-catalog=FILE](#)

Global resource options

[--enable-globalresources=true|false](#)
[--gr, --globalresourcefile=FILE](#)
[--gc, --globalresourceconfig=VALUE](#)

ZIP file options

[--recurse=true|false](#)

Error options

[--error-limit=N|unlimited](#)
[--error-format=text|shortxml|longxml](#)

Message options

[--log-output=FILE](#)
[--verbose=true|false](#)

Help and version options

[--h, --help](#)
[--version](#)

valxbrltaxonomy (dts)

The **valxbrltaxonomy** | **dts** command validates one or more XBRL taxonomies (schemas) according to the XBRL 2.1, Dimensions 1.0 and Formula 1.0 specifications.

```
raptorxmlxbrl valxbrltaxonomy | dts [options] InputFile
```

The *InputFile* argument is the XBRL taxonomy to validate. To validate multiple documents, either: (i) list the files to be validated on the CLI, with each file separated from the next by a space; or (ii) list the files to be validated in a text file (.txt file), with one filename per line, and supply this text file as the *InputFile* argument together with the [--listfile](#) option set to `true` (see the *Options list below*).

Examples

- **raptorxmlxbrl** valxbrltaxonomy c:\Test.xsd
 - **raptorxmlxbrl** dts --listfile c:\FileList.txt
-

Options

To go to an option's description, click the option or its group header.

- Boolean option values are set to `true` if the option is specified without a value.
- The values of all options can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required.

XBRL schema options

[--dimensions=true|false](#)
[--formula=true|false](#)
[--preload-formula-schemas=true|false](#)
[--preload-xbrl-schemas=true|false](#)

Processing options

[--listfile=true|false](#)
[--script=FILE](#)
[--xinclude=true|false](#)

XML Schema options

[--schemalocation-hints=load-by-schemalocation|load-by-namespace|load-combining-both|ignore](#)

[--schema-imports=load-by-schemalocation|
load-preferring-schemalocation|
load-by-namespace|
load-combining-both|
license-namespace-only](#)
[--schema-mapping=prefer-schemalocation|prefer-namespace](#)

Catalog options

[--catalog=FILE](#)
[--user-catalog=FILE](#)

Global resource options

[--enable-globalresources=true|false](#)
[--gr, --globalresourcefile=FILE](#)
[--gc, --globalresourceconfig=VALUE](#)

ZIP file options

[--recurse=true|false](#)

Error options

[--error-limit=N|unlimited](#)
[--error-format=text|shortxml|longxml](#)

Message options

[--log-output=FILE](#)
[--verbose=true|false](#)

Help and version options

[--h, --help](#)
[--version](#)

valany

The **valany** command validates an XBRL instance document or XBRL taxonomy according to the XBRL 2.1, Dimensions 1.0 and Formula 1.0 specifications. The type of document is detected automatically.

```
raptorxmlxbrl valany [options] InputFile
```

The *InputFile* argument is the document to validate. Note that only one document can be submitted as the argument of the command. The type of the submitted document is detected automatically.

Examples

- **raptorxmlxbrl valany c:\Test.xsd**
- **raptorxmlxbrl valany --errorformat=text c:\Test.xbrl**

Options

To go to an option's description, click the option or its group header.

- Boolean option values are set to `true` if the option is specified without a value.
- The values of all options can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required.

XML Schema options

[--schemalocation-hints=load-by-schemalocation|
load-by-namespace|
load-combining-both|
ignore](#)
[--schema-imports=load-by-schemalocation|
load-prefering-schemalocation|
load-by-namespace|
load-combining-both|
license-namespace-only](#)
[--schema-mapping=prefer-schemalocation|prefer-namespace](#)

Catalog options

[--catalog=FILE](#)
[--user-catalog=FILE](#)

Global resource options

[--enable-globalresources=true|false](#)
[--gr, --globalresourcefile=FILE](#)
[--gc, --globalresourceconfig=VALUE](#)

ZIP file options

[--recurse=true|false](#)

Error options

[--error-limit=N|unlimited](#)
[--error-format=text|shortxml|longxml](#)

Message options

[--log-output=FILE](#)
[--verbose=true|false](#)

Help and version options

[--h, --help](#)
[--version](#)

XSLT Commands

The XSLT commands are:

- [xslt](#): for transforming XML documents with an XSLT document
- [valxslt](#): for validating XSLT documents

The arguments and options for each command are listed in the sub-sections, [xslt](#) and [valxslt](#).

xslt

The **xslt** command takes an XSLT file as its single argument and uses it to transform an input XML file to produce an output file. The input and output files are specified as [options](#).

```
xslt [options] XSLT-File
```

The *XSLT-File* argument is the path and name of the XSLT file to use for the transformation. An input XML file ([--input](#)) or a named template entry point ([--template-entry-point](#)) is required. If no [--output](#) option is specified, output is written to standard output. You can use XSLT 1.0, 2.0, or 3.0. By default XSLT 3.0 is used.

Examples

- `xslt --input=c:\Test.xml --output=c:\Output.xml c:\Test.xslt`
- `xslt --template-entry-point=StartTemplate --output=c:\Output.xml c:\Test.xslt`
- `xslt --input=c:\Test.xml --output=c:\Output.xml --param date=//node/@att1 --p=title:'stringwithoutspace' --param=title:''string with spaces'' --p=amount:456 c:\Test.xslt`

Options

To go to an option's description, click the option or its group header.

- Boolean option values are set to `true` if the option is specified without a value.
- The values of all options can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required.

Options specific to the 'xslt' command

[--indent-characters=VALUE](#)
[--input=FILE](#)
[--output=FILE](#)
[--p, --param=KEY:VALUE](#)
[--streaming-serialization-enabled=true|false](#)

Options common to the 'xslt' and 'valxslt' commands

[--chartext-disable=true|false](#)
[--dotnetext-disable=true|false](#)
[--javaext-barcode-location=FILE](#)
[--javaext-disable=true|false](#)
[--template-entry-point=VALUE](#)
[--template-mode=VALUE](#)
[--xslt-version=1|2|3](#)

Catalog options

[--catalog=FILE](#)
[--user-catalog=FILE](#)

Global resource options

[--enable-globalresources=true|false](#)
[--gr, --globalresourcefile=FILE](#)
[--gc, --globalresourceconfig=VALUE](#)

Error options

[--error-limit=N|unlimited](#)

Message options

[--verbose=true|false](#)

XML instance options

[--xinclude=true|false](#)
[--xml-mode=wf|id|valid](#)

XML Schema options

[--schemalocation-hints=load-by-schemalocation|](#)
[load-by-namespace|](#)
[load-combining-both|](#)
[ignore](#)
[--schema-imports=load-by-schemalocation|](#)
[load-prefering-schemalocation|](#)
[load-by-namespace|](#)
[load-combining-both|](#)
[license-namespace-only](#)
[--schema-mapping=prefer-schemalocation|prefer-namespace](#)
[--xsd-version=1.0|1.1|detect](#)

Help and version options

[--h, --help](#)
[--version](#)

valxslt

The `valxslt` command takes an XSLT file as its single argument and validates it.

```
valxslt [options] XSLT-File
```

The *XSLT-File* argument is the path and name of the XSLT file to be validated. Validation can be according to the XSLT 1.0, 2.0, or 3.0 specification. By default XSLT 3.0 is the specification used.

Examples

- `valxslt c:\Test.xslt`
 - `valxslt --xslt-version=2 c:\Test.xslt`
-

Options

To go to an option's description, click the option or its group header.

- Boolean option values are set to `true` if the option is specified without a value.
- The values of all options can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required.

Options common to the 'xslt' and 'valxslt' commands

[--chartext-disable=true|false](#)
[--dotnetext-disable=true|false](#)
[--javaext-barcode-location=FILE](#)
[--javaext-disable=true|false](#)
[--template-entry-point=VALUE](#)
[--template-mode=VALUE](#)
[--xslt-version=1|2|3](#)

Catalog options

[--catalog=FILE](#)
[--user-catalog=FILE](#)

Global resource options

[--enable-globalresources=true|false](#)
[--gr, --globalresourcefile=FILE](#)
[--gc, --globalresourceconfig=VALUE](#)

Error options

[--error-limit=N|unlimited](#)

Message options

[--verbose=true|false](#)

XML instance options

[--xinclude=true|false](#)
[--xml-mode=wf|id|valid](#)

XML Schema options

[--schemalocation-hints=load-by-schemalocation|](#)
[load-by-namespace|](#)
[load-combining-both|](#)
[ignore](#)
[--schema-imports=load-by-schemalocation|](#)
[load-preferring-schemalocation|](#)
[load-by-namespace|](#)
[load-combining-both|](#)
[license-namespace-only](#)
[--schema-mapping=prefer-schemalocation|prefer-namespace](#)
[--xsd-version=1.0|1.1|detect](#)

Help and version options

[--h, --help](#)
[--version](#)

XQuery Commands

The XQuery commands are:

- [xquery](#): for executing XQuery documents, optionally with an input document
- [valxquery](#): for validating XQuery documents

The arguments and options for each command are listed in the sub-sections, [xquery](#) and [valxquery](#).

xquery

The **xquery** command takes an XQuery file as its single argument and executes it with an optional input file to produce an output file. The input and output files are specified as options.

```
xquery [options] XQuery-File
```

The argument *xquery-File* is the path and name of the XQuery file to be executed.

Examples

- `xquery --output=c:\Output.xml c:\TestQuery.xq`
 - `xquery --input=c:\Input.xml --output=c:\Output.xml --var company=Altova -var date=2006-01-01 c:\TestQuery.xq`
 - `xquery --input=c:\Input.xml --output=c:\Output.xml --xparam source="doc('c:\test\books.xml')//book "`
 - `xquery --output=c:\Output.xml --omit-xml-declaration=false --output-encoding=ASCII c:\TestQuery.xq`
-

Options

To go to an option's description, click the option or its group header.

- Boolean option values are set to `true` if the option is specified without a value.
- The values of all options can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required.

[Options specific to the 'xquery' command](#)

[--indent-characters=VALUE](#)
[--input=FILE](#)
[--output=FILE](#)
[--output-encoding=VALUE](#)
[--output-indent=true|false](#)
[--output-method=xml|html|xhtml|text](#)
[--p, --param=KEY:VALUE](#)

[Options common to the 'xquery' and 'valxquery' commands](#)

[--omit-xml-declaration=true|false](#)
[--xquery-version=1|3](#)

Catalog options

[--catalog=FILE](#)
[--user-catalog=FILE](#)

Global resource options

[--enable-globalresources=true|false](#)
[--gr, --globalresourcefile=FILE](#)
[--gc, --globalresourceconfig=VALUE](#)

Error options

[--error-limit=N|unlimited](#)

Message options

[--verbose=true|false](#)

XML instance options

[--xinclude=true|false](#)
[--xml-mode=wf|id|valid](#)

XMLSchema options

[--xsd-version=1.0|1.1|detect](#)

Help and version options

[--h, --help](#)
[--version](#)

valxquery

The **valxquery** command takes an XQuery file as its single argument and validates it.

```
valxquery [options] XQuery-File
```

The **xquery-File** arguments is the path and name of the XQuery file to be validated.

Examples

- `valxquery c:\Test.xquery`
- `valxquery --xquery-version=1 c:\Test.xquery`

Options

To go to an option's description, click the option or its group header.

- Boolean option values are set to `true` if the option is specified without a value.
- The values of all options can be specified without quotes except in two cases: (i) when the value string contains spaces, or (ii) when explicitly stated in the description of the option that quotes are required.

Options common to the 'xquery' and 'valxquery' commands

[--omit-xml-declaration=true|false](#)
[--xquery-version=1|3](#)

Catalog options

[--catalog=FILE](#)
[--user-catalog=FILE](#)

Global resource options

[--enable-globalresources=true|false](#)
[--gr, --globalresourcefile=FILE](#)
[--gc, --globalresourceconfig=VALUE](#)

Error options

[--error-limit=N|unlimited](#)

Message options

[--verbose=true|false](#)

XML instance options

[--xinclude=true|false](#)
[--xml-mode=wf|id|valid](#)

XMLSchema options

[--xsd-version=1.0|1.1|detect](#)

Help and version options

[--h, --help](#)
[--version](#)

Help and License Commands

This section describes two important features of RaptorXML:

- [Help Command](#): Describes how to display information about available commands, or about a command's arguments and options
- [License Commands](#): Describes how to license RaptorXML

Help Command

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

`help Command`

Note: When no argument is submitted, running the `help` command causes all available commands to be displayed, each with a short description of what it does.

Example

Example of the `help` command:

```
RaptorXML help valany
```

The command above contains one argument: the command `valany`, for which help is required. When this command is executed, it will display help information about the `valany` command.

The `--help` option

Help information about a command is also available by using the `--help` option with that command. For example, using the `--help` option with the `valany` command, as follows:

```
RaptorXML valany --help
```

achieves the same result as does using the `help` command with an argument of `valany`:

```
RaptorXML help valany
```

In both cases, help information about the `valany` command is displayed.

License Commands

Localization Commands

You can create a localized version of the RaptorXML application for any language of your choice. Four localized versions (English, German, Spanish, and Japanese) are already available in the `<ProgramFilesFolder>\Altova\RaptorXMLServer2013\bin\` folder. These four language versions therefore do not need to be created.

Create a localized version in another language as follows:

1. Generate an XML file containing the resource strings. Do this with the [exportresourcestrings](#) command. The resource strings in the generated XML file will be one of the four supported languages: English (`en`), German (`de`), Spanish (`es`), or Japanese (`ja`), according to the argument used with the command.
2. Translate the resource strings from the language of the generated XML file 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 RaptorXML DLL file from your translated XML file.
4. After you receive your localized DLL file from [Altova Support](#), save the DLL in the `<ProgramFilesFolder>\Altova\RaptorXMLServer2013\bin\` folder. Your DLL file will have a name of the form `RaptorXMLServer__lc.dll`. The `_lc` part of the name contains the language code. For example, in `RaptorXMLServer_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 RaptorXML application to use. For the argument of the [setdeflang](#) command, use the language code that is part of the DLL name.

Note: Altova is delivered with support for four languages: English, German, Spanish, and Japanese. So you do not need to create a localized version of these languages. To set

any of these four languages as the default language, use the CLI's [setdeflang](#) command.

exportresourcestrings

The **exportresourcestrings** command outputs an XML file containing the RaptorXML resource strings. The command takes two arguments: (i) the language of the resource strings in the output XML file, and (ii) the path and name of the output XML file. Allowed export languages (with their language codes in parentheses) are: English (*en*), German, (*de*), Spanish (*es*), and Japanese (*ja*).

exportresourcestrings *LanguageCode XMLOutputFile*

Note: On Linux systems, use an all-lowercase to call the executable.

Arguments

The **exportresourcestrings** command takes the following arguments:

<i>LanguageCode</i>	Specifies the target language of the export, that is, the language of resource strings in the exported XML file. Supported languages are: <i>en</i> , <i>de</i> , <i>es</i> , <i>ja</i>
<i>XMLOutputFile</i>	Specifies the location and name of the exported XML file..

Example

This command creates a file called `Strings.xml` at `c:\` that contains all the resource strings of the RaptorXML application translated into German.

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

setdeflang

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

setdeflang | *sdl* *LanguageCode*

Note: On Linux systems, use an all-lowercase to call the executable.

Example

This command sets the default language of the application's messages to German.

```
setdeflang de
```

Supported languages

The table below lists the languages currently supported together with their language codes.

en	English
de	German
es	Spanish
ja	Japanese

Options

This section contains a description of all CLI options, organized by functionality. To find out which options may be used with each command, see the description of the respective commands.

- [Catalogs](#)
- [Errors](#)
- [Global Resources](#)
- [Help and Version](#)
- [Messages](#)
- [Processing](#)
- [XBRL Evaluation](#)
- [XBRL Schemas](#)
- [XML Document](#)
- [XML Validation](#)
- [XML Schema Document \(XSD\)](#)
- [XQuery](#)
- [XSLT](#)
- [ZIP Files](#)

Catalogs

[\[--catalog, --user-catalog\]](#)

--catalog=FILE

Specifies the absolute path to a root catalog file that is not the installed root catalog file. The default value is the absolute path to the installed root catalog file.

--user-catalog=FILE

Specifies the absolute path to an XML catalog to be used in addition to the root catalog.

Note: Boolean option values are set to `true` if the option is specified without a value.

Errors

[\[--error-limit, --error-format\]](#)

--error-limit=N|unlimited

Specifies the error limit. Default value is `100`. Useful for limiting processor use during validation. When the error limit is reached, validation stops.

--error-format=`text|shortxml|longxml`

Specifies the format of the error output. Default value is `text`. The other options generate XML formats, with `longxml` generating more detail.

Note: Boolean option values are set to `true` if the option is specified without a value.

Global Resources

[**--enable-global-resources**](#), [**--globalresourcefile**](#), [**--globalresourceconfig**](#)

--enable-globalresources=`true|false`

Enables global resources. Default value is `false`.

--gr, **--globalresourcefile=***FILE*

Specifies the global resource file (and enables global resources).

--gc, **--globalresourceconfig=***VALUE*

Specifies the active configuration of the global resource (and enables global resources).

Note: Boolean option values are set to `true` if the option is specified without a value.

Help and Version

[**--help**](#), [**--version**](#)

--h, **--help**

Displays help text for the command. For example, `valany --h`. (Alternatively the `help` command can be used with an argument. For example: `help valany`.)

--version

Displays the version of RaptorXML. If used with a command, place **--version** before the command.

Note: Boolean option values are set to `true` if the option is specified without a value.

Messages

[**--log-output**](#), [**--verbose**](#)

--log-output=*FILE*

Writes the message output to the specified file URL instead of to the console. Ensure that the CLI has write permission to the output location.

--verbose=`true|false`

A value of `true` enables output of additional information during validation. Default value is `false`.

Note: Boolean option values are set to `true` if the option is specified without a value.

Processing

[**--listfile**](#), [**--script**](#), [**--streaming-serialization-enabled**](#), [**--streaming**](#)

--listfile=`true|false`

If `true`, treats the command's *InputFile* argument as a text file containing one filename per

line. Default value is `false`. (An alternative is to list the files on the CLI with a space as separator. Note, however, that CLIs have a maximum-character limitation.) Note that the `--listfile` option applies only to arguments, and not to options.

--script=File

Executes the Python script in the submitted file after validation has been completed.

--streaming=true|false

Enables streaming validation. Default is `true`. In streaming mode, data stored in memory is minimized and processing is faster. The downside is that information that might be required subsequently—for example, a data model of the XML instance document—will not be available. In situations where this is significant, streaming mode will need to be turned off (by giving `--streaming` a value of `false`). When using the `--script` option with the `valxml-withxsd` command, disable streaming.

Note: Boolean option values are set to `true` if the option is specified without a value.

XBRL Evaluation

[`--assertions-output-format`](#), [`--assertions-output-format`](#), [`--formula-execution`](#), [`--formula-output`](#), [`--formula-parameters-file`](#), [`--formula-parameters`](#)

--assertions-output-format=json|xml

Specifies the output format of the assertion evaluation. Default is `json`.

--assertions-output=FILE

Writes the output of the assertion evaluation to the specified *FILE*.

--formula-execution=true|false

Enables evaluation of XBRL formulas. Default is `false`.

--formula-output=FILE

Writes the output of formula evaluation to the specified *FILE*.

--formula-parameters-file=FILE

Specifies a *FILE* containing the parameters for XBRL formula evaluation.

--formula-parameters=JSON-ARRAY

Specifies the parameters for XBRL formula evaluation as an array of JSON maps.

--validate-dts-only=true|false

The DTS is discovered by starting from the XBRL instance document. All referenced taxonomy schemas and linkbases are discovered and validated. The rest of the XBRL instance document is ignored. Default value is `false`.

Note: Boolean option values are set to `true` if the option is specified without a value.

