

Altova FlowForce Server 2024



User & Reference Manual

Altova FlowForce Server 2024

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: 2024

© 2018-2024 Altova GmbH

Table of Contents

1	Introduction	8
1.1	New Features.....	9
1.1.1	Version 2024.....	9
1.1.2	Version 2023.....	9
1.1.3	Version 2022.....	10
1.1.4	Version 2021.....	10
1.1.5	Version 2020.....	11
1.2	Overview.....	12
1.3	Terminology.....	15
1.4	Important Paths.....	18
2	Installation and Configuration	20
2.1	Installation and Licensing.....	21
2.1.1	Setup on Windows.....	21
2.1.2	Setup on Linux.....	30
2.1.3	Setup on macOS.....	37
2.1.4	Upgrade FlowForce Server.....	44
2.2	Configuration via Setup Page.....	45
2.2.1	Create New Server Instance.....	46
2.2.2	Configure Instance Parameters.....	47
2.2.3	Set Up SSL Encryption.....	50
2.2.4	Install and Start Services.....	60
2.3	Configuration via Configuration Files and CLI.....	64
2.3.1	Overview of Configuration Files.....	64
2.3.2	Instance Parameters in Configuration Files.....	67
2.4	Administration Tasks.....	72
2.4.1	Define Users and Roles.....	72
2.4.2	Back Up, Restore, and Migrate Data.....	75
2.4.3	Localize FlowForce Server.....	79

3	Web UI Reference	81
3.1	Home.....	85
3.1.1	Job Info on Home Page.....	85
3.1.2	Job Statuses.....	89
3.2	Configuration.....	92
3.2.1	Permissions and Containers.....	92
3.3	Log	104
3.3.1	Job Info in Log.....	104
3.3.2	Instance Log.....	106
3.4	Administration.....	109
3.4.1	Users	109
3.4.2	Roles	112
3.4.3	Reports.....	114
3.4.4	Password Policies.....	116
3.4.5	Settings.....	118
4	Job Configuration	128
4.1	Input Parameters.....	130
4.2	Job Execution Steps.....	132
4.2.1	Execution Steps.....	132
4.2.2	Choose Steps.....	133
4.2.3	For-Each Steps.....	135
4.2.4	Error/Success-Handling Steps.....	136
4.2.5	Postpone Steps.....	140
4.3	Execution Result.....	145
4.4	Cache Result.....	148
4.5	Triggers.....	151
4.5.1	Timer Triggers.....	152
4.5.2	File System Triggers.....	154
4.5.3	HTTP Triggers.....	155
4.6	Jobs as Web Services.....	158
4.7	Credentials.....	162

4.7.1	Credential Type: Password.....	163
4.7.2	Refer to Credentials from Jobs.....	164
4.8	Queue Settings.....	166
4.9	Expressions.....	171
4.9.1	Compute an Expression.....	171
4.9.2	Expression Language Rules.....	172
4.9.3	Embed Expressions in String Fields.....	175
4.9.4	Call Expression Functions.....	176
4.9.5	FlowForce Data Types.....	177
4.9.6	Operators.....	178
4.10	Expression Functions.....	180
4.10.1	General Utility Functions.....	180
4.10.2	Boolean Functions.....	185
4.10.3	MIME/Stream Functions.....	188
4.10.4	Result Functions.....	202
4.10.5	List Functions.....	209
4.10.6	File System Functions.....	214
4.10.7	String Functions.....	219
4.10.8	Execution State Functions.....	229
4.10.9	Runtime Information Functions.....	232
4.11	System Functions.....	235
4.11.1	/system.....	235
4.11.2	/system/filesystem.....	241
4.11.3	/system/ftp.....	245
4.11.4	/system/mail.....	278
4.11.5	/system/maintenance.....	284
4.11.6	/system/shell.....	285
4.12	Import/Export Configuration Data.....	288
4.12.1	Export Configuration Data.....	288
4.12.2	Import Configuration Data.....	292
5	Job Examples	294
5.1	Create a "Hello, World!" Job.....	296
5.2	Check if a Path Exists.....	299

5.3	Copy Files.....	303
5.4	Create a Job from a MapForce Mapping.....	308
5.5	Use a Job as Step of Another Job.....	316
5.6	Create a Directory Polling Job.....	319
5.7	Add Error Handling to a Job.....	325
5.8	Expose a Job as a Web Service.....	331
5.9	Post JSON to FlowForce Web Service.....	339
5.10	Cache Job Results.....	347
5.11	Create a Job from a StyleVision Transformation.....	351
5.12	Validate a Document with RaptorXML.....	359
5.13	Validate XML with Error Logging.....	361
5.14	Run XSLT with RaptorXML.....	366
5.15	Generate PDFs from XML Files.....	371