XBRL Schemas

[`--dimensions`](#), [`--formula`](#), [`--preload-formula-schemas`](#), [`--preload-xbrl-schemas`](#)

--dimensions=true|false

Enables XBRL Dimension 1.0 extensions. Default is `true`.

--formula=true|false

Enables XBRL Formula 1.0 extensions. Default is `true`.

--preload-formula-schemas=true|false

Preloads schemas of the XBRL Formula 1.0 specification. Default is `false`.

--preload-xbrl-schemas=true|false

Preloads schemas of the XBRL 2.1 specification. Default is `true`.

Note: Boolean option values are set to `true` if the option is specified without a value.

XML Instance

[***--xinclude, --xml-mode***](#)

--xinclude=true|false

Enables XML Inclusions (XInclude) support. Default value is `false`. When `false`, XInclude's `include` elements are ignored.

--xml-mode=wf|id|valid

Specifies the XML processing mode to use: `wf`=wellformed check; `id`=wellformed with ID/IDREF checks; `valid`=validation. Default value is `wf`.

Note: Boolean option values are set to `true` if the option is specified without a value.

XML Instance Validation

[***--dtd, --xsd, --namespaces, --assessment-mode***](#)

--dtd=FILE

Specifies the external DTD document to use for validation. If a reference to an external DTD is present in the XML document, then the CLI option overrides the external reference.

--xsd=FILE

Specifies one or more XML Schema documents to use for the validation of XML instance documents. Add the option multiple times to specify multiple schema documents.

--namespaces=true|false

Enables namespace-aware processing. This is useful for checking the XML instance for errors due to incorrect namespaces. Default value is `false`.

--assessment-mode=skip|lax|strict

Specifies the schema-validity assessment mode as defined in the XSD specifications. Default value is `strict`. The XML instance document will be validated according to the mode specified with this option.

Note: Boolean option values are set to `true` if the option is specified without a value.

XML Schema Document (XSD)

[***--schemalocation-hints, --schema-imports, --schema-mapping, --xsd-version***](#)

[***\[Note about schema location hints\]***](#)

--schemalocation-hints=load-by-schemalocation|

load-by-namespace|

load-combining-both|

ignore

- The `load-by-schemalocation` value uses the [URL of the schema location](#) in the `xsi:schemaLocation` and `xsi:noNamespaceSchemaLocation` attributes in XML instance documents. This is the **default value**.
- The `load-by-namespace` value takes the [namespace part](#) of `xsi:schemaLocation` and an empty string in the case of `xsi:noNamespaceSchemaLocation` and locates the schema via a catalog mapping.
- If `load-combining-both` is used and if either the namespace part or the URL part has a catalog mapping, then the catalog mapping is used. If both have catalog mappings, then the value of the [--schema-mapping](#) option decides which mapping is used. If neither the namespace nor URL has a catalog mapping, the URL is used.
- If the option's value is `ignore`, then the `xsi:schemaLocation` and `xsi:noNamespaceSchemaLocation` attributes are both ignored.

```
--schema-imports=load-by-schemalocation|
    load-preferring-schemalocation|
    load-by-namespace|
    load-combining-both|
    license-namespace-only
```

Specifies the behaviour of `xs:import` elements, each of which has an optional `namespace` attribute and an optional `schemaLocation` attribute: `<import namespace="someNS" schemaLocation="someURL">`. The behavior is as follows:

- `load-by-schemalocation`: The value of the `schemaLocation` attribute is used to locate the schema, taking account of catalog mappings. If the namespace attribute is present, the namespace is imported (licensed).
- `load-preferring-schemalocation`: If the `schemaLocation` attribute is present, it is used, taking account of catalog mappings. If no `schemaLocation` attribute is present, then the value of the namespace attribute is used via a catalog mapping. This is the **default value**.
- `load-by-namespace`: The value of the namespace attribute is used to locate the schema via a catalog mapping.
- `load-combining-both`: If either the namespace or `schemaLocation` attribute has a catalog mapping, then the mapping is used. If both have catalog mappings, then the value of the [--schema-mapping](#) option decides which mapping is used. If no catalog mapping is present, the `schemaLocation` attribute is used.
- `license-namespace-only`: The namespace is imported. No schema document is imported.

```
--schema-mapping=prefer-schemalocation|prefer-namespace
```

If either the `--schemalocation-hints` or the `--schema-imports` option has a value of `load-combining-both`, and if the namespace and URL parts involved both have catalog mappings, then the value of this option specifies which of the two mappings to use (namespace mapping or URL mapping; the `prefer-schemalocation` value refers to the URL mapping). Default is `prefer-schemalocation`.

--xsd-version=1.0|1.1|detect

Specifies the W3C Schema Definition Language (XSD) version to use. Default is 1.0. This option can also be useful to find out in what ways a schema which is 1.0-compatible is not 1.1-compatible. The `detect` option is an Altova-specific feature. It enables the version of the XML Schema document (1.0 or 1.1) to be detected by reading the value of the `version` attribute of the document's `<xs:schema>` element. If the value of the `@version` attribute is 1.1, the schema is detected as being version 1.1. For any other value, the schema is detected as being version 1.0. Note that this latter mechanism is not defined in the XML Schema specification; it is intended to conform with the schema-detection mechanism used in other Altova applications.

Note about schema location hints

Instance documents can use hints to indicate the schema location. Two attributes are used for hints:

- `xsi:schemaLocation` for schema documents with target namespaces. The attribute's value is a pair of items, the first of which is a namespace, the second is a URL that locates a schema document. The namespace name must match the target namespace of the schema document.

```
<document xmlns="http://www.altova.com/schemas/test03"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://www.altova.com/schemas/test03
```

```
Test.xsd">
```

- `xsi:noNamespaceSchemaLocation` for schema documents without target namespaces. The attribute's value is the schema document's URL. The referenced schema document must have no target namespace.

```
<document xmlns="http://www.altova.com/schemas/test03"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:noNamespaceSchemaLocation="Test.xsd">
```

The `--schemalocation-hints` option specifies how these two attributes are to be used as hints, especially how the `schemaLocation` attribute information is to be handled (*see the option's description above*). Note that `noNamespaceSchemaLocation` considers the namespace part of the `xsi:noNamespaceSchemaLocation` value to be the empty string.

Schema location hints can also be given in an `import` statement of an XML Schema document.

```
<import namespace="someNS" schemaLocation="someURL">
```

In the `import` statement, too, hints can be given via a namespace that can be mapped to a schema in a catalog file, or directly as a URL in the `schemaLocation` attribute. The [--schema-imports](#) option specifies how the schema location is to be selected.

XQuery**Options specific to the `xquery` command**

[\[--indent-characters, --input, --output, --output-encoding, --output-indent, --output-method, --param\]](#)

- indent-characters=VALUE**
Specifies the character string to be used as indentation.
- input=FILE**
The URL of the XML file to be transformed.
- output=FILE**
The URL of the primary-output file. For example, in the case of multiple-file HTML output, the primary-output file will be the location of the entry point HTML file. If no `--output` option is specified, output is written to standard output.
- output-encoding=VALUE**
The value of the encoding attribute in the output document. Valid values are names in the IANA character set registry. Default value is `UTF-8`.
- output-indent=true|false**
If `true`, the output will be indented according to its hierarchic structure. If `false`, there will be no hierarchical indentation. Default is `false`.
- output-method=xml|html|xhtml|text**
Specifies the output format. Default value is `xml`.
- p, --param=KEY:VALUE**
Specifies the value of an external parameter. An external parameter is declared in the XQuery document with the `declare variable` declaration followed by a variable name and then the `external` keyword followed by the trailing semi-colon. For example:
`declare variable $foo as xs:string external;`
Because of the `external` keyword `$foo` becomes an external parameter, the value of which is passed at runtime from an external source. The external parameter is given a value with the CLI command. For example:
`--param=foo:'MyName'`
In the description statement above, *KEY* is the external parameter name, *VALUE* is the value of the external parameter, given as an XPath expression. Parameter names used on the CLI must be declared in the XQuery document. If multiple external parameters are passed values on the CLI, each must be given a separate `--param` option. Double quotes must be used if the XPath expression contains spaces.

Note: Boolean option values are set to `true` if the option is specified without a value.

Options common to the `xquery` and `valxquery` commands

[`--omit-xml-declaration`](#), [`--xquery-version`](#)

- omit-xml-declaration=true|false**
Serialization option to specify whether the XML declaration should be omitted from the output or not. If `true`, there will be no XML declaration in the output document. If `false`, an XML declaration will be included. Default value is `false`.
- xquery-version=1|3**
Specifies whether the XQuery processor should use XQuery 1.0 or XQuery 3.0. Default value is 3.

Note: Boolean option values are set to `true` if the option is specified without a value.

XSLT

Options specific to the `xslt` command

[`--indent-characters`](#), [`--input`](#), [`--output`](#), [`--param`](#), [`--streaming-serialization-enabled`](#)

`--indent-characters=VALUE`

Specifies the character string to be used as indentation.

`--input=FILE`

The URL of the XML file to be transformed.

`--output=FILE`

The URL of the primary-output file. For example, in the case of multiple-file HTML output, the primary-output file will be the location of the entry point HTML file. If no `--output` option is specified, output is written to standard output.

`--p, --param=KEY:VALUE`

Specifies a global stylesheet parameter. *KEY* is the parameter name, *VALUE* is an XPath expression that provides the parameter value. Parameter names used on the CLI must be declared in the stylesheet. If multiple parameters are used, the `--param` switch must be used before each parameter. Double quotes must be used around the XPath expression if it contains a space—whether the space is in the XPath expression itself or in a string literal in the expression. *For example:*

```
xslt --input=c:\Test.xml --output=c:\Output.xml --param date=//node/@att1
--p=title:'stringwithoutspace' --param=title:"'string with spaces'"
--p=amount:456 c:\Test.xslt
```

`--streaming-serialization-enabled=true|false`

Enables streaming serialization. Default value is `true`.

Note: Boolean option values are set to `true` if the option is specified without a value.

Options common to the `xslt` and `valxslt` commands

[`--chartext-disable`](#), [`--dotnetext-disable`](#), [`--javaext-barcode-location`](#), [`--javaext-disable`](#), [`--template-entry-point`](#), [`--template-mode`](#), [`--xslt-version`](#)

`--chartext-disable=true|false`

Disables chart extensions. Default value is `false`. *Not available in Development Edition.*

`--dotnetext-disable=true|false`

Disables .NET extensions. Default value is `false`.

`--javaext-barcode-location=FILE`

Specifies the location of the barcode extension file.

`--javaext-disable=true|false`

Disables Java extensions. Default value is `false`.

`--template-entry-point=VALUE`

Gives the name of a named template in the XSLT stylesheet that is the entry point of the transformation.

--template-mode=VALUE

Specifies the template mode to use for the transformation.

--xslt-version=1|2|3

Specifies whether the XSLT processor should use XSLT 1.0, XSLT 2.0, or XSLT 3.0. Default value is 3.

Note: Boolean option values are set to `true` if the option is specified without a value.

ZIP Files

[--recurse](#)

--recurse=true|false

Used to select files within a ZIP archive. If `true`, the command's *InputFile* argument will select the specified file also in subdirectories. For example: `test.zip|zip\test.xml` will select files named `test.xml` at all folder levels of the zip folder. The wildcard characters `*` and `?` may be used. So, `*.xml` will select all `.xml` files in the zip folder. The parameter's default value is `false`. It is not available in the Development Edition.

Note: Boolean option values are set to `true` if the option is specified without a value.

Chapter 3

FlowForce Tutorial

3 FlowForce Tutorial

The aim of the tutorial is to:

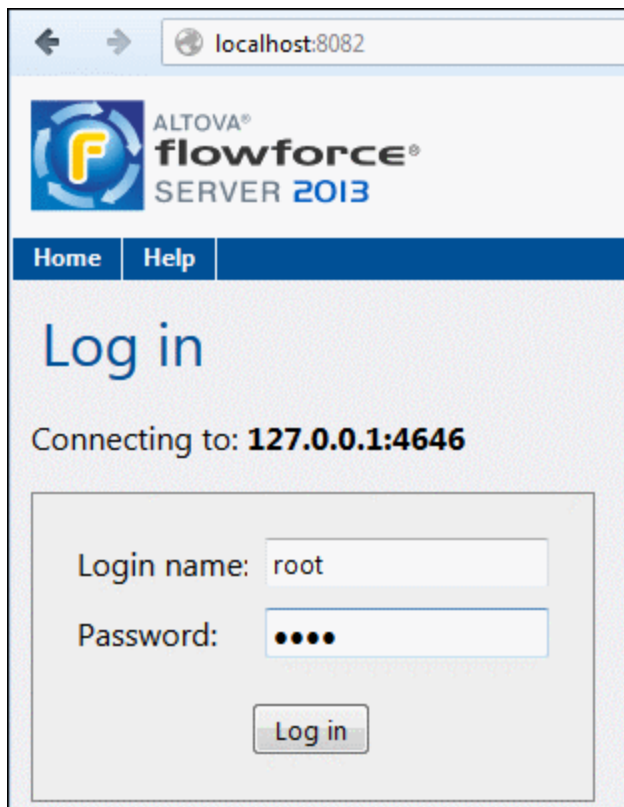
- [Deploy a MapForce](#) mapping to FlowForce Server, and create and execute a job at a specific time.
- [Create a subjob](#) that copies the previously generated output files into an archive directory.
- [Check a directory](#) for new files to be passed on to a job, that uses the new file as an input file.
- Supply [job parameters](#) at runtime, to a deployed mapping, that queries a database.
- Use a deployed mapping as a [web service](#), and view the mapping results in a browser.
- Deploy a [StyleVision transformation](#) file to FlowForce Server and have it run at a specific time each day.
- Using RaptorXML Server to [validate a document](#).

Both FlowForce Server and FlowForce Administration Interface need to be started to deploy mappings or to manage the server.

This tutorial assumes that FlowForce Server and MapForceServer have been [installed](#) and registered with LicenseServer, and have also been assigned the correct licenses by your Administrator. The browser interface is identical from this point on for Windows and Linux.

To log in to the FlowForce Web Administration Interface:

1. Start your browser and enter <http://localhost:8082>. If you changed the port on the FlowForce Server Configuration page, use the one you entered there.
You are now connected to FlowForce Web Server and the Login page for FlowForce Server is opened.



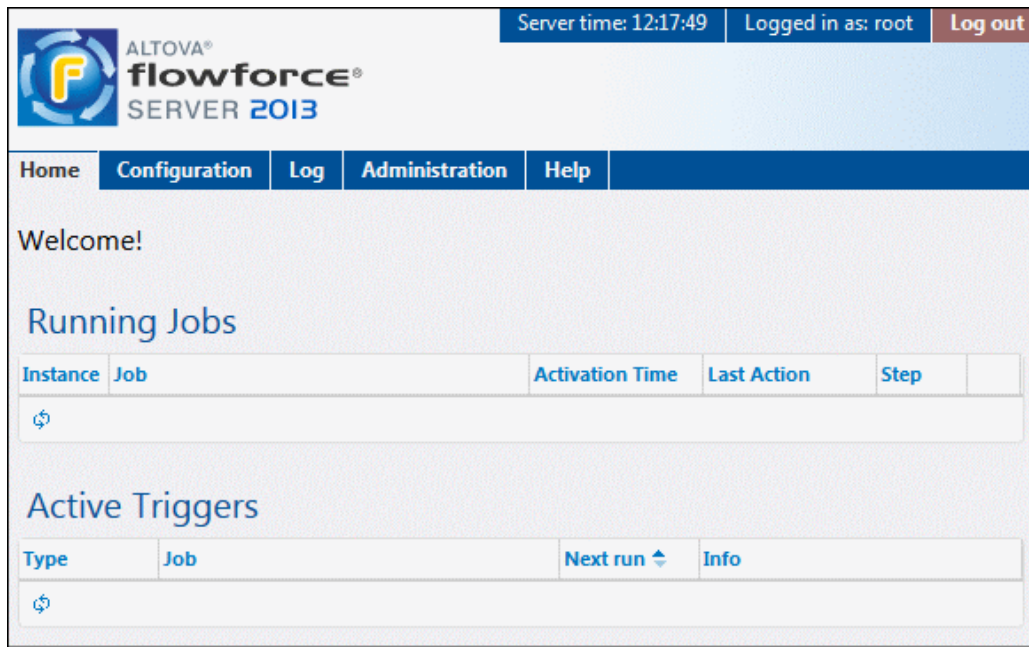
A screenshot of the FlowForce Server 2013 login interface. The browser address bar shows 'localhost:8082'. The page header includes the 'ALTOVA® flowforce® SERVER 2013' logo and navigation links for 'Home' and 'Help'. The main heading is 'Log in'. Below it, it says 'Connecting to: 127.0.0.1:4646'. There is a login form with two fields: 'Login name:' with the value 'root' and 'Password:' with four dots. A 'Log in' button is at the bottom of the form.

Enter login name "**root**", as well as the password "**root**" if this is the first time that you have started FlowForce Server.

2. Click the "Log in" button to log in.

You have now logged onto FlowForce Server.

Connection information, as well as any running jobs and active triggers are visible on the Home screen.



A screenshot of the FlowForce Server 2013 Home screen after login. The top status bar shows 'Server time: 12:17:49', 'Logged in as: root', and a 'Log out' button. The navigation bar includes 'Home', 'Configuration', 'Log', 'Administration', and 'Help'. The main content area starts with a 'Welcome!' message. Below it is a section titled 'Running Jobs' with a table that has columns: 'Instance', 'Job', 'Activation Time', 'Last Action', and 'Step'. The table is currently empty. Below this is a section titled 'Active Triggers' with a table that has columns: 'Type', 'Job', 'Next run', and 'Info'. This table is also empty.

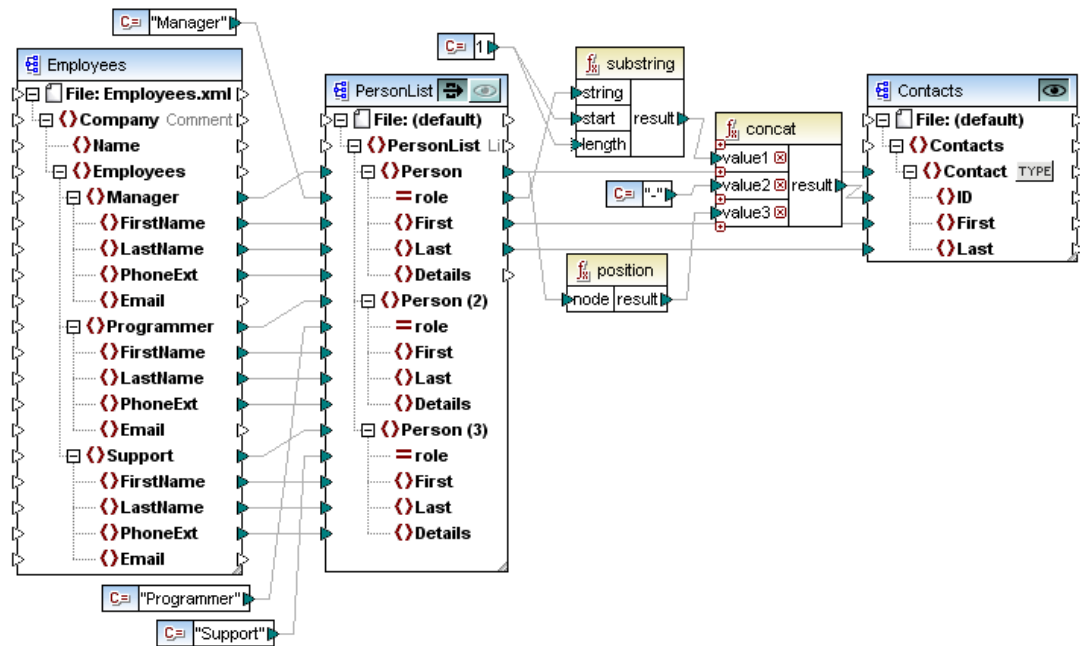
3.1 Deploying a MapForce mapping

Aim: to deploy a MapForce mapping.