6 Command Line 383

6.1	help	387
6.2	assignlicense.....	388
6.3	compactdb.....	389
6.4	createdb.....	390
6.5	debug.....	391
6.6	exportresourcestrings.....	392
6.7	foreground.....	393
6.8	initdb.....	394
6.9	install.....	395
6.10	licenseserver.....	396
6.11	migratedb.....	397
6.12	repair.....	398
6.13	resetpassword.....	399
6.14	setdeflang (sdl).....	400
6.15	start	401
6.16	uninstall.....	402
6.17	upgradedb.....	403
6.18	verifylicense.....	404

7	Integration with Altova Products	405
7.1	Prepare Files for Server Execution.....	406
7.2	Deploy Mappings to FlowForce Server.....	412
7.3	Run Mappings and Transformations as Jobs.....	417
7.3.1	Credentials in Mapping Functions.....	421
7.3.2	Dynamic Authentication.....	422
7.3.3	Resources.....	423
7.4	Access the Mapping/Transformation Result.....	431
7.5	Integration with RaptorXML Server.....	433
7.6	Tool Files.....	436
	Index	438

1 Introduction

FlowForce Server is a cross-platform software solution that enables you to automate XML/XBRL-processing tasks, data mappings, data transformations, common server tasks, and many other tasks. FlowForce Server has a Web interface that allows you to create, administer, and modify jobs. The Web interface also enables you to create and manage users and roles, define privileges and permissions, and configure various other settings. FlowForce Server continuously checks for trigger conditions, starts and monitors job execution, and logs all system- and job-related activities. FlowForce Server is available on Windows, Linux, and macOS.



You can integrate FlowForce Server with other Altova products, which will enable you to automate various data-processing tasks:

- For example, you can deploy [MapForce](#) mappings and [StyleVision](#) transformations directly to FlowForce Server, which allows converting your data transformations to flexibly configurable jobs.
- To run jobs created from deployed MapForce mappings and StyleVision transformations, FlowForce Server calls [MapForce Server](#) and [StyleVision Server](#), respectively.
- You can also configure [RaptorXML Server and RaptorXML+XBRL Server](#) tasks as FlowForce Server job steps. This will enable you to validate XML and JSON files, XBRL instance documents and XBRL taxonomies, and perform XSLT and XQuery operations.

With FlowForce Server, you can also create and automate common server tasks, such as sending emails, managing files on the local system or network, running shell scripts, and others. FlowForce Server Advanced Edition can send and accept AS2 messages and supports the distributed execution of jobs on multiple servers running as a cluster.

Last updated: 8 April 2024

1.1 New Features

This section describes new features of each FlowForce Server release. For more details, see the respective subsection.

1.1.1 Version 2024

Version 2024 Release 2

- The Setup page has been enhanced to allow users to customize the data directory before creating instance data and installing services. This makes the setup procedures more flexible and user-friendly. For details, see [Configuration via Setup Page](#)⁴⁵.
- When you deploy your MapForce mapping to FlowForce Server, you can choose to attach the mapping files for later retrieval. This will prevent you from losing your mapping files and enable you to download them at any time. For details, see [Deploy Mappings to FlowForce Server](#)⁴¹².
- The Error/Success-Handling step now enables you to set a timeout that will abort a job step if it is not successful after the specified time. For details, see [Error/Success-Handling Steps](#)¹³⁸.
- It is now possible to add a Resume step inside a protected block, which allows continuing execution even after an error has occurred. For more details, see [Error/Success-Handling Steps](#)¹³⁸.
- New expression functions have been introduced to create the result that an error-handling block should return. For details, see [Result Functions](#)¹³⁸.
- Internal updates and optimizations.

Version 2024

- FlowForce Server now allows you to send a test email to [test SMTP parameters](#)¹¹⁸.
- The *Client Credentials*, *Password*, and *Implicit* grant types are now supported in OAuth credentials, in addition to the *Authorization Code* grant type (*Advanced Edition*).
- Internal updates and optimizations.

1.1.2 Version 2023

Version 2023 Release 2

- It is now possible to view exported log entries. For more information, see [Job Info in Log](#)¹⁰⁶ and [Instance Log](#)¹⁰⁷.
- Internal updates and optimizations.

Version 2023

- In the [Execution Steps](#)¹³² section (**Configuration** page), it is now possible to expand or collapse all execution steps, which might be useful when you want to do a search in the browser or print the page.
- It is now also possible to expand/collapse information in the [instance log](#)¹⁰⁸.