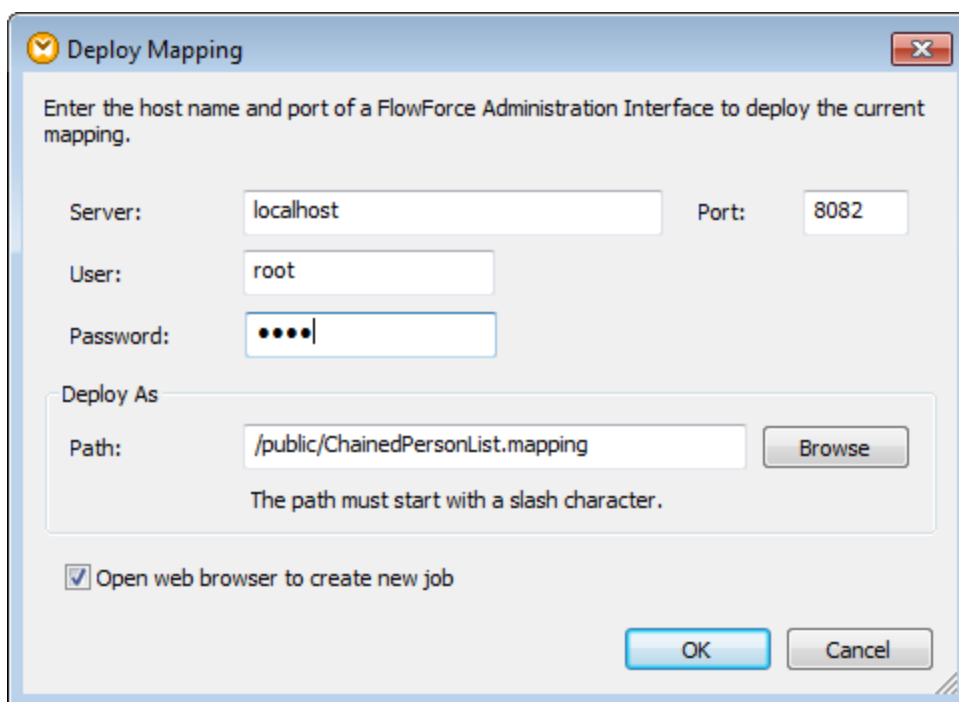
Deploying a mapping means that MapForce organizes all the mapping resources, used by the specific mapping, into an object and passes it on to the server/machine running FlowForce.

To deploy a mapping in MapForce

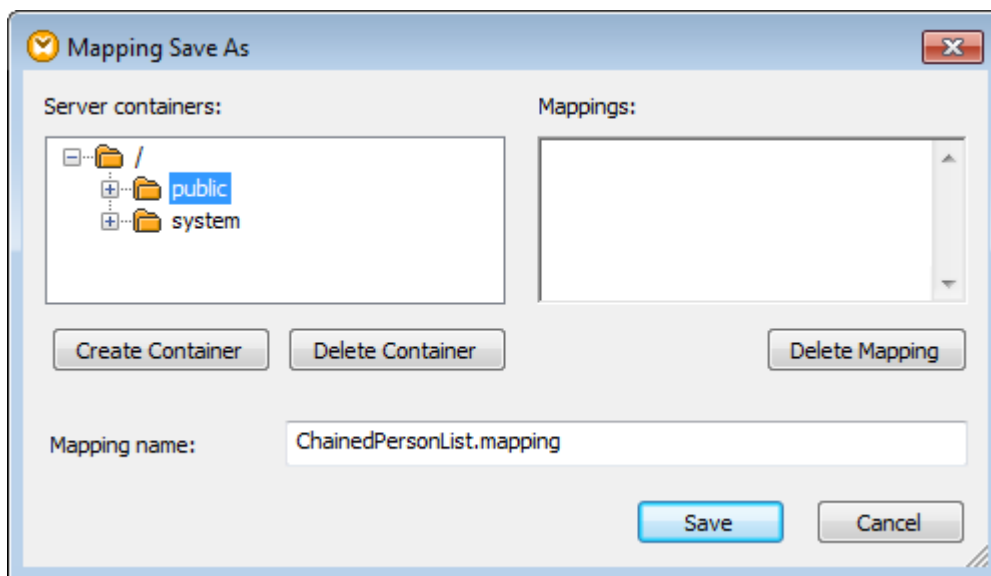
1. Open a mapping in MapForce e.g. **ChainedPersonList.mfd**.



2. Select the menu option **File | Deploy to FlowForce Server**.
3. Enter the Server name and Port of the web administration interface in the respective fields, e.g. localhost and 8082 if FlowForce Server is running on the same machine and the default port is used.
4. Enter the User Name and Password needed to access the server, e.g. "root" and "root".



5. Optionally, click on the "Browse" button to define where the mappings are going to be placed inside the FlowForce Server's object system ("public" is selected by default), then click Save.
Make sure the "Open web browser to create new job" check box is active.



7. Click OK to deploy.

The messages window shows if the mapping deployed successfully. The FlowForce Server Administration Interface is automatically opened in your web browser, and a partially filled in job page is displayed.

Create job in [/ public /](#)

Job name:

Job description:

Job Input Parameters

[+](#)

Execution Steps

[+](#)

Execute function
/public/ChainedPersonList.mapping

Parameters:

Employees:	(input)	+
PersonList:	(in/out)	+
Contacts:	(output)	+
Working-directory:		+

= Assign this step's result to

[new Execution step](#) [new Choose step](#) [new For-each step](#) [new error/success handling step](#)

Triggers

[new Timer](#) [new Filesystem trigger](#) [new HTTP trigger](#)

The next thing to do is to [define the rest of the job](#), i.e. the Job Triggers, the Job Credentials, and the Execution Steps.

3.2 Defining a job - triggers & execution steps

Aim: To define a simple job that:

- Uses the deployed mapping function(s) from MapForce
- Uses a local, or predefined credential: see [Administration Guide](#)
- Triggers the job at a specific time

As this mapping was deployed from MapForce and the "Open web browser..." check box was activated, you do not have to navigate to the Jobs page, it is automatically created for you.

To define the job execution steps:

Some of the fields of the Execution steps group have been filled out automatically. (You would normally have to click the "+" button to add a new Execution step.)

The "Execute function" is: /public/ChainedPersonList.mapping

Execution Steps			
+ Execute function /public/ChainedPersonList.mapping			
Parameters:	Employees:	(input)	+
	PersonList:	(in/out)	+
	Contacts:	(output)	+
	Working-directory:	c:\temp	
= Assign this step's result to <input type="text" value="name"/>			

1. Click the "+" button of the Working-directory entry to enter a different directory, e.g. c:\temp. Note that this must be a path on the server machine (that runs FlowForce), not on your local machine.

As we do not want to **override** any of the parameter settings defined by the deployed mapping, we are not going to change any other Parameter settings.

To define the trigger:

1. Click the "new Timer" button, in the Triggers group.
2. Click into the (Start) date field and select the start date from the date picker.
3. Enter the time the job is to be triggered. Note that the time is entered in 24 hour format. For testing purposes, use a time close to your current time.

As soon as the trigger time is reached, the job is executed and the output files, generated by the mapping, appear in the c:\temp directory. The output files are PersonList.xml and Contacts.xml.

To define the credentials:

There are two ways that you can define job credentials:

- In the Credential group, click the "Select existing credential" combo box and select a previously defined credential e.g. "Cred_production".
- Manually enter your personal server credentials in the User name and Password fields of the local credential group.

Note:

Entering credentials manually, forces you to update the credentials of this job if your **server** login changes.

To save your job:

1. Click the Save button.
A save confirmation message appears at this point. If any mandatory fields were left out, you will be shown where this is the case.

Running the job:

As soon as the trigger time has been reached, (i.e. 11:32) the trigger will fire and the job will be executed. Two XML files are placed in the c:\temp folder: Contacts.xml and PersonList.xml.

Viewing the job log

- Click the [View log](#) button near the top of the left of the Job page, to open the Log View filtered for the current job.

The execution parameters and the execution status are displayed in the Log View table.

Log View

☒ Show last 7 days
☐ Show from 2013-02-01 to 2013-02-08

filter by job path: Minimum severity: Info [Show](#)

Page 1 of 2 25

Date	Severity	Module	User	Instan	Message
2013-02-08 15:27:00	INFO	flowforce	root	4	Finished job execution: /public/ChainedPersonList.job
2013-02-08 15:27:00	ERROR	flowforce	root	4	Step MapForce.Mapping completed with status: 1 more
2013-02-08 15:27:00	INFO	flowforce	root	4	Executing MapForce.Mapping with parameters: {"Worki "Employees": "altova://packagedfile/C:/ProgramData/A /MapForceExamples/Employees.xml", "PersonList": "Per

Viewing the defined job

1. Click the ChainedPersonListJob **link** in the Message column.
This opens the previously executed job definition page.

Notes:

Click the "Configuration" button to see the various containers. Click a folder e.g. public, to see its contents. To see the root container contents, click "/".

Home Configuration Log Administration Help

Container / public / [Search](#) ☒ Recursive

<input type="checkbox"/> Name	Type	Next run	
<input type="checkbox"/> ChainedPersonList.mapping	function		
<input type="checkbox"/> ShortApplicationInfo.mapping	function		
<input type="checkbox"/> ChainedPersonList.job	job		View log
<input type="checkbox"/> Test1	job		View log

[Create](#) [Delete Selected Objects](#) [Permissions](#)

Each container object, i.e. function, job, credential, etc., has a check box to the left of its name to select it. The topmost check box, to the left of the column header "Name", selects/deselects **all** objects in the list.

The "Create" button lets you create Containers, Jobs, and Credentials.

The "Delete Selected Objects" button becomes active when objects have been selected in the list; clicking it deletes the selected objects.

See: [Defining a subjob](#)

3.3 Defining a subjob

Aim: To create and integrate a separate job that:

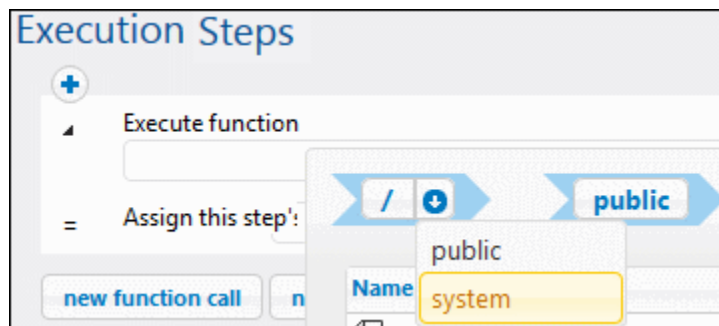
- Copies the mapping result files of the previous job into an archive directory.
This job will act as a subjob to the calling job "ChainedPersonListJob".

To create a new job:

1. Click the "Configuration" button, then click the container "public".
2. Click the "Create" button and select "Create Job" from the popup menu.
3. Enter the name of the job, e.g. "copy2archive".
There is no need to define a trigger for this job, as it will be called by another job.

Defining the subjob execution steps:

1. In the Execution Steps group click the New Execution step button, then click into the "Execute function" field.
A popup window opens allowing you to select the function (or job) from the public folder.



2. Click the down arrow of the left-hand tag and select "system".
Another popup window is opened.
3. Select **//filesystem/copy**.
4. Enter the name of the file that you want to copy in the Source field, e.g. c:\temp\Contacts.xml.
5. Enter the name of the destination directory, e.g. c:\archive. (Click the "+" button next to the Overwrite field, and activate the check box, if you want to be able to overwrite the same file at the destination.)
6. Enter the name of the working directory, e.g. c:\temp.
7. Click the "new Execution step" button under the previously defined Execution function to add a second copy function.
Source c:\temp\PersonList.xml, destination c:\archive, and working directory, c:\temp.
8. Click "Save" to save the job.

Execution Steps

+ **Execute function** /system/filesystem/copy

Parameters: Source: c:\temp\Contacts.xml
Target: c:\archive
Overwrite: ☒
Working directory: c:\temp

= Assign this step's result to name

+ **Execute function** /system/filesystem/copy

Parameters: Source: c:\temp\PersonList.xml
Target: c:\archive
Overwrite: ☒
Working directory: c:\temp

= Assign this step's result to name

new Execution step new Choose step new For-each step new error

9. Select an existing credential or enter your local credentials.

Calling a job from another job:

1. Click the "Configuration" button, then the public container, and select ChainedPersonList.job.
2. Scroll down to the Execution Steps group and click the "New Execution step" button to add a new execution step.
3. Click the Execute function combo box and select **copy2archive** from the popup.

Execution Steps


+


Execute function


/public/ChainedPersonList.mapping


▼

Parameters:

Employees: (input)  +

PersonList: (in/out)  +

Contacts: (output)  +

Working-directory:  +

= Assign this step's result to

+

Execute function

/public/copy2archive

▼

= Assign this step's result to

4. Update the timer trigger and click the "Save" button.
5. After the job has been triggered, click the "View Log" button at the top of the job page.

Log View

☐ Show last 7 days
 ☐ Show from to

filter by: Job Path

Page 1 of 5

25

Date	Severity	Module	User	InstanceID	Message
2012-08-08 17:21:01	INFO	flowforce	root	1455	Finished job execution: /public/ChainedPersonList/job
2012-08-08 17:21:00	INFO	flowforce	root	1455	Step FlowForce.move completed with status: 0 more
2012-08-08 17:21:00	INFO	flowforce	root	1455	Executing FlowForce.move with parameters: {"Source": "c:\\temp\\PersonList.xml", "Working-directory": "c:\\temp", "Destination": "c:\\archive\\", "Overwrite": "true"}
2012-08-08 17:21:00	INFO	flowforce	root	1455	Step FlowForce.move completed with status: 0 more
2012-08-08 17:21:00	INFO	flowforce	root	1455	Executing FlowForce.move with parameters: {"Source": "c:\\temp\\contacts.xml", "Working-directory": "c:\\temp", "Destination": "c:\\archive\\", "Overwrite": "true"}
2012-08-08 17:21:00	INFO	flowforce	root	1455	Step MapForce.Mapping completed with status: 0 more
2012-08-08 17:21:00	INFO	flowforce	root	1455	Executing MapForce.Mapping with parameters: {"PersonList": "PersonList.xml", "Working-directory": "c:\\temp", "Employees": "altova://packagedfile/C:/Users/alp/Documents/Altova/MapForce2013/MapForceExamples/Employees.xml", "Contacts": "Contacts.xml"}
2012-08-08 17:21:00	INFO	flowforce	root	1455	Starting job execution: /public/ChainedPersonList/job

You can now see the status of the job and its subjob.

The two XML files generated by the first job have been copied to the archive directory.

Note:

If you want to rename a file when it is copied, enter the new file name in the "Destination" field.

See: [Directory polling - acting on a trigger file](#)

Altova FlowForce

© 2013 Altova GmbH

3.4 Directory polling - acting on a trigger file

Aim: to check a directory for new XML files, execute the deployed mapping with those files, and copy the result files into an archive directory.

Deploying the mapping and creating the job:

1. Open the **ShortApplicationInfo.mfd** mapping in MapForce.
2. Select the menu option **File | Deploy to FlowForce Server**.
3. Enter the password in the Password field; make sure that the "Open web browser..." check box is active, then click OK.
This generates a new job in the public directory of FlowForce.
4. A default job name is automatically entered, i.e. ShortApplicationInfo.job.

5. Select **"/public/ShortApplicationInfo.mapping"** from the Function combo box to use the previously defined mapping (if not already selected).
6. Click the "+" icon to the right of the **SectionedPage** label.

This creates an edit field.

6. Click the **Set to** button to the right of the expanded SectionedPage field, and select "triggerfile".
7. This causes the input field contents to change to {triggerfile}.
8. Click the Working directory "+" button and enter the working directory e.g. c:\temp.

The file in the directory being polled, will now be used as the input file in Parameters group of the Execution step.

Note that the name of the (output) file is also shown as a parameter in this group, i.e. "ShortInfo.xml".

Creating the polling trigger:

1. In the Triggers section, click the "new File System" trigger button.
2. Select Modified Date in the Check field.
3. Enter the directory name and file types that you want to check for, e.g. c:\temp*.xml.
4. Enter the polling interval, e.g. 60 seconds.

5. Select the credential you want to use for this job, or enter your local credentials.

Note:

When you select "File system trigger", the "triggerfile" entry is automatically added to the "Job input parameters" group.

Adding a second job execution step:

1. Click the "new Execution step" button below the execution step that was just created.
2. Use the Execute function combo box to select "/system/filesystem/move".
3. In the Source field, enter the path/file name of file that you want to move, e.g. c:\temp\ShortInfo.xml.
4. Enter the location of the Destination directory, e.g. c:\archive.
5. Enter the working directory c:\temp.

6. Click the "Save" button to save the job.

As soon as the trigger start time has been reached, the trigger will be active and c:\temp folder will be polled every 60 seconds.

Note:

When entering file names in the Source and Destination fields be aware that the entries are **case sensitive!** (e.g. shortinfo.xml will not work as the name of the source file, as the file name generated by the mapping deployment is **ShortInfo.xml**).

To start the dirPolling job:

- Navigate to your ...\Altova\source folder and copy the file that starts the mapping process to the c:\temp folder (e.g. **ApplicationsPage.xml**)

Result:

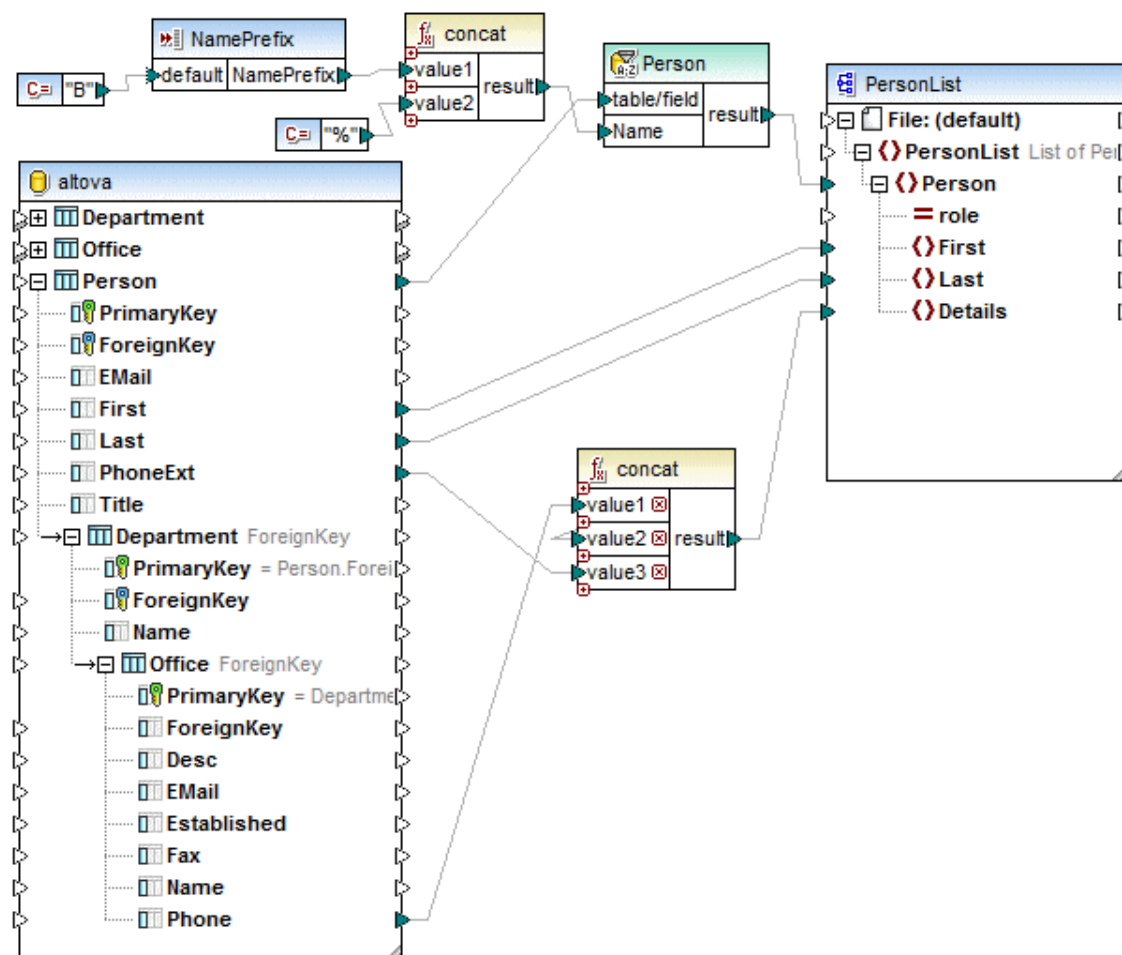
- As soon as a new XML file is found, that XML file becomes the input file for the deployed mapping.
- The job is started and the result of the processed input file (ShortInfo.xml) is moved to the archive directory.

See: [Using parameters to query a database](#)

3.5 Using parameters to query a database

Aim: to query a database using job input parameters, via a web browser.

- This example uses the DB_PhoneList.mfd mapping available in the ... \MapForceExamples folder.
- The NamePrefix input parameter of the mapping, will be used to supply the query data in the browser client.



Deploying the mapping and creating the job:

1. Open the DB_PhoneList.mfd mapping in MapForce.
2. Select the menu option **File | Deploy to FlowForce Server**.
3. Enter the password (root) in the Password field; make sure that the "Open web browser..." check box is active, then click OK.
This generates a new job in the public directory of FlowForce.
4. Click the "+" button of Job input parameters and enter NamePrefix.

Create job in / public /

Job name: DB_PhoneList.job

Job description:

Job input parameters

+
Name: NamePrefix Type: string Default: + Description:

+
Execution Steps

+
Execute function /public/DB_PhoneList.mapping

Parameters: NamePrefix: {NamePrefix}
PersonList: (output) +
Working-directory: +

5. Click the "+" button to the right of the NamePrefix entry in the Execution steps group.
6. Click the "Set to" button and select NamePrefix (the parameter name is automatically available).

Execution Steps

+
Execute function /public/DB_PhoneList.mapping

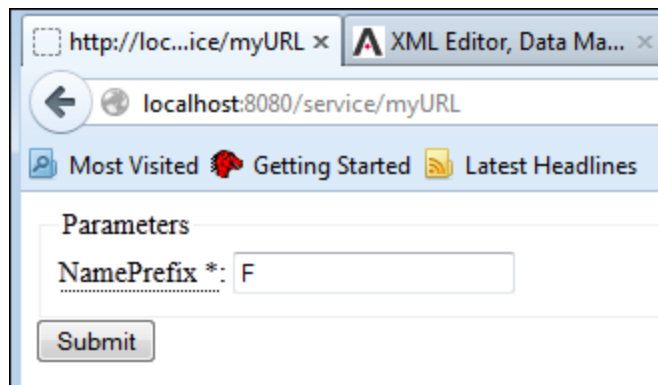
Parameters: NamePrefix: {NamePrefix} as xs:string (optional) Set to
PersonList: (output) +
Working-directory: +

7. Click the check box in the Service group and enter **myURL** in the text box.
8. Select the credential you want to use for this job, e.g. cred_production.
9. Click the "Save" button to save the job.

Note: No triggers were defined because the web browser is used to access the service.

Using the browser to run the job and query the database:

1. Open your browser and enter <http://localhost:4646/service/myURL> in the URL text box. If you changed the [port number](#) for FlowForce Server, please use that one.
2. Enter the letter of the last name of the person(s) you are looking for, e.g. "F".



http://loc...ice/myURL x XML Editor, Data Ma... x

localhost:8080/service/myURL

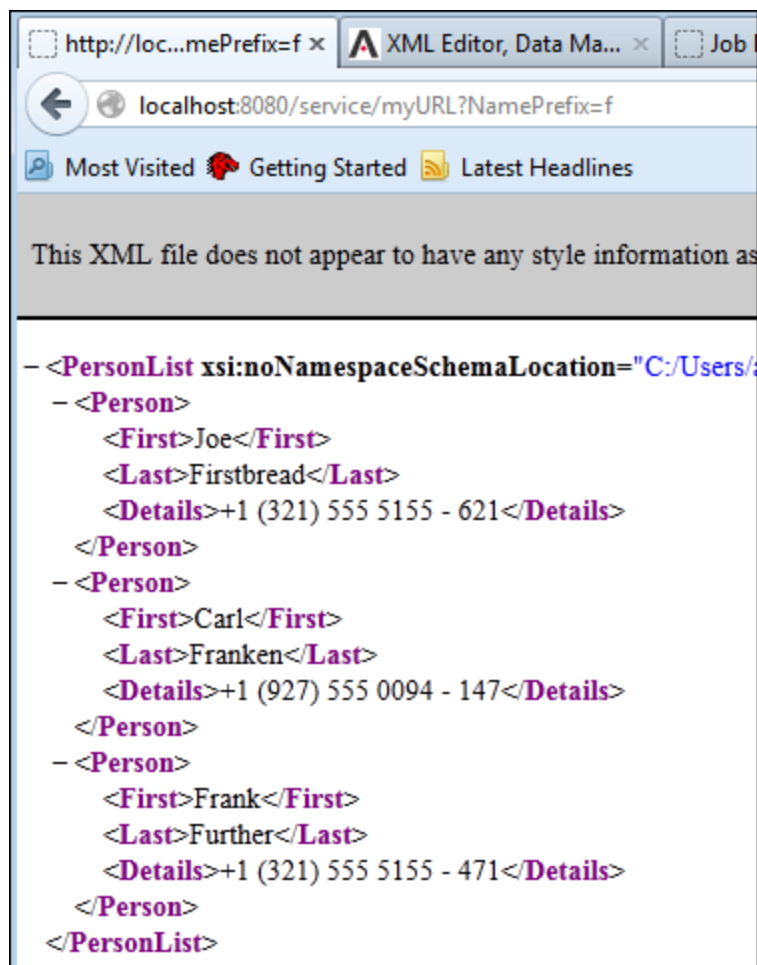
Most Visited Getting Started Latest Headlines

Parameters

NamePrefix *: F

Submit

3. Click the "Submit" button to send the query to the server.



http://loc...mePrefix=f x XML Editor, Data Ma... x Job I

localhost:8080/service/myURL?NamePrefix=f

Most Visited Getting Started Latest Headlines

This XML file does not appear to have any style information as

```
- <PersonList xsi:noNamespaceSchemaLocation="C:/Users/Job I/
- <Person>
  <First>Joe</First>
  <Last>Firstbread</Last>
  <Details>+1 (321) 555 5155 - 621</Details>
</Person>
- <Person>
  <First>Carl</First>
  <Last>Franken</Last>
  <Details>+1 (927) 555 0094 - 147</Details>
</Person>
- <Person>
  <First>Frank</First>
  <Last>Further</Last>
  <Details>+1 (321) 555 5155 - 471</Details>
</Person>
</PersonList>
```

The resulting XML file is now displayed in the browser.

Please note:

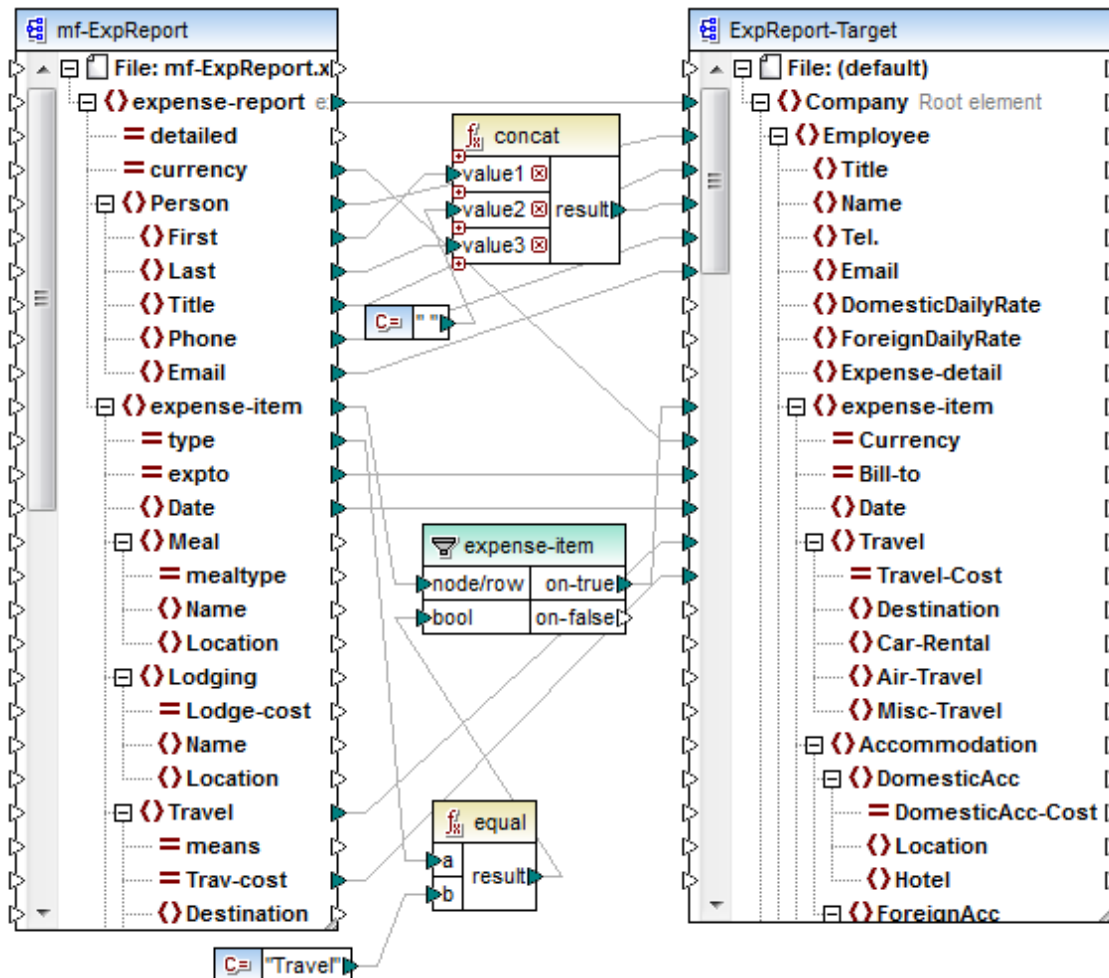
You can press the "Back" button and reenter a query for a different Last name.

See: [Using a deployed mapping as a web service](#)

3.6 Using a deployed mapping as a web service

Aim: to supply various XML files as source files, and view the mapping result via a browser client.

- This example uses the **Tut-ExpReport.mfd** file available in the ... \MapForceExamples\Tutorial folder.
- The XML source file, mf-ExpReport.xml, will be replaced at runtime in the browser client.



Deploying the mapping and creating the job:

1. Open the **Tut-ExpReport.mfd** mapping in MapForce.
2. Select the menu option **File | Deploy to FlowForce Server**.
3. Enter the password (root) in the Password field; make sure that the "Open web browser..." check box is active, then click OK.
This generates a new job in the /public directory of FlowForce.
4. Click the "+" button of the Job input parameters group and enter Expenses.

Job Input Parameters

+
 Name: Type: Description:

Execution Steps

+
 Execute function

Parameters:

mf-ExpReport:	(input)		+
ExpReport-Target:	(output)		+
Working-directory:			+

= Assign this step's result to

Triggers

Service

☒ Make this job available via HTTP at URL

- Click the "Type" combo box of Job input parameters and select **stream**. This defines the parameter to represent a file uploaded to the server with the HTTP POST request.
- Click the "+" button to the right of the mf-ExpReport entry in the Execution Steps group.
- Enter the expression **{as-file(Expenses)}**. Please see: [Streaming functions](#) for more information.

Execution

+
 Execute function

Parameters:

mf-ExpReport:	(input)		<input type="text" value="{as-file(Expenses)}"/>	as xs:string (optional)	<input type="button" value="Set to"/>
ExpReport-Target:	(output)		+		
Working-directory:			+		

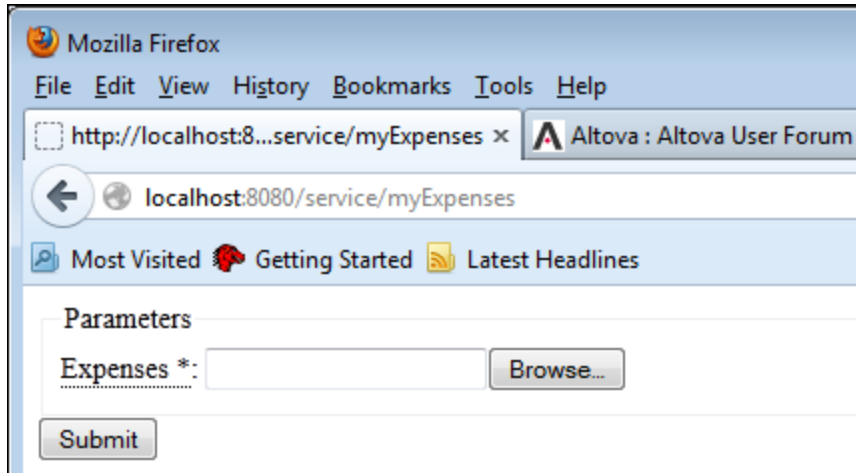
= store result in

- Click the check box in the Service group and enter **myExpenses** in the text box.
- Select the credential you want to use for this job.
- Click the "Save" button to save the job.

Note: No triggers were defined because the browser is used to access the service.

Using the browser to run the job and choose different XML input files:

1. Open your browser and enter <http://localhost:4646/service/myExpenses> in the URL text box.
2. Click the "Browse" button.



3. Select mf-ExpReport.xml in the ...\\MapForceExamples\\Tutorial folder from the dialog box, and click "Submit".

```
<Company xsi:schemaLocation="http://my-company.com/namespace C:/Users
- <Employee>
  <Title>Project Manager</Title>
  <Name>Fred Landis</Name>
  <Tel>123-456-78</Tel>
  <Email>f.landis@nanonull.com</Email>
  - <expense-item Currency="USD" Bill-to="Development">
    <Date>2003-01-02</Date>
    <Travel Travel-Cost="337.88"/>
  </expense-item>
  - <expense-item Currency="USD" Bill-to="Accounting">
    <Date>2003-07-07</Date>
    <Travel Travel-Cost="1014.22"/>
  </expense-item>
  - <expense-item Currency="USD" Bill-to="Marketing">
    <Date>2003-02-02</Date>
    <Travel Travel-Cost="2000"/>
  </expense-item>
</Employee>
</Company>
```

4. Click the browser "Back" button, then the myExpenses link, and click the "Browse" button.
5. Select mf-ExpReport2.xml from the dialog box, and click "Submit".

```
- <Company xsi:schemaLocation="http://my-company.com/namespace C:/Users  
- <Employee>  
  <Title>Manager</Title>  
  <Name>James Johnson</Name>  
  <Tel.>456-789-123</Tel.>  
  <Email>j.john@nanonull.com</Email>  
- <expense-item Currency="Euro" Bill-to="Sales">  
  <Date>2004-02-03</Date>  
  <Travel Travel-Cost="150.44"/>  
</expense-item>  
- <expense-item Currency="Euro" Bill-to="Operations">  
  <Date>2004-08-08</Date>  
  <Travel Travel-Cost="1020"/>  
</expense-item>  
- <expense-item Currency="Euro" Bill-to="Support">  
  <Date>2004-03-03</Date>  
  <Travel Travel-Cost="70"/>  
</expense-item>  
</Employee>  
</Company>
```

Two completely different Expense reports were processed by the deployed mapping and output to the browser.

Note:

The path of the XML output instance file is saved with the deployed mapping. There is therefore no need to supply the working directory.

3.7 Deploying a StyleVision transformation

The **File | Deploy to FlowForce** command of Altova StyleVision enables you to deploy a `.transformation` file from StyleVision to your Altova FlowForce Server. The `.transformation` file contains all the files and information required to carry out transformations as designed in the SPS (stylesheet) you created with Altova StyleVision.

After the `.transformation` file has been deployed to the FlowForce Server, you can create jobs in Altova FlowForce Server that use the `.transformation` file to generate transformations according to triggers specified in the job definition.

In this section, we describe how to deploy a StyleVision transformation to FlowForce Server and important points related to deployment. You can try out the deployment process described below by using the `AutoCalc.sps` file in the `StyleVisionExamples` folder as your starting point. The file `AutoCalc.sps` can also be opened from the `Basics` folder of StyleVision's `Examples` project.

A `.transformation` file is generated from a Portable XML Format (PXF) file. So, the **File | Deploy to FlowForce** command of StyleVision can be used when a PXF file is active. (If an SPS file is active, the **Deploy to FlowForce** command will be active, but clicking it will prompt you to save the SPS file as a PXF file. To create a PXF file from an SPS file, use the **File | Save As** command and select PXF as the format to save as.)

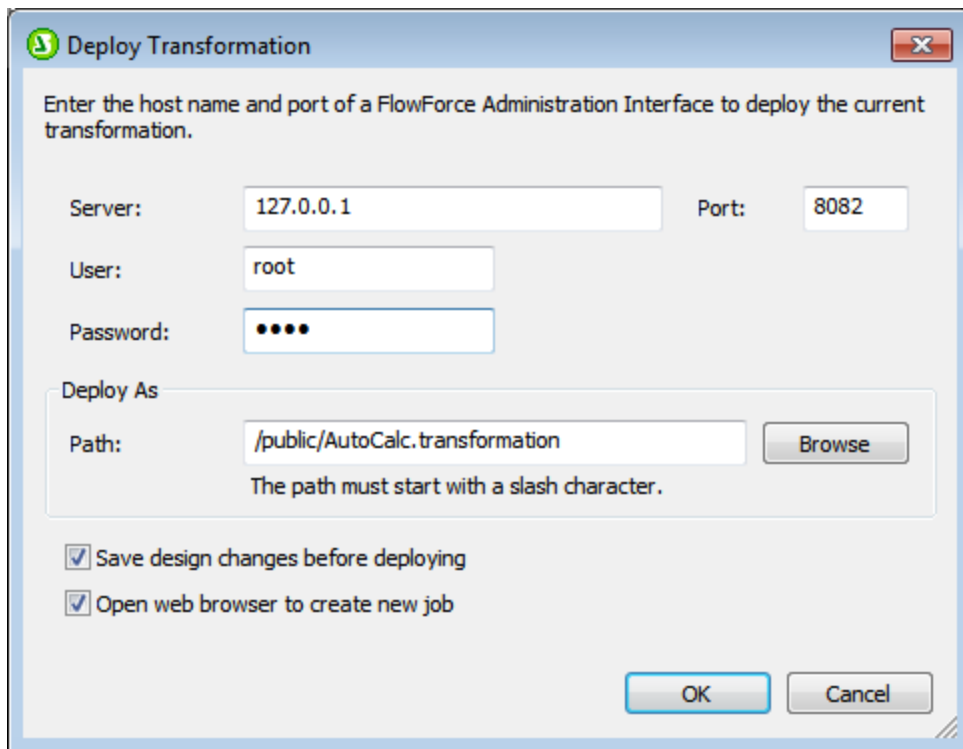
Note: When a PXF file is saved, an option is provided for including external files (such as image files) in it. If an external file is not included in the PXF file but is required for the transformation, then the external file must be saved on the FlowForce Server. Since the external files will be accessed from the working directory (specified in the FlowForce Server job definition), they should be placed relative to the working directory, in such a way that links originating in the working directory will correctly access them.

Note: Designs containing a database schema source are currently not supported, and cannot be deployed to FlowForce Server.

Note: When a FlowForce Server job requiring a StyleVision transformation is executed, the job is passed to StyleVision Server, and StyleVision Server will extract the contents of the PXF file to the working directory that was specified in the job's parameters. To ensure that there is no filename collision when this extraction occurs, there should be no file in the working directory that has the same name as a file contained in the PXF file.

Before running the **Deploy to FlowForce** command in StyleVision, make sure that Altova FlowForce Server and Altova StyleVision Server are correctly licensed and running. (StyleVision Server is packaged with FlowForce Server.)

The **File | Deploy to FlowForce** command in StyleVision pops up the Deploy Transformation dialog (*screenshot below*).



In this dialog, you specify the following:

- The address and port number of the FlowForce Web Server (not the FlowForce Server), together with access details (user and password) for the FlowForce Server.
- The filename of the transformation file and the location on the FlowForce Server where it is to be saved. The filepath must start with a slash, which represents the root directory of the FlowForce Server.
- If changes have been made to the design since the file was last saved, the *Save design changes before deploying* check box will be enabled. Check the box if you wish to save these changes; otherwise uncheck the box.

On clicking **OK**, the `.transformation` file is deployed to the FlowForce Server at the location specified. If you have checked the *Open web browser to create new job* check box (see *screenshot above*), a web browser is opened in which the draft job created during the deployment step can be edited (see *screenshot below*). The next section describes how to configure a transformation job.

Configuring the transformation job

The transformation job specifies the name of the job, the input files and parameters for the transformation, the output files of the transformation, triggers and security credentials.

The screenshot below shows the top half of the job configuration page.

ALTOVA®
flowforce®
SERVER 2013

Home Configuration Log Administration Help

Create job in / public /

Job name: AutoCalc.transformation.job

Job description:

Job input parameters

Execution Steps

Execute function /public/AutoCalc.transformation

Parameters:	InputXml:	altova://packagedfile/Data.xml	as xs:string (required)	Set to
	OutHtml:	AutoCalc.html	as xs:string (optional)	Set to
	OutRtf:			
	OutPdf:			
	OutDocx:			
	Working-directory:		as xs:string (required)	Set to

Notice the following points:

- The **job name** is `AutoCalc.transformation.job`. You could give the job any name you like.
- **Execution:** A function is being executed, namely the transformation `AutoCalc.transformation` that was deployed to the FlowForce Server from StyleVision. Compare the file name and location given when the transformation was deployed to FlowForce Server (see screenshot above).
- The **Working directory** is the location on the FlowForce Server where StyleVision Server will unpack the input files and save the output file.

In the bottom half of the job configuration page, you set up the triggers for the job and/or set up the job as a service. The credentials are your user name and password on the current client machine.

The screenshot displays the FlowForce configuration interface, organized into four main sections:

- Triggers:** Includes a 'Run' dropdown set to 'daily', a frequency of 'every 1 day(s)', a 'Repeat' button with a plus sign, a 'Start' date/time selector (2013-04-11 13:53:00), an 'Expires' button with a plus sign, a 'Time zone' dropdown set to 'Europe/Berlin', and an 'enabled' checkbox. Below these are buttons for 'new Timer', 'new Filesystem trigger', and 'new HTTP trigger'.
- Service:** Features a checkbox 'Make this job available via HTTP at URL' followed by a text field containing 'http://<FlowForce server>/service/' and a button 'AutoCalc'.
- Credential:** Contains a section 'Run job using credential:' with two radio buttons: 'Select existing credential:' (unselected) and 'Define local credential:' (selected). The 'Define local credential' section includes a 'User name' text field with 'MyUserName' and a 'Password' field with a 'Change password' button.
- Queue settings:** Includes a 'Minimum time between runs' field set to '0 seconds' and a 'Maximum parallel runs' field set to '1 instances'.

At the bottom of the interface are 'Save' and 'Delete' buttons.

A trigger is an event that starts execution of the job. In the screenshot above, we have used a timer event as the trigger: The job will be executed once a day at the specified time. We have also specified that the job will be available as a service at the URL:

`http://<FlowForce Server>/service/AutoCalc`

To view the result of the job execution (in this case, the transformation), enter the URL of the service in a web browser. If, for example, the FlowForce Server is on the current machine and its port is 4646, then the URL for the service would be:

`http://localhost:4646/service/AutoCalc`

or

`http://127.0.0.1:4646/service/AutoCalc`

After you have finished configuring the job, click Save. The job will now be executed according to the triggers and when the service is called. You can view the log of FlowForce Server activity by clicking the **View Log** button at top left of the job configuration page.

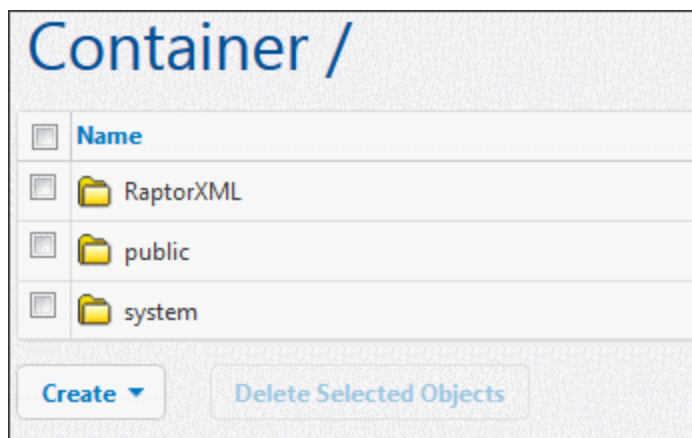
3.8 Using RaptorXML Server to validate a document

Aim: to validate an XML Schema file supplied with RaptorXML Server.

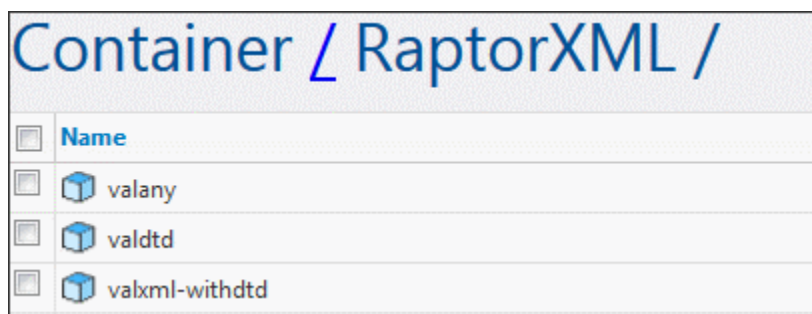
- This example uses the **address.xsd** file available in the .../RaptorXMLServer2013 folder (if you used the default installation path).

To create a new job:

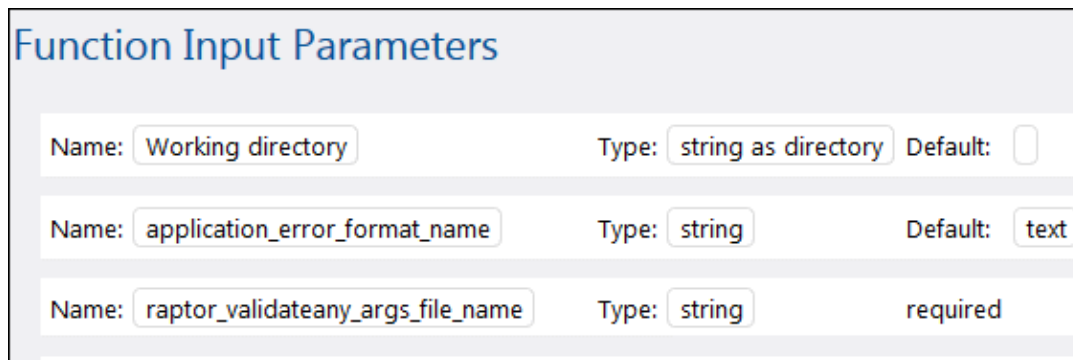
1. Click the "Configuration" button, then click the container "RaptorXML".



This opens the container and displays all the default functions available for RaptorXML.



2. Click the **valany** entry. This displays a window showing all the Function Input Parameters along with their types and default values.



3. Click the "Create Job" button at the bottom of the page. This creates a job with the default name "valany.job". Edit the job name if necessary.

Job name:

Job description:

Job Input Parameters

Execution Steps

Execute function

Parameters:

Working directory:	
application_error_format_name:	
raptor_validateany_args_file_name:	<input type="text"/>
core_error_limit_name:	
core_verbose_name:	
xml_user_catalog_path_name:	
xsd_import_strategy_name:	
xsd_mapping_strategy_name:	
xsd_xsi_schemalocation_strategy_name:	

= Assign this step's result to

The valany parameters are now visible on the page. Note that the **mandatory** parameter that you need to supply, is shown as an expanded field.

- Click in the **raptor_validateany_args_file_name** field and enter the path and file name of the file that you want to validate, e.g. C:\Program Files (x86) \Altova\RaptorXMLServer2013\examples\address.xsd.
 - Define a trigger that you want to execute the job, e.g. timer trigger daily at 12:00.
 - Enter the user credentials (or select one a predefined one).
 - Click save to save the job.
- After the allotted time has passed click the "View log" button to see the job execution details.

Log entry details: 2013-06-06 15:52:19

Date: 2013-06-06 15:52:19

Severity: INFO

Module: flowforce

User: root

Instance ID: 13

Message: Step RaptorXML.raptor.validateany completed with status: 0

file:///c:/Program%20Files%20(x86)/Altova/RaptorXMLServer2013/examples/address.xsd: result="OK"

If the validation process was successful you will see result="OK". If the file did not validate you will see "FAIL".

Note:

To see an explanation of all the RaptorXML functions and their associated parameters, please see: [RaptorXML Commands](#).

Chapter 4

User Guide

4 User Guide

Object System

Jobs, functions, triggers, etc. are stored in the object system inside the FlowForce Server database in a hierarchical structure.

The properties and capabilities of the FlowForce Server object system are similar to those of commonly used file systems. File systems use folders, while FlowForce Server uses containers.

Containers can have access permissions assigned to them, or inherit permissions from their parent container(s).

- The root of the object system is the "/" container (the **root container**) which can contain other containers, or other user-defined objects.
- The predefined container **"/system"** contains the system functions(s) and should not be used for user-defined objects.
- The predefined container **"/public"** is the default location to create user-defined objects like jobs, functions, credentials and other containers. You can of course create any number of containers.

Deploying mappings / transformations

Deploying a mapping means MapForce organizes all the mapping resources, used by the specific mapping, into a FlowForce Server function and passes it on to the server/machine running FlowForce, where it will be processed.

A deployed mapping function can then be used in a job execution step.

Jobs

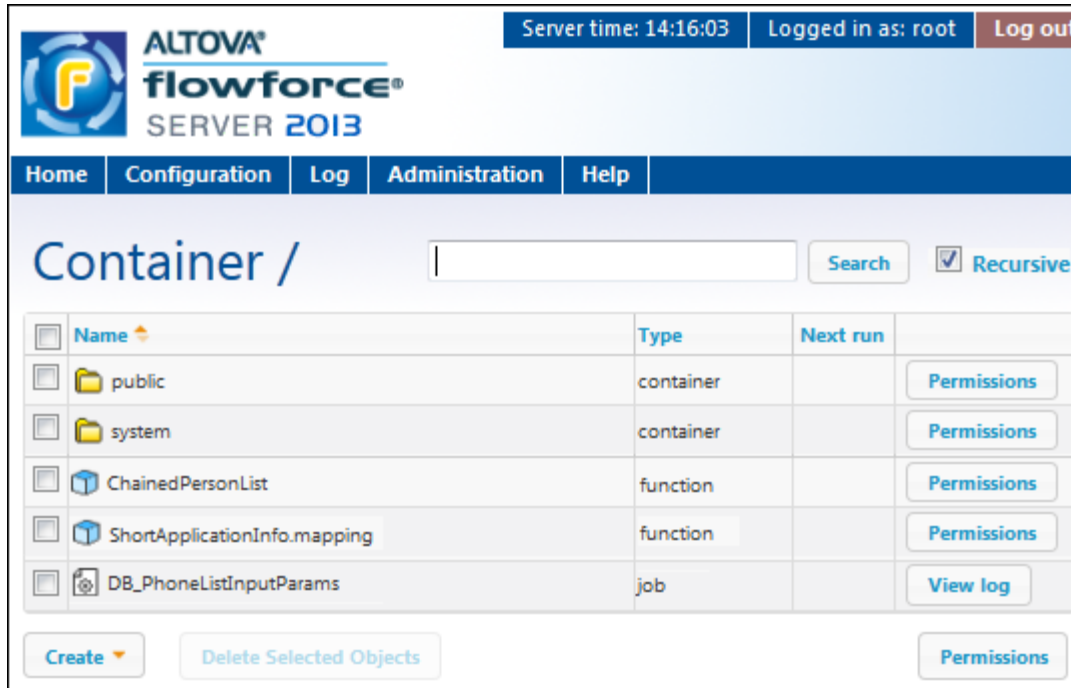
Jobs consist of [triggers](#), execution steps and various other settings. The triggers define when the job will be executed, and the execution steps define what the job actually does when it executes. Multiple triggers and execution steps can be defined per job.

Jobs can call other jobs allowing you to create subjobs.

Jobs can contain placeholder values that can be passed to the job at runtime. These placeholders are called "[Job input values](#)" and can supply values through a default, or through manual input via an HTTP client, e.g. internet browser.

4.1 FlowForce Administration Interface

The FlowForce Server Administration Interface is used to define jobs and triggers and to display the log table.



There are several menu items available in the browser window:

- **Home** shows you the connection details and any active jobs.
- **Configuration** shows containers and the objects they contain: jobs, credentials, functions, etc.
- **Log** shows you the server logs. The Log View contents can be configured.
- **Administration** shows you the Users, Roles, and Settings
- **Help** shows this help file

Configuration

FlowForce Server has a hierarchy of "items" visible in the Configuration page.

- The top-level is the "Container". Containers can contain "objects".
- Objects can be other Containers, Jobs, Instances, or Functions. Clicking the "Name" check box (top left corner of the table) selects/deselects all the objects in the list.
- Clicking a container displays the objects that it contains.
- Clicking a job displays the job definition page, showing the triggers, execution steps, and other settings that make up the job.

4.2 FlowForce concepts

Configuration

Configuration data in FlowForce Server's database are comprised of various objects that define the operation of FlowForce Server. This includes jobs, credentials, functions, triggers, and other objects.

Configuration objects are organized in a freely defined hierarchy of containers. Some configuration settings are edited together (e.g. jobs include triggers), and other settings can also be stored as standalone objects under their own name (e.g. credentials and functions).

Container

A container is similar to a folder in a commonly used file system. It is used to create a hierarchical structure for storing configuration objects and other containers. Containers can be assigned access permissions.

Two predefined containers exist in FlowForce Server: /system which contains system functions, e.g. copy, move, etc., and /public which is the default container when deploying a mapping to FlowForce Server from MapForce. Other containers can be created as needed, e.g. for departments or user groups.

Function

A FlowForce Server function performs a specific operation when used in a job execution step. It may have input parameters that need to be passed to it by the caller. Available functions include the [system functions](#) delivered with FlowForce Server, deployed MapForce mappings or StyleVision transformations, and the execution steps of other jobs.

Job

A Job consists of Triggers, Execution steps, input parameters, and other settings. Triggers define when a job will be executed, and the execution steps define what the job actually does when it executes. Multiple triggers and execution steps can be defined per job.

Trigger

Triggers define under which circumstances a job will be executed. Three types of triggers can currently be defined: [Timer triggers](#), [File system triggers](#), and [HTTP triggers](#). Multiple triggers can be defined per job.

Service

FlowForce Server permits exposing jobs as web services via the HTTP protocol. This allows interactive or automated access to these jobs.

Credential

Credentials are stored login data used to execute FlowForce Server jobs. Credentials can be defined as standalone "objects" and be assigned to various jobs, or they can be manually entered for a specific job.

Queue

The queue settings in a FlowForce Server job allow limiting the number of parallel job executions to control use of server resources.

Access Control

All important operations in FlowForce Server are linked to permissions or privileges which need to be assigned to the user to successfully execute them.

User

FlowForce Server users are persons that have been added to FlowForce Server by the FlowForce Server administrator with a login name and a password. Depending on the assigned rights and privileges, users can define FlowForce Server jobs, deploy mappings, or view logs.

Two special users are predefined by FlowForce Server: "root" is the initial administrator user, and "anonymous" is a special user account used for FlowForce Server services that should be available to users without explicit log in to FlowForce Server.

Role

Roles are used to manage privileges and object permissions for user groups as opposed to individual users.

Having defined users, you can assign them to a role thus creating user groups. The users become "members" assigned to the specific role.

Permission

Permissions control access to containers and configurations. Unlike privileges they can be redefined on every level of the container hierarchy, and are by default inherited from parent containers.

Permissions, like privileges, are inherited from all roles the user is a member of, as well as from permissions directly assigned to the user.

Privilege

Privileges control user rights globally. This means privilege settings cannot be overridden in the container hierarchy of FlowForce Server.

When a user logs into FlowForce Server, the set of effective privileges is determined by the user privileges and all role privileges the user is member of.

4.3 Job configuration

The job page lets you define how the job will be processed and when it will be executed. The triggers define when the job will be executed, and the execution steps define what the job actually does when it executes. Multiple triggers and execution steps can be defined per job.

You can create a job from [scratch](#), or you can [deploy a mapping](#) from MapForce (or a transformation from StyleVision) that generates a partially filled in job in FlowForce Server.

Jobs can be created in any container/folder where you have sufficient access permissions.

Structure of a job

The job name and job description fields make it easy to administer your jobs.

[Job input parameters](#) are placeholders for values (or files) that can be supplied at runtime.


[Execution Steps](#) define what the job does when it runs.

[Triggers](#) let you define how, or when, a job will be executed. The following triggers are currently available: Timer triggers, File system triggers, and HTTP triggers.

The [Service](#) option makes it possible to access jobs as webservice via the HTTP protocol.

[Credentials](#) are stored login data that define the operating system user account used to execute FlowForce Server jobs.

[Queue settings](#) let you define how many parallel jobs can run in parallel and the minimum time between runs.

 **ALTOVA® flowforce®**
SERVER 2013

Home Configuration Log Administration Help

Create job in [/](#)

Job name:

Job description:

Job Input Parameters

[+](#)

Execution Steps

[new Execution step](#) [new Choose step](#) [new For-each step](#) [new error/success handling step](#)

Triggers

[new Timer](#) [new Filesystem trigger](#) [new HTTP trigger](#)

Service

☐ Make this job available via HTTP at URL `http://<FlowForce server>/service/`

Credential

Run job using credential: ☐ Select existing credential:

☒ Define local credential: User name:

Password:

Queue settings

Minimum time between runs: seconds

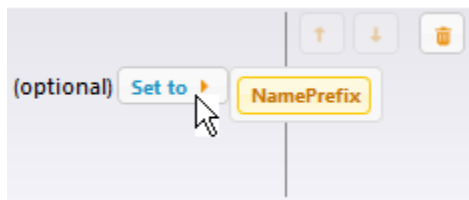
Maximum parallel runs: instances

4.3.1 Job input parameters

Job input parameters are placeholders for values (or files) that can be supplied at runtime.

Name	Type	Default	Description
NamePrefix	string		

Creating an input parameter, e.g. NamePrefix, automatically makes it available for selection in the "Set to" popup window, of any of the execution step parameters.



For an example please see: [Using a deployed mapping as a web service](#), or [Using parameters to query a database](#).

When adding a file system or HTTP trigger to a job, FlowForce Server automatically creates a parameter named "triggerfile" that contains the name of the file that activated the trigger.

Type: string

Use the string type for most parameters that you use.

Type: stream

Allows you to select files when using the job as a service.

Default

Allows you to specify a default value for the parameter that is used when no value is passed to the job at runtime.

Job parameter values are determined at runtime as follows:

- File system and HTTP triggers set the parameter "triggerfile".
- If a job is called as service via HTTP, all parameters are passed from the HTTP request. Please see the [Service](#) section for more information.
- If a job is called from an execution step in another job, parameters are passed from the step definition in the calling job.
- Parameters that are not supplied any value get the default value defined in the "Default" field.

4.3.2 Execution steps

The execution steps define what actions the job executes when it is started by a trigger or an HTTP request.

A job can have one or more execution steps. The list of execution steps is processed from top to bottom. FlowForce Server can also execute job steps conditionally and can use expressions to define flow control sequences.

[New Execution step](#)

Lets you execute specific system calls, MapForce mappings, or StyleVision transformations, that were deployed to the respective server.

[New Choose step](#)

Lets you define specific conditions that will apply to individual job steps.

[New For-Each step](#)

Lets you execute one or more job steps repeatedly.

[New error/success handling step](#)

Lets you define actions (cleanup actions, notifications/emails) depending on the final state of a the enclosed steps, i.e. on error, on success, or always.

Execution step

The standard "Execute function" lets you execute a specific FlowForce function. Available functions include the [system functions](#) delivered with FlowForce Server, deployed MapForce mappings or StyleVision transformations, and the execution steps of other jobs.

Clicking in the "Execute function" field opens a popup from which you can select a [built-in function](#) from the /system container, or deployed mappings/transformations from any other container. You can also select another job here, in which case the job steps of the selected job will be executed as subjob.

Execute function:

The combo box **initially** displays a popup window allowing you to select system functions, or MapForce mappings (StyleVision transformations) from the configuration containers.

MapForce mapping

Parameters for MapForce actions are defined by the specific mappings.

Parameters are defined by:

- input components
- in/out components
- output components

Parameters allow you to **override** the input and output file names that were defined when the mapping was deployed from MapForce. When the job executes, the new files will be used instead of the ones defined in the mapping. Note that all file paths in job execution steps must

be a path on the server machine (that runs FlowForce), not on your local machine.

E.g.

Using the deployed ChainedPersonList as an example:

- Employees (input)** is the source component with Employees.xml as the instance document.
- PersonList (in/out)** is the intermediate document and therefore shown as in/out, because it is used as both a source and target document in the chained mapping.
- Contacts (output)** is the target document

Execution Steps

Execute function /public/ChainedPersonList.mapping

Parameters:	Employees:	(input)		
	PersonList:	(in/out)		
	Contacts:	(output)		
	Working-directory:	c:\temp	as string (optional)	

= Assign this step's result to

To change the source/destination files:

Click the "+" button next to the list of parameters to expand the optional fields.

Execution steps

Function: /public/ChainedPersonList (MapForce mapping)

Parameters:	Employees:	(input)		:dfile/C:/Employees.xml	as xs:string (optional)		
	PersonList:	(in/out)	<td></td> <td></td> <td></td> <td></td>				
	Contacts:	(output)	<td></td> <td></td> <td></td> <td></td>				
	Working-directory:	c:\temp	as string	(optional)			

Click in the Employees (input) field and delete the "altova://packagedfile/C:/Documents and Settings.../Employees.xml" file and replace it with the file you want to use instead (e.g. PersonList.xml).

Execution steps

+

Function: /public/ChainedPersonList (MapForce mapping) ▼

Parameters:	Employees:	(input)		C:/PersonList.xml	as xs:string (optional)	<div style="text-align: right;"> Set to ▶ </div>
	PersonList:	(in/out)		<div style="text-align: center;">+</div>		
	Contacts:	(output)		<div style="text-align: center;">+</div>		
	Working-directory:			c:\temp	as string (optional)	<div style="text-align: right;"> Set to ▶ </div>

Note:

Any path starting with "**altova://packagedfile/**" refers to the file content that was **deployed** together with the mapping, and not to the current version of that file in any path on the server.

Acting on files that cause the trigger to fire

If you create a "File system trigger" (by clicking the "new File system trigger" button) this automatically adds the "triggerfile" entry into the Job input parameters field.

Job Input Parameters

+

Name:	triggerfile	Type:	string
-------	--	-------	---

+

Select the deployed mapping file you want to use in the Execute function field (e.g. ShortApplicationInfo.mapping).

Clicking the "+" icon to the right of SectionedPage creates an edit field.

Click the Set to ▶ button to the right of the expanded SectionedPage field, and select "triggerfile" entry.

Execution Steps

+

▲

Execute function /public/ShortApplicationInfo.mapping

Parameters:	SectionedPage:	(input)		{triggerfile}	
	ShortInfo:	(output)		<div style="text-align: center;">+</div>	
	Working-directory:			c:\temp	

This causes the input field contents to change to {triggerfile}.

The file, in the directory being polled, will then be used as the input file for the execution step.

Note:

Using a file that caused the trigger to fire, does not work with time-based triggers.

Please see: [Directory change - act on trigger file](#)

Function - Subjob

Once a job has been defined it can be used in an execution step of another job as a subjob. In the screen shot below, the subjob is added as an extra execution step, by clicking the Function combo box and selecting /public/copy2archive. If the selected subjob has job input parameters, they appear below the function and can be filled with values.

Execution Steps

+

Execute function

/public/ChainedPersonList.mapping

Parameters:

Employees:	(input)		+
PersonList:	(in/out)		+
Contacts:	(output)		+
Working-directory:			+

= Assign this step's result to

+

Execute function

/public/copy2archive

= Assign this step's result to

Choose step

The "new Choose step" button allows you to define conditions under which specific job steps should be executed.

The condition [expression](#) is entered in the "When" field and the step(s) to be executed can be selected by clicking the "+" button (below the When field). This opens a popup allowing you to select the specific step to insert: Execution step, Choose step, For-each, or error/success handling step.

Any number of conditions can be defined, and as soon as one of the conditions is true, that conditional step is executed and the remaining conditions of that conditional group are ignored. The execution sequence is then continued with the step following the successful conditional group.

Execution Steps

Execute function

/public/ChainedPersonList.mapping

Parameters:

Employees:

(input)

+

PersonList:

(in/out)

+

Contacts:

(output)

+

Working-directory:

c:\temp

=

Assign this step's result to

name

Choose

When

contains('Contacts.xml', 'Contacts')

Execute function

/system/filesystem/copy

Parameters:

Source:

Contacts.xml

Target:

c:\archive

Overwrite:

+

Working directory:

c:\temp

The "Otherwise" group lets you define what will be executed if all the previous choices fail.

Otherwise

Execute function

/system/mail/send

Parameters:

From:

ames@pool.com

To:

control@far.com

Subject:

Bad news

Message body:

File is unavailable, sorry not according to plan. Thanks


Attachment:

+

=

Assign this step's result to

name

Note: Placing the mouse cursor on the arrow-up,  or arrow-down icons to the right of the block allow you to move each individual step within the step group. The step to be moved is also highlighted when the mouse is over the icon.

For-each step

The "For-each step" button allows you to repeat an execution step any number of times.

The variable/counter is entered in the "For each" field, while the sequence to iterate is entered as an [expression](#) in the "in sequence" field.

The execution step to execute repeatedly is selected by clicking the "+" button (below the For-each step field). This opens a popup allowing you to select the specific condition: Execution step, Choose step, For-each step, or error/success handling step.

E.g

A new "For-each step" step was added after the execution of the ChainedPersonList.mapping job step. It iterates through all the XML files in the c:\temp folder and copies them to a different directory.

For-each <i>file</i>	in sequence	<i>list-files('c:\temp*.xml')</i>
Execute function is		/system/shell/commandline
Command is		copy {file} c:\archive\

Error/success handling step

The "error/success handling step" button allows you to define specific actions that should take place upon step completion.

Step completion can mean: success, failure, or always. This allows you to perform cleanup actions after step completion and generate notifications if errors should occur.

Whenever a error/success handling process is complete, irrespective of whether it was successful or not, the result of the execution is checked. If the execution result matches the value selected in the combo box (on error, on success, or always) then the notification or cleanup block, i.e. below the "do" keyword is executed.

In the example below the ChainedPersonList.mapping is executed and the result is compared to the "on success" selection in the combo box. If the job completed successfully the system function .../mail/send is executed and a mail is sent to the person entered in the "To" field.

Execution Steps

Execute function

/public/ChainedPersonList.mapping

Parameters:

Employees:

(input)

+

PersonList:

(in/out)

+

Contacts:

(output)

+

Working-directory:

c:\temp

=

Assign this step's result to

name

Execute with error/success handling

On success

do

Execute function

/system/mail/send

Parameters:

From:

ronnie.biggs@strains.com

To:

fedshed@fedres.com

Subject:

Success

Message body:

Mapping completed successfully and the files were archived.

Attachment:

+

=

Assign this step's result to

name

new error/success handler

To define a error/success handling step:

1. Click the "error/success handling step" button of the Execution Steps group.

Execution Steps

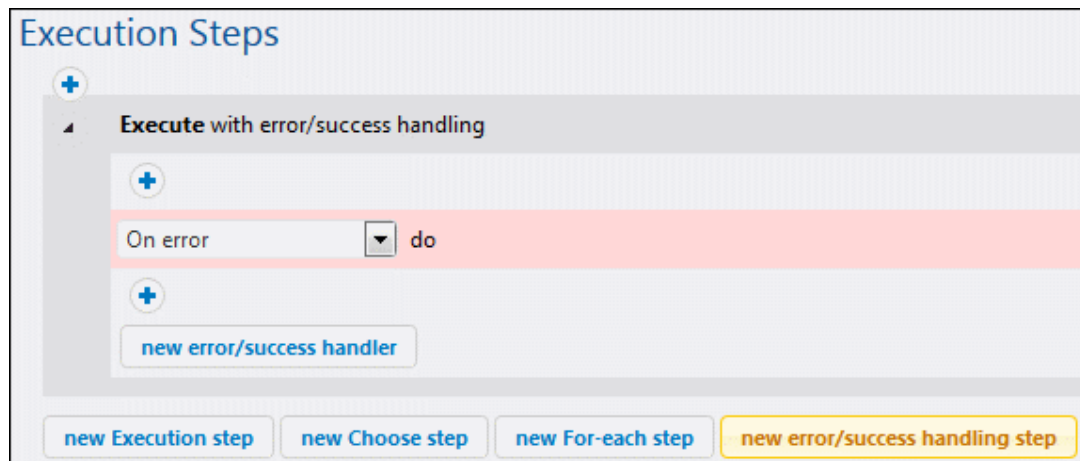
new Execution step

new Choose step

new For-each step

new error/success handling step

This creates an Execution Steps group as well as a notification group.

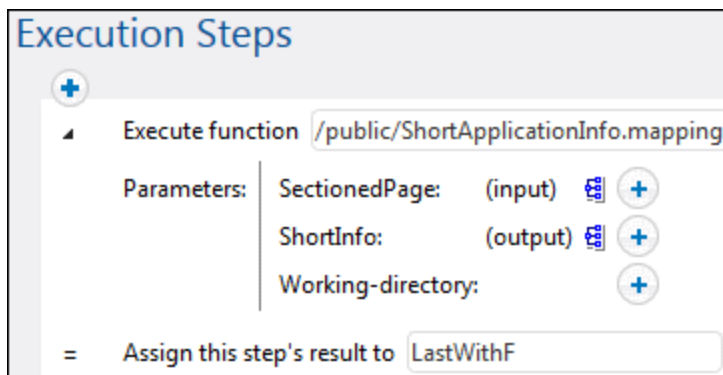


2. Click the top "+" button to create the Execute group.
3. Click the lower "+" button to create the Execute function of the notification group.
This is what will be executed if the result of the execution function is the same as the selection in the combo box, e.g. "on error" in the example above.

Step results

Step results are variables defined by the name entered in the "Assign this step's result to" field.

The "Assign this step's result to" field, available in most job steps, is used to assign the output/result of that step to a variable entered in this field. This result/variable can then be used with specific step result functions by any of the following steps in the Parameter fields or in other expressions.



Please see: [Step result functions](#) for list of the expressions that can be used together with step results.

Step expressions

Step Result Functions

Step result functions are applicable to results from steps like the command line step. To obtain a result, use the name entered in a previous step's "Assign this step's result to" field. You cannot apply step result functions to the result of steps which do not provide it, doing so gives an error during saving the job.

stdout

stdout(result) of type stream

Returns the standard output of result, fails if the result does not provide standard output.

Example: If you use the /system/shell/commandline step with the command
cmd.exe /c echo "hello world"

Then the stdout of this step result is a stream containing "hello world".

stderr

stderr(result) of type stream

Returns the standard error of result, fails if the result does not provide a standard error.

exitcode

exitcode(result) of type number

Returns the exit code of result, as returned by the executed program.

results

results(result, name) of type stream

results(result) of type stream

Returns the a list of result streams of the specified result, optionally filtered by name. Use the function "nth" to access a particular one.

Stream Functions

Streams can be passed to FlowForce Server via the service interface, and also during job execution within the various step results.

as-file

as-file(stream) of type string

The stream is stored in a file and the function returns the name of the temporary file.

This is useful to pass the output of one step as a parameter to another: If MapForce component

CompletePO produces a result and you intend to further process it via some other tool, you can use {as-file(results(MapForceMapping, "CompletePO"))} as the command line.

Note:

It is also possible to pass a stream from a Job Input Parameter to any **function** that expects a file name. In the [tutorial example](#) a Job Input Parameter named **Expenses** (of type stream) has been defined. This input parameter is then used with the function "as-file" to define the input parameter of the MapForce mapping:

```
{as-file(Expenses)}
```

content

content(stream, encoding = 'UTF-8') of type string

Reads the content of the specified stream as text in the specified encoding.

File System Functions

File system functions permit access to the file system. Note: The access restrictions of the specified user credential always apply.

list-files

list-files(path) of type "list of string"

Lists the files in the path (which may terminate with a wildcard) and returns the resulting string list.

If the path does not end with a path separator and is not a wildcard, a search is made for exactly the specified item in the parent directory.

list-directories

list-directories(path) of type "list of string"

Lists the subdirectories in the path (which may terminate with a wildcard) and returns the resulting string list.

read-lines

read-lines(path) of type "list of string"

Reads the lines from the given file and returns them as a list of strings.

List Functions

List functions are used to create and disassemble lists. Lists always contain items of a single type, e.g. only strings, only number, or only nested lists with the same item type, there are no mixed type lists.

list

list(item1, item2, ...) of type list

Builds a list from single items. All items must be of the same type, the resulting list is a list of items of that type.

join

join(list of lists, separator = empty list) of type list

Concatenates the lists given by the first argument using the second argument as separator between each pair of lists.

nth

nth(list, index) of type item

Returns the specified item from the list. The index is zero-based. Fails if the index is out of bounds.

from-to

from-to(from, to) of type list of number

Produces the list of integers between "from" and "to" inclusive. If from > to, this list is empty.

length

length(list) of type number

Returns the number of items in the list.

String Functions

The string functions deal with basic string operations.

concat

concat(string1, string2, ...) of type string

Concatenates/joins all of the separate strings into one string.

Same as string-join(list(string1, string2, ...)).

string-join

string-join(list of strings, separator = an empty string) of type string

Joins the "list of strings", inserts the separator in between each.

number

number(string) of type number

Computes the number representation of the string, i.e. converts the string into a number.

string

string(number) of type string

Computes the string representation of the given number, i.e. converts the number into a string.

split

split(string, separator) of type list of string

Splits the string at each occurrence of separator.

find-all

find-all(string, pattern) of type list of string

Extracts all occurrences of pattern in the string, where pattern is a regular expression.

trim

trim(string) of type string

Removes leading and trailing whitespace from the string (space, tab, linefeed, carriage return, form-feed and vertical tab).

trim-start

trim-start(string) of type string

Removes leading whitespace, (see trim).

trim-end

trim-end(string) of type string

Removes trailing whitespace, (see trim).

contains

contains(string, substring) of type boolean

Returns true if the first string contains at least one occurrence of substring, otherwise false.

starts-with

starts-with(string, start) of type boolean

Returns true if the first string starts off with the string "start".

ends-with

ends-with(string, end) of type boolean
Returns true if the first string ends with the string "end".

string-length

string-length(string) of type number
Returns the number of characters in the string.

substring

substring(string, start, end = string-length(string)) of type string
Returns the specified substring. Start and end are zero-based character positions.

Boolean Functions**not**

not(boolean) of type boolean
Returns the negation of the boolean.

all

all(boolean1, boolean2, ...) of type boolean
Returns true if all boolean values are true; stops evaluation after the first false value and returns false.

any

any(boolean1, boolean2, ...) of type boolean
Returns true if any boolean value is true; stops evaluation after the first true value. Returns false if all values are false.

if

if(boolean, valueTrue, valueFalse) of type ...
Returns valueTrue if the boolean is true, and valueFalse if false. Only the selected subexpression is evaluated. Both subexpressions must be of the same type, which is also the return type.

Example: To pass a boolean as XML Schema conformant value, use

```
if(b, "true", "false")  
or  
if(b, "1", "0")
```

true

true() of type boolean
Returns true.

false

false() of type boolean
Returns false.

Operators

Basic mathematical operators can be used to work on strings and numbers.

$a == b$

checks if a and b are equal (numerically equal for numbers, code-point equal for strings)

$a != b$

is equivalent to not ($a == b$). $a <> b$ is another variation.

$a < b$

checks if a is less than b (numerically less for numbers, see below for strings)

$a >= b$

is equivalent to not ($a < b$)

$a > b$ is equivalent to $b < a$

$a <= b$ is equivalent to $b >= a$.

String comparisons are performed as follows:

- The common prefix of the two strings are ignored (evaluated on codepoints)
- If both remaining strings are non-empty, their first codepoints are compared numerically
- Empty strings are less than non-empty strings

Operators on numbers:

$a + b$, $a - b$, $a * b$, a / b

All compute the normal arithmetic results.

4.3.3 Triggers

Three types of triggers can currently be defined: Timer, File system triggers, and HTTP triggers.

[Timer](#) trigger

This type of trigger allows you to schedule your jobs. Timer triggers have a Start date/time, Expire date/time and period of recurrence (daily, weekly etc.).

[File system](#) trigger

This type of trigger lets you check a specified directory, as well as check the content of a specific file(s).

[HTTP](#) trigger

This type of trigger lets you poll a specified URI for changes.

You do not need to define any triggers if you intend to make the job available as a [service](#) via HTTP.

Timer trigger types & common settings

Multiple triggers can be defined per job allowing maximum flexibility. This means that several triggers can be active simultaneously, and that whenever any of the triggers is fired, all execution steps of the job associated with the trigger are processed.

Triggers

Run: every 1 day(s)

Repeat: every 60 minutes from 08:00:00 to 20:00:00

Start: 2012-08-10 12:00:00

Expires: 2012-08-16 11:00:00

Time zone: Europe/Berlin

☒ enabled

[new Timer](#) [new Filesystem trigger](#) [new HTTP trigger](#)

Run

The "Run" combo box allows you to define the specific days that the trigger can be activated. This option only refers to dates! There is no time component when you make this selection.

Run period options:

[Once](#)

[Daily](#)

[On days of week](#)

[On days of months](#)

[On days of weeks of months](#)

Common trigger properties:

Repeat

The Repeat options define the interval between successive trigger firings, per day. The days when this will occur, are defined by the selection made with the "Run" combo box.

The "every" field lets you define the period between the job runs, in minutes.

The "from" and "to" fields define the time range between which the triggers will fire.

Start - date and time

Start date/time entries are only mandatory for a timer trigger that uses the "Run - Once" option.

Start date/time are optional for file system and HTTP triggers. Clicking in the Date field opens a pop-up calendar from which you can select the start date.

E.g. Repeat every "60" minutes from "08:00" to "20:00" with start time at 09:33. This means that the trigger will become active at 09:33, and as the repeat interval is 60 minutes, it will fire for the first time at 10:00, with repeats at every full hour.

Expires date - time.

The Expires fields allows you to define the date/time from when the job is to expire. The trigger will not fire after this date/time.

Time Zone

This field allows you to define timers that will fire at the same time of day, even if there are daylight saving time switches. Clicking in the field opens a pop-up time zone picker. The default time zone is defined in the server administration settings.

Enabled

The "enabled" check box allows you to enable/disable each individual trigger in the trigger list. This option is useful when creating and testing new jobs.

Waste basket 

The waste basket allows you to delete the whole trigger, or any of the sub elements that are part of it.

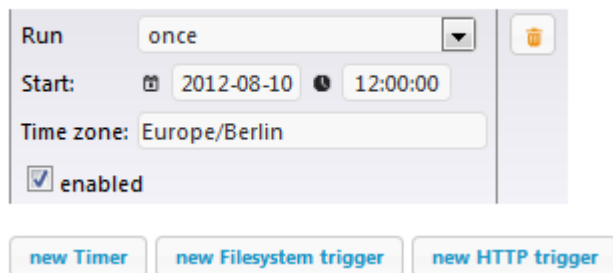
Note: Triggers and defaults



If you use job parameters with triggers, make sure that all parameters have defaults or the job will not execute.



Run Once

This type of trigger will fire once on the day specified, at the exact time given in the time field.

Triggers



Run  

Start:  

Time zone:

☒ enabled

[new Timer](#) [new Filesystem trigger](#) [new HTTP trigger](#)

Run Daily

This type of trigger will fire every day between the dates specified, with the first firing at 12:00, and repeat every full hour.

Triggers

The screenshot shows the configuration for a 'Run Daily' trigger. The 'Run' dropdown is set to 'daily'. The 'Repeat' section is set to 'every 60 minutes from 08:00:00 to 20:00:00'. The 'Start' date is '2012-08-10' at '12:00:00'. The 'Expires' date is '2012-08-16' at '11:00:00'. The 'Time zone' is 'Europe/Berlin'. The 'enabled' checkbox is checked. There is a trash icon in the top right corner.

[new Timer](#)
[new Filesystem trigger](#)
[new HTTP trigger](#)

Run On days of week

This type of trigger will fire every week on Tuesday and Thursday between the dates specified. The first time it fires will be at 12:00, and repeat every full hour.

Triggers

The screenshot shows the configuration for a 'Run On days of week' trigger. The 'Run' dropdown is set to 'on days of week'. The 'Repeat' section is set to 'every 60 minutes from 08:00:00 to 20:00:00'. The 'Start' date is '2012-08-10' at '12:00:00'. The 'Expires' date is '2012-08-16' at '11:00:00'. The 'Time zone' is 'Europe/Berlin'. The 'enabled' checkbox is checked. The 'Days of week' section shows a table with columns for Mon, Tue, Wed, Thu, Fri, Sat, and Sun. The 'all' checkbox is unchecked, and the checkboxes for Tue and Thu are checked. There is a trash icon in the top right corner.

[new Timer](#)
[new Filesystem trigger](#)
[new HTTP trigger](#)

Run On days of months

This type of trigger will fire on the 1st and 15th every month between the dates specified. The first time it fires will be at 12:00, and repeat every full hour.

Triggers

Run on days of months

Days of month: ☐ all ☒ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 ☐ 8 ☐ 9 ☐ 10 ☐ 11 ☐ 12 ☐ 13 ☐ 14 ☐ 15 ☒ 16 ☐ 17 ☐ 18 ☐ 19 ☐ 20 ☐ 21 ☐ 22 ☐ 23 ☐ 24 ☐ 25 ☐ 26 ☐ 27 ☐ 28 ☐ 29 ☐ 30 ☐ 31 ☐ last

Months: ☒ all ☐ Jan ☐ Feb ☐ Mar ☐ Apr ☐ May ☐ Jun ☐ Jul ☐ Aug ☐ Sep ☐ Oct ☐ Nov ☐ Dec

Repeat every 60 minutes from 08:00:00 to 20:00:00

Start: 2012-08-10 12:00:00

Expires: 2012-08-16 11:00:00

Time zone: Europe/Berlin

☒ enabled

[new Timer](#) [new Filesystem trigger](#) [new HTTP trigger](#)

Run On days in weeks of months

This type of trigger will be fired on Monday, and Wednesday of every second week, of every month between the dates specified. The first time it fires will be at 12:00, and repeat every full hour.

Triggers

Run on days in weeks of months

Days of week: ☐ all ☒ Mon ☐ Tue ☒ Wed ☐ Thu ☐ Fri ☐ Sat ☐ Sun

Weeks of month: ☐ all ☐ 1 ☒ 2 ☐ 3 ☐ 4 ☐ last

Months: ☒ all ☐ Jan ☐ Feb ☐ Mar ☐ Apr ☐ May ☐ Jun ☐ Jul ☐ Aug ☐ Sep ☐ Oct ☐ Nov ☐ Dec

Repeat every 60 minutes from 08:00:00 to 20:00:00

Start: 2012-08-10 12:00:00

Expires: 2012-08-16 11:00:00

Time zone: Europe/Berlin

☒ enabled

[new Timer](#) [new Filesystem trigger](#) [new HTTP trigger](#)

File system trigger

File system trigger:

This type of trigger lets you check a specified directory, as well as check the content of a specific file(s). A directory can be checked for updated or new files (you cannot check for deleted files). Wildcards can be used to filter specific files of the directory.

Check - Modified date:

The trigger checks the last modification timestamp of all the specified files. If any dates have

changed, or a new file has been added, then the trigger fires.

Check - Content:

This option computes and stores a hash code for the specified file. After the polling interval has passed, the hash code is recomputed and compared to the stored value. If there is a difference then the trigger fires. Note that this can place considerable load on the server.

If any dates have changed, or a new file has been added, then the trigger also fires.

Wait 'S' seconds to settle:

Defines the time in seconds that the server will wait before starting the next job.

Triggers

Check: **Modified Date** of file or directory: **c:\temp*.xml** polling interval: **60**

Start: +

Expires: +

Time zone: **Europe/Berlin**

☒ enabled

When adding a file system trigger, FlowForce Server automatically adds the "triggerfile" parameter to the job. This parameter is set at runtime to the name of the file that triggered the job execution. This file name can then be passed to an execution step to process the file.

Job input parameters

+ Name: **triggerfile** Type: **string** Default:

+

HTTP trigger

HTTP trigger

This type of trigger lets you poll a specified URI for changes.

Check - Modified date: If a URI is being polled, then the "Last-Modified" HTTP header is checked. If the HTTP header is missing, then "Check Content" is used.

Check - Content: If a URI is being polled, then the "Content-MD5" optional header field is checked. This is a 128 bit "digest" used as a message integrity check. If the MD5 header has changed after the polling interval has passed, then the trigger fires. If the header is not provided by the server, the content is retrieved and hashed locally.

Triggers

Check of URI: polling interval: seconds.

Start:

Expires:

Time zone:

☒ enabled

[new Timer](#) [new Filesystem trigger](#) [new HTTP trigger](#)

When adding an HTTP trigger, FlowForce Server automatically adds the "triggerfile" parameter to the job. This parameter is set at runtime to the name of a temporary file that contains the downloaded content at the URI that triggered the job execution. This file name can then be passed to an execution step to process the file.

Job input parameters

Name: Type: Default:

4.3.4 Service

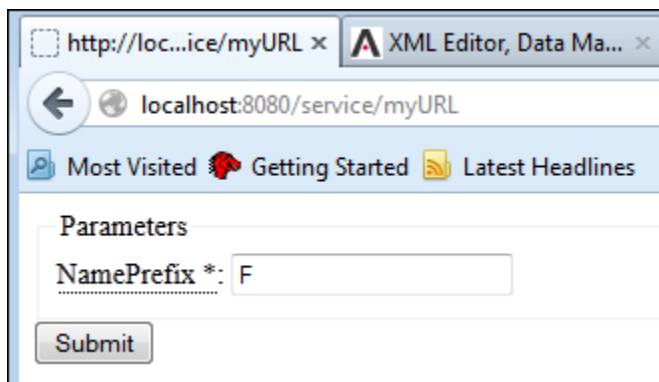
FlowForce Server permits exposing jobs as web services via the HTTP protocol. This allows programmatic and interactive access to these jobs, making it possible to use them on-demand.



All job parameters automatically become parameters for the service. If a job parameter does not have a default, it is mandatory and must be provided when invoking the service; otherwise it is optional and can be provided, taking the default value if it is not provided.

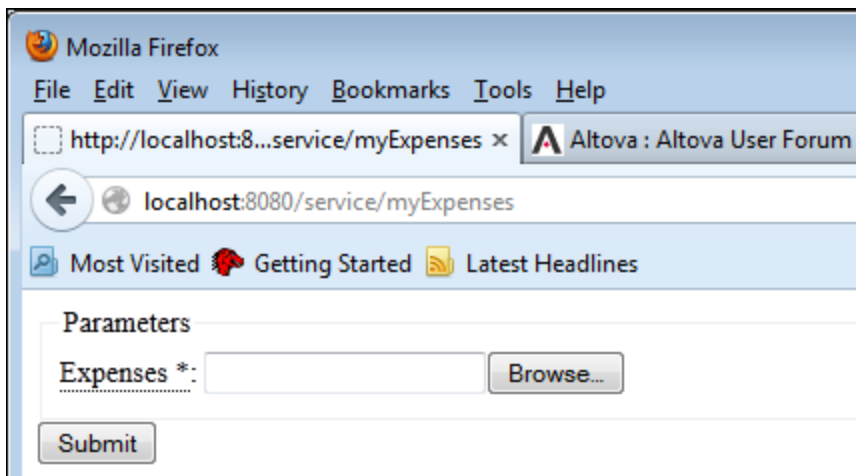
FlowForce Server checks if all mandatory parameters are provided when the service is invoked. If some are missing the service execution fails. For testing purposes FlowForce Server supplies a simple HTML form allowing parameters to be entered manually. Note: When using Internet Explorer 9 as your browser, please disable the option "Show friendly HTTP error messages" in the Advanced tab, to view the form.

This form allows you to enter a value for all parameters.



Please see the tutorial example: [Using parameters to query a database](#).

Streams need a file to be uploaded using the "Browse" button. When clicking "Submit" the data is transferred to FlowForce, and if accepted, FlowForce Server starts the job and waits for the result.



After the job has finished executing FlowForce Server delivers a response. If the job failed for any reason, FlowForce Server will return an error message, otherwise it will return the first result file of the last execution step executed, or the last execution step's standard output, if no result file is available.

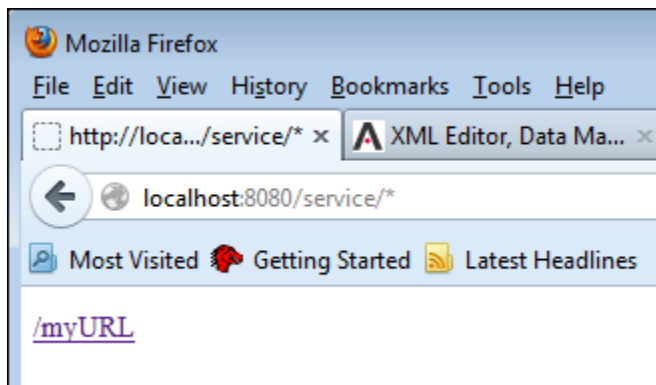
Services remain active as long as FlowForce server is running.

Please see the tutorial example: [Using a deployed mapping as a web service](#).

To access FlowForce Server services through your browser:

1. Open your browser and enter <http://localhost:4646/service/> in the URL text box. If you are using a remote FlowForce server installation, make sure it accepts connections from other machines.

This command shows all the services currently running on the server.



See also:

[Technical details](#)

Technical details

The service interface is primarily meant for machine consumption. At the request URL specified, FlowForce Server starts a listener service which accepts HTTP GET and POST requests.

It then runs the job execution steps specified and returns the first result file of the last step or

the standard output of the last step, if no result files are produced (e.g. for FlowForce Server system commands).

A valid result is returned with a HTTP 200 status, with the Content-Type header set according to the result.

The Content-Type header depends on the actual result. A MapForce mapping will result in text/xml if it has XML output, or text/plain for text output. Standard output of other functions is also returned as text/plain. The result is returned as the response body, without any embellishments.

Authentication:

FlowForce Server uses HTTP Basic authentication as the means of user authentication. If you want a service available without credentials, you have to grant "use service" permission to the default "anonymous" user. Any other user credentials are checked against the FlowForce Server user database (so you can use the same usernames and passwords you use for logging into the FlowForce Server configuration GUI).

You can still supply HTTP credentials when a service is available for anonymous use. The credentials are then checked against the FlowForce Server user database and the service execution is attributed to the authenticated user instead of user anonymous.

Invalid credentials

If you supply invalid credentials, the request interface will return an HTTP status of 401. If you did not supply credentials and service use has not been granted to anonymous on this service, the request interface will also return an HTTP status of 401.

If you supplied valid credentials, but the authenticated user is not granted "use" access on this service, the request interface will return an HTTP 4xx failure status. If you try accessing a service that does not exist, an HTTP 4xx failure status is returned.

When the client is permitted to use the service, FlowForce Server will verify the supplied request parameters against the defined parameters of the job. Every parameter that does not have a default must be specified, parameters having a default may also be specified. If parameter validation fails, FlowForce Server will return a 5xx HTTP status. For debugging and testing purposes FlowForce Server also returns a simple HTML parameter form in this case.

The built-in parameter named **showform**, when present (regardless of value), will display the testing HTTP form regardless of any parameter validation errors.

Requests can generally be sent as both HTTP GET or HTTP POST (with multipart/form-data Content-Type), with the exception of parameters of type **stream**, which are only supported for HTTP POST requests.

Service execution behaves like execution via trigger, and is subject to the same queue constraints. You should set the queue limits accordingly.

Execution errors are reported as HTTP 5xx status with a generic error message; detailed information can be found in the FlowForce Server log.

4.3.5 Queue settings

Each Job has a queue assigned to it allowing you to define how many instances of the same job can be run in parallel.

Queue settings

Minimum time between runs:

seconds

Maximum parallel runs:

instances

Maximum parallel runs:

Enter the number of times the same job may be executed in parallel on the server.

Minimum time between runs:

Enter the time in seconds that must pass after each of the parallel jobs starts before another one may start.

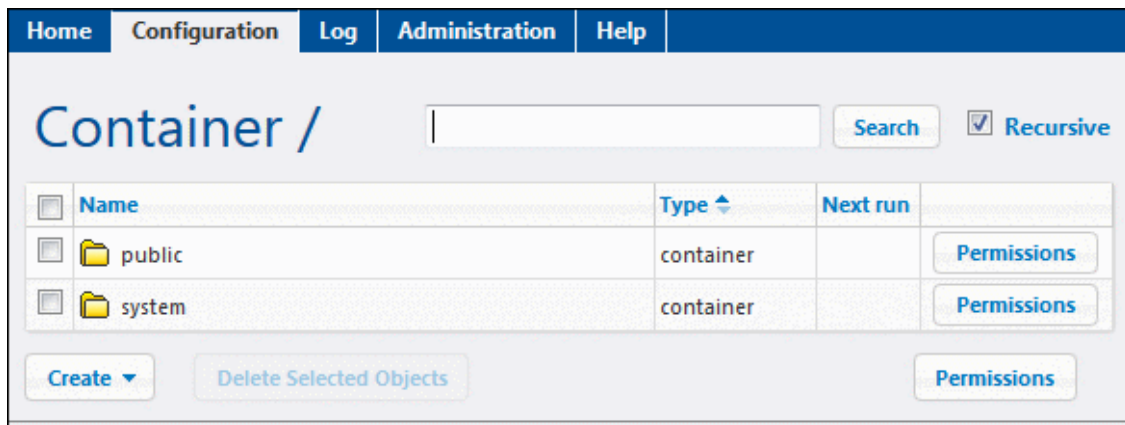
4.4 Credentials

Credentials are stored login data used to execute FlowForce Server jobs. Credentials can be defined as standalone "objects" and be assigned to various jobs, or they can be manually entered for a specific job.

Jobs are started automatically by FlowForce Server when the defined trigger conditions are met. FlowForce Server then runs these jobs using a specific operating system user account, ensuring that job steps do not access unauthorized data. Note that [file watch triggers](#) are also assigned credentials.

Credentials can be created, or deleted, on the Configuration (Container) page. Note that job credentials, i.e. username and password, can also be entered for individual jobs on the Job page.

Any user that has "write" access to the "Configuration" permission, can edit or remove credentials.



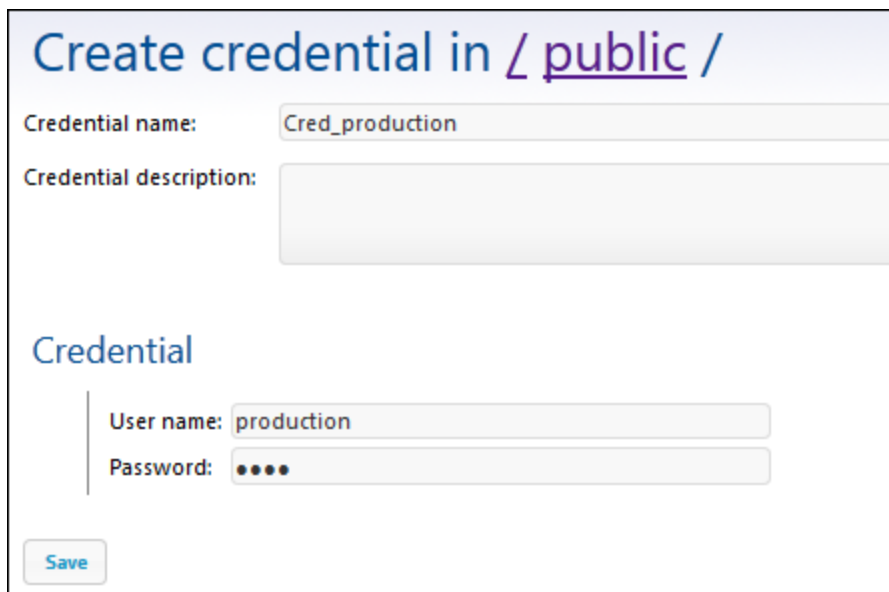
To add a credential to FlowForce Server:

1. Click the container you want to create the new credential in, e.g. public.
2. Click the "Create" button and select the "Create Credential" entry.



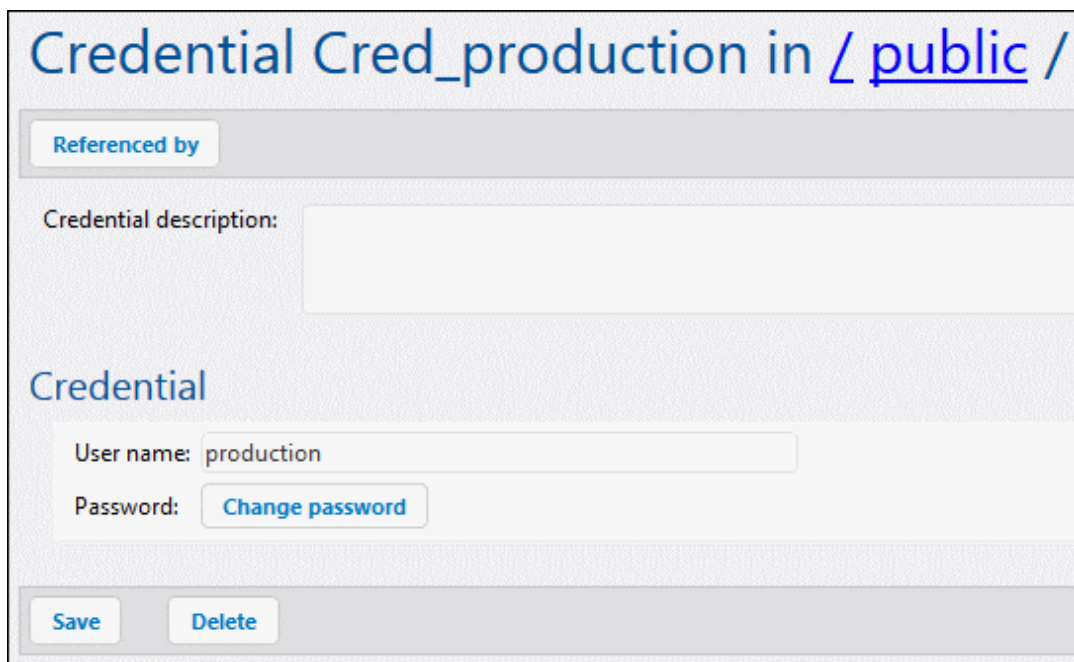
3. Enter the name of the credential as well as the **operating system** user name and password. To specify a user name in a Windows domain, please use the form

username@domain.



The screenshot shows a web form titled "Create credential in / public /". It contains two input fields: "Credential name:" with the value "Cred_production" and "Credential description:" which is empty. Below these is a section titled "Credential" with two sub-fields: "User name:" with the value "production" and "Password:" with four dots. A "Save" button is located at the bottom left of the form.

4. Click Save.
The new credential "Cred_production" has been saved in the /public container.
5. Click the "Configuration" button to return to the Container page.



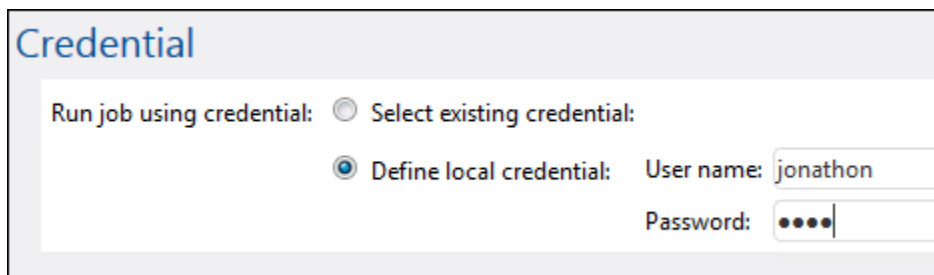
The screenshot shows a web page titled "Credential Cred_production in / public /". It features a "Referenced by" button at the top. Below it is a "Credential description:" field. The "Credential" section contains a "User name:" field with the value "production" and a "Password:" field with a "Change password" button. At the bottom, there are "Save" and "Delete" buttons.

Please see [Permissions](#) for information on the container permissions that can be defined.

Credentials and jobs

Every job **MUST** have a credential assigned to it for the job steps to be executed. This defines the **operating system** user account used to run the job steps.

A predefined credential can be selected using the "existing credential" combo box, or the local credential can be manually entered in the "User name" and "Password" fields. This is done on the "Create job..." page.



The screenshot shows a window titled "Credential". Inside, there is a section "Run job using credential:" with two radio buttons. The first radio button is labeled "Select existing credential:" and is unselected. The second radio button is labeled "Define local credential:" and is selected. To the right of the selected radio button, there are two input fields. The first is labeled "User name:" and contains the text "jonathon". The second is labeled "Password:" and contains five dots, indicating a masked password.

Note:

If you **manually** enter the user name and password, as shown above, you will have to update them for those specific jobs, whenever your server credentials are changed.

Credentials:

- Credentials can be created in any container a user has access to.
- The credential password may be an empty string.
- As the clear text password needs to be sent to the operating system's login function, passwords are stored in a reversible encrypted form in the FlowForce Server database. The administrator should make sure to restrict access to the FlowForce Server database file.

FTP credentials

When using FTP system functions, e.g. ftp/retrieve, in the "Execute function" field you must also enter your FTP login credentials for the FTP server. You can select an existing credential or enter a local one.

4.5 Built-in functions

The /system container contains various file system, ftp, and mail functions that can be used in job execution steps. This allows you to copy or move files, create directories, or execute arbitrary command lines.

Note that all file paths in job execution steps must be a path on the server machine (that runs FlowForce), not on your local machine.

[File system](#)

[FTP](#)

[Mail](#)

[Shell](#)

[Compute](#)

4.5.1 filesystem - File system functions

Note that all file paths in job execution steps must be a path on the server machine (that runs FlowForce), not on your local machine.

/system/filesystem/copy - copy file(s)

Parameters:

Source	Enter the path and file name of the source file that you want to copy.
Destination	Enter the path and file name of the destination directory. You can enter a different file name in the destination field if you want to rename it as well.
Overwrite	Clicking generates a check box. Activation causes destination file to be overwritten
Working - directory	Enter a working directory, e.g. C:\Temp. If this is empty, the temporary directory is used.

/system/filesystem/delete - delete file(s)

Path	The path and file name of the file you want to delete.
Working - directory	Enter a working directory, e.g. C:\Temp. If this is empty, the temporary directory is used.

/system/filesystem/mkdir - create directory

Parameters:

Path	Enter the path/location of the new directory
MakeParents	Clicking generates a check box. Activation allows a hierarchical path to be created in one step. E.g. working directory is c:\temp, and path is temp2\temp3. Creates the new directory c:\temp\temp2\temp3.
Working - directory	Enter a working directory, e.g. C:\Temp. If this is empty, the temporary directory is used.

/system/filesystem/move - move or rename file(s)

Parameters:

Source	Enter the path and file name of the source file that you want to move.
Destination	Enter the path and file name of the destination directory. If you only supply the directory name in this field, then the original file name will be retained.
Overwrite	Clicking generates a check box. Activation causes destination file to be overwritten.
Working - directory	Enter a working directory, e.g. C:\Temp. If this is empty, the temporary directory is used.

Note - Working directory:

This entry must be a path on the server machine (that runs FlowForce), not on your local machine.

/system/filesystem/rmdir - remove directory

Path	Enter the path/location of the directory/folder you want to delete
Working - directory	Enter a working directory, e.g. C:\Temp. If this is empty, the temporary directory is used.

4.5.2 ftp - FTP client

Allows you to use FTP commands on remote servers.

system/ftp/delete

Parameters:

FTP server	Address of the remote FTP server, either as a URL or IP address.
Port	The port number used to connect to the FTP server
Directory on host	The name of the directory, on the host, from which you want to delete a file.
Login username	User name needed to connect to the host.
Login password	Password needed to connect to the host.
Use passive mode	Use the passive mode FTP connection, if connection problems occur, e.g. routers or firewalls may be set up to avoid active connections.
Target file	The name of the file that you want delete from the server.
Account	The FTP account name of the user allowed access to the files on the remote server.

system/ftp/mkdir

Parameters:

FTP server	Address of the remote FTP server, either as a URL or IP address.
Port	The port number used to connect to the FTP server
Directory on host	The name of the directory, on the host, from within which you want to create a new directory.
Login username	User name needed to connect to the host.
Login password	Password needed to connect to the host.
Use passive mode	Use the passive mode FTP connection, if connection problems occur, e.g. routers or firewalls may be set up to avoid active connections.
Target directory	The name of the directory that you want create on the server.
Account	The FTP account name of the user allowed access to the files on the remote server.

system/ftp/move

Parameters:

FTP server	Address of the remote FTP server, either as a URL or IP address.
Port	The port number used to connect to the FTP server

Directory on host	The name of the directory, on the host, from where you want to move the file.
Login username	User name needed to connect to the host.
Login password	Password needed to connect to the host.
Use passive mode	Use the passive mode FTP connection, if connection problems occur, e.g. routers or firewalls may be set up to avoid active connections.
Source file	Name of the source file that you want to move to a different location.
Target file	Name of the copied file at the target location. Use a different name if you want to rename the copied file.
Account	The FTP account name of the user allowed access to the files on the remote server.

system/ftp/retrieve (file from server)

Parameters:

FTP server	Address of the remote FTP server, either as a URL or IP address.
Port	The port number used to connect to the FTP server
Directory on host	The name of the directory, on the host, from where you want to retrieve the file.
Login username	User name needed to connect to the host.
Login password	Password needed to connect to the host.
Use passive mode	Use the passive mode FTP connection, if connection problems occur, e.g. routers or firewalls may be set up to avoid active connections.
Source file	Name of the source file that you want to retrieve.
Target file	Name the file should have once it has been retrieved (change the name to rename it).
Overwrite target	Clicking generates a check box. Activation causes destination file to be overwritten.
Working directory	Directory you want to retrieve the file from. Note that this must be a path on the server machine (that runs FlowForce), not on your local machine.
Account	The FTP account name of the user allowed access to the files on the remote server.

system/ftp/rmdir

Parameters:

FTP server	Address of the remote FTP server, either as a URL or IP address.
Port	The port number used to connect to the FTP server
Directory on host	The name of the directory, on the host, from within which you want to remove a directory.
Login username	User name needed to connect to the host.
Login password	Password needed to connect to the host.
Use passive mode	Use the passive mode FTP connection, if connection problems occur, e.g. routers or firewalls may be set up to avoid active connections.
Target directory	The name of the directory that you want to remove on the server.
Account	The FTP account name of the user allowed access to the files on the remote server.

system/ftp/store

Parameters:

FTP server	Address of the remote FTP server, either as a URL or IP address.
Port	The port number used to connect to the FTP server
Directory on host	The name of the directory, on the host, where you want to store the file.
Login username	User name needed to connect to the host.
Login password	Password needed to connect to the host.
Use passive mode	Use the passive mode FTP connection, if connection problems occur, e.g. routers or firewalls may be set up to avoid active connections.
Source file	Name of the file that you want to store.
Target file	Name of the file to be stored at the target location. Use a different name if you want to rename the copied file.
Working directory	Directory where you want to store the transferred file. Note that this must be a path on the server machine (that runs FlowForce), not on your local machine.
Account	The FTP account name of the user allowed access to the files on the remote server.

4.5.3 mail - Sending E-mail

Sends a mail from FlowForce Server to the specified recipients, generally the administrator.

Note: The mail server settings are global and can be defined in Administration - [Settings](#).

system/mail/send - send a mail

Parameters:

From	Email address from which the mail is to be sent.
To	Recipients email address.
Subject	Subject line of the message.
Message body	Body text of the message.
Attachment	File name of the attachment sent with the email.

4.5.4 shell - Command line execution

Allows you to execute a shell command line.

/system/shell/commandline - execute any command line

Parameters:

Command	Enter any command line command to execute, e.g. batch files or other executables.
Working-directory	Enter a working directory, e.g. C:\Temp. If this is empty, the temporary directory is used. Note that this must be a path on the server machine (that runs FlowForce), not on your local machine.

4.5.5 compute - Evaluating expressions

This step function computes the result of an expression and returns the computed value.

system/compute - compute an expression

Parameters

Parameters expression

By assigning a step result name to the compute step, the computed value can be used in parameters or expressions of following execution steps.

You can also use this function as the last step to define the output of a job that is used as a service. In the screenshot shown below the result of the first step is assigned to "hello". The compute function then evaluates the expression and outputs its value to the caller.

E.g.

Using the execute function system/shell/commandline

Command echo "hello world"
Assign step's result to hello

Execute function system/compute
expression concat(stdout(hello))

The screenshot shows a configuration window titled "Execution Steps". It contains two steps, each with a plus icon in a circle to its left. The first step is "Execute function /system/shell/commandline". Its parameters are "Command: echo \"hello world\"" and "Working directory: c:\\temp". Below the parameters is a field "Assign this step's result to" with the value "hello". The second step is "Execute function /system/compute". Its parameters are "Expression: concat(stdout(hello))". Below the parameters is a field "Assign this step's result to" with the value "name". At the bottom of the window are four buttons: "new Execution step", "new Choose step", "new For-each step", and "new error/success handling step".

Index

8

8082,
TCP port - defining, 14

A

Accessing,
setup page from any machine, 14

Act,
on triggering file, 241

Action, 241
on trigger file, 189

Administration Interface,
FlowForce Server, 69
starting and using, 14

Administrator guide, 6

Administrator interface, 59

Alert emails, 65

Altova LicenseServer,
(see LicenseServer), 29

Altova ServiceController, 34

Always, 220

Architecture,
FlowForce and servers, 7

Assigning licenses, 54, 62

B

Batch file,
DoTransform.bat, 141

C

Call,
function, 215

Choice,

conditions, 218

Comman line,
options, 167

Command exportresourcestrings, 135

Command generate, 136

Command help, 135

Command line,
and XQuery, 162
exportresourcestrings, 119
help, 119
licenseserver, 117
MapForce Server commands, 116
run, 116
Set default lang, 118
usage summary, 144

Command line usage, 134

Command setdeflang, 138

Command setfopath, 139

Commandline, 116, 117, 118, 119

Commandline - MapForce Server, 116

Concepts, 9, 210

Condition,
test, 220

Conditions,
choice, 218

Configuration files,
FlowForce Server, 68

Configuration page, 59
opening on Linux, 39
opening on Mac OS X, 41
opening on Windows, 37
URL of, 37
URL of (Linux), 39
URL of (Mac OS X), 41

Configuration tool,
starting, 14

Container,
and objects, 209
navigating, 183
public/system, 209

Cpcommand licenseserver, 138

Credential,
adding to job, 183
FTP, 86, 238
local credential, 86, 238

Credentials, 238

D

Days of months, 230

Days of week, 230

Days of weeks of months, 231

Default,

containers, 209

Default language,

command line, 118

Default password, 37

Deploy,

MapForce mapping, 180

Deployed mapping,

function, 209

Deployed package,

changing input/output files, 241

Directory change,

polling for, 189

Domain,

user name entry, 86

DoTransform,

batch file, 141

E

Enabled,

trigger definition, 228

Execution,

for each, 220

exportresoucestrings,

command line, 119

exportresourcestrings command, 135

Expressions,

results, 222

step, 222

F

Features, 122

File system,

file system trigger, 189

File system trigger - trigger types, 228

FlowForce,

Administrator guide, 6

deploy MapForce mapping, 180

Introduction, 3

job and triggers - defining, 183

modules, 7

starting, 14

Tutorial, 178

FlowForce Administration Interface, 7

starting and using, 14

FlowForce Architecture, 7

FlowForce Server, 7

configuration files, 68

data storage, 68

registering with LicenseServer, 46

FO processing, 139

For each,

counter, 220

FTP,

credential, 86, 238

Function, 241

call, 215

deployed mapping / OS function, 209

Functionality, 122

Functions,

adding system functions to a step, 186

G

generate command, 136

Getting starting, 11

H

Help,

command line, 119

help command, 135

Help command on CLI, 164

Home, 69

HTTP trigger, 228, 232

I**Input parameters,**

- job input parameters, 214

Input/Output,

- changing from deployed package, 241

Installation,

- paths - Windows XP, Vista, Windows 7, 13

Installation on Linux, 110, 125**Installation on Mac OS X, 113, 127****Installation on Windows, 124****Interface,**

- FlowForce Administration, 69

Interval,

- polling interval, 189

Interval counted from,

- trigger definition, 228

Introduction,

- to FlowForce, 3

J**Job,**

- credential for, 183
- input parameters, 214
- steps, 241
- steps - defining, 183
- Store result in, 215

Job log,

- viewing, 183

Job step,

- adding additional to job, 186

Jobs, 69

- and triggers - defining, 183

L**License commands on CLI, 165****License Pool, 43, 59****Licenses,**

- assigning, 54, 62
- uploading, 43, 59

Licenseserver,

- command line - registering MapForce, 117
- Configuration page, 59
- installation on Mac OS X, 33
- installation on Windows, 31, 32
- interface with, 59
- registering FlowForce Server with, 46
- registering MapForce Server with, 50
- registering StyleVision Server with, 53
- settings, 65
- starting, 35
- steps for assigning licenses, 35

licenseserver command, 138**LicenseServer configuration page,**

- (see Configuration page), 37, 39, 41

Licensing, 138**Licensing on Linux, 110, 111, 125, 130****Licensing on Mac OS X, 113, 127, 132****Licensing on Windows, 108, 124, 129****Linux,**

- installation on, 110, 125
- licensing on, 110, 111, 125, 130

Local credential, 86, 238**Localization, 165**

- of StyleVision Server, 135, 138

Log, 69

- job log - viewing, 183

Log file,

- generating - redirecting, 116

Login, 14**Logout, 67****M****Mac OS X,**

- installation on, 113, 127
- licensing on, 113, 127, 132

MapForce, 7**MapForce Server, 7**

- command line commands, 116
- registering with LicenseServer, 50

Mapping,

- Deploy to FlowForce, 180

Messages, 67**Modules,**

- FlowForce installation, 7, 13

N

Network information, 30

Network settings, 65

O

Object system,

jobs, packages, triggers, 208

Objects,

and containers, 209

On error, 220

Onsuccess, 220

Output,

of StyleVision Server, 136

Output/Input,

changing from deployed package, 241

P

Package,

system/sys, 241

Packages, 69

Parameter,

adding, 192

Parameters,

job input parameters, 214

Password,

default at startup, 37

PDF output, 139

Permission,

adding, 83

Permissions,

restricting, 81

Polling,

directories, 189

Port, 14

Privileges, 79

Properties,

common to triggers, 228

Protected,

step, 220

Public,

container, 209

Q

Query,

database, 192

Queue settings, 237

R

RaptorXML Server, 141

Register MapForce Server, 117

Registering FlowForce Server with LicenseServer, 46

Registering MapForce Server with LicenseServer, 50

Registering StyleVision Server with LicenseServer, 53

Repeat,

trigger definition, 228

Restricting,

permissions, 81

Restricting user rights, 77

Rights,

restricting, 77

Role, 75

Roles, 69

Roles and Users, 72

Root,

container, 208

Run,

command line, 116

generate log file, 116

Run daily, 230

Run on,

trigger definition, 228

Run once, 229

S

Security, 71

Server Management tab, 54, 62

Server Monitoring tab, 65

Service, 234

ServiceController, 34

- setdeflang command, 138**
- setfopath command, 139**
- Settings, 65, 69**
- Setup page,**
 - URL of, 14
- Source,**
 - files - changing from deployed package, 241
- Source file,**
 - use trigger file, 189
- Starting,**
 - FlowForce, 14
- Step,**
 - adding to an existing job, 186
- Step protected, 220**
- Steps,**
 - defining job steps, 183
- store result in,**
 - variable, 215
- StyleVision Server,**
 - registering with LicenseServer, 53
- System,**
 - container, 209
 - functions - adding (copy/move etc.), 186
- System action, 241**
- System/ sys package, 241**

T

- Target,**
 - files - changing from deployed package, 241
- TCP port - defining, 14**
- Technical details, 235**
- Timer trigger, 228**
- Timer trigger periods, 228**
- Trigger file,**
 - use as source file, 189
- Triggering file,**
 - act on, 241
- Triggers,**
 - and jobs - defining, 183
 - common properties, 228
- Tutorial,**
 - FlowForce, 178

U

- Uploading licenses, 43, 59**
- URL of Setup page, 14**
- User,**
 - adding, 72
- User guide, 208**
- User name,**
 - domain use, 86
- Users, 69**
- Users and Roles, 72**

V

- Validation,**
 - of any document, 149, 157
 - of DTD, 147
 - of XBRL instance, 154
 - of XBRL instance and taxonomy, 153
 - of XBRL taxonomy, 156
 - of XML instance with DTD, 145
 - of XML instance with XSD, 146
 - of XQuery document, 163
 - of XSD, 148
 - of XSLT document, 160
- View,**
 - job log view, 183

W

- Web service, 195**
- Well-formedness check, 150**
- Windows,**
 - installation on, 124
 - licensing on, 108, 124, 129

X

- XBRL validation,**
 - see Validation, 153

XQuery commands, 162
XQuery document validation, 163
XQuery execution, 162
XSLT commands, 158
XSLT document validation, 160
XSLT transformation, 159