- The system function `/system/sftp/retrieve-wildcard` has a new parameter called *Target directory* that specifies where retrieved files will be stored (*FlowForce Server Advanced Edition*).
- FlowForce Server now allows you to copy over previously set values of function parameters into parameters of a new step function. For details, see [Input Parameters](#) ¹³⁰.
- FlowForce Serve now displays more detailed information about [file triggers](#) ¹⁵⁴ on the **Active Triggers and Services** page, which gives you more fine-grained control over jobs triggered by file triggers. For more information, see [Monitor Job Execution](#) ⁸⁸.
- Internal updates and optimizations.

1.1.3 Version 2022

Version 2022 Release 2

- A new severity filter called *Verbose* is now available on the [Log View](#) ¹⁰⁴ page. The *Verbose* messages can be useful for troubleshooting [file system triggers](#) ¹⁵⁴.

Version 2022

- The [Running Jobs](#) ⁸⁵ section of the **Home** page now displays the following job execution details: *all jobs*, *recently finished*, *starting*, and *running jobs*.
- A new system function called [create-file](#) ²⁴⁰ has been introduced. This function allows you to store stream content in a file you would like to keep for future use.

1.1.4 Version 2021

Version 2021 Release 2

- The existing FlowForce built-in functions from the [/system/ftp](#) ²⁴⁵ library now support options for connecting to a server via FTPS (FTP via SSL).
- New FTP functions are available that enable uploading, retrieving, and deleting files on a remote FTP server using wildcards. Specifically, if you connect to the FTP server through FTP or FTPS, you can use the functions `delete-wildcard`, `retrieve-wildcard`, and `store-wildcard` from the [/system/ftp](#) ²⁴⁵ library.
- When creating a [file system trigger](#) ¹⁵⁴, you can set the minimum polling interval to 1 second (previously, the minimum interval was 30 seconds).
- Statistics Detail Page: changes in the color scheme and labeling.
- File Path Functions: [join-paths](#) ²¹⁵ is a new function that allows combining paths supplied as arguments into one path.

Version 2021

- The [Log View](#) ¹⁰⁴ page has been optimized to load records faster and includes new navigation and filtering options, as well as the ability to save the current state of the log as a permanent link.
- A new [Log Instance](#) ¹⁰⁶ page is available that is dedicated exclusively to viewing one logged job instance at a time. From this page, you can export the logged information to a .zip archive in order to

view it later or send it to another party. You can also load previously exported job instances into the "Log Instance" page and view them for *post mortem* debugging, for example.

- There are new [Logging Settings](#)¹²² available that let you configure whether certain logging details should be stored or skipped for logging purposes. You can also configure the level of logging detail based on the job outcome. For example, on job failure, you might want to keep full tracing information in the log, whereas on successful execution you might want to keep only the most basic information.
- You can configure certain logging settings not only at application level, but also for specific FlowForce Server jobs. See [Logging rules at object level](#)¹²⁵.

1.1.5 Version 2020

Version 2020 Release 2

- It is possible to retry the execution of one or more steps multiple times in case of error, see [Retry on Error](#)¹³⁷.
- A job can execute steps in a postponed way, after returning the result, which is particularly suitable in case of jobs invoked through Web service calls, see [Postponed Steps](#)¹⁴⁰.
- A new optional **Host name** field is available in the setup page, see [Defining the Network Settings](#)⁴⁷. This makes SSL configuration more flexible, and also enables you to test run Web services directly from the job configuration page.
- It is possible to configure [file system triggers](#)¹⁵⁴ to fire when new files or directories are added to a specified directory. This trigger is different from the existing "Modified date" in that it does not fire if files within the polled directory are subsequently modified.
- The procedure for accessing the [Setup Page](#)⁴⁷ has been simplified.

Version 2020

- You can define credentials both in MapForce and FlowForce Server, and either embed them into the mapping at design time, or supply them as parameters to the execution step in FlowForce Server, see [Credentials in Mapping Functions](#)⁴²¹.
- When [defining a credential object](#)¹⁶³, you can restrict it to a specific domain of usage. "Usage" can be one or more of the following: job execution, FTP, HTTP.
- Portable file, folder, and database references defined in MapForce (also known as "Global Resources") can be deployed to FlowForce Server and be consumed by a mapping function. If necessary, you can change directly in FlowForce the resources (file, folder, or database references) used by a mapping function—this will affect all FlowForce jobs using that function. You can also create or edit resources directly in FlowForce Server, with some limitations, see [Resources](#)⁴²³.
- When exporting job configuration data to another FlowForce Server instance or to a .zip archive, you can optionally choose to export sensitive data as well, see [Importing and Exporting Configuration Data](#)²⁸⁸.

1.2 Overview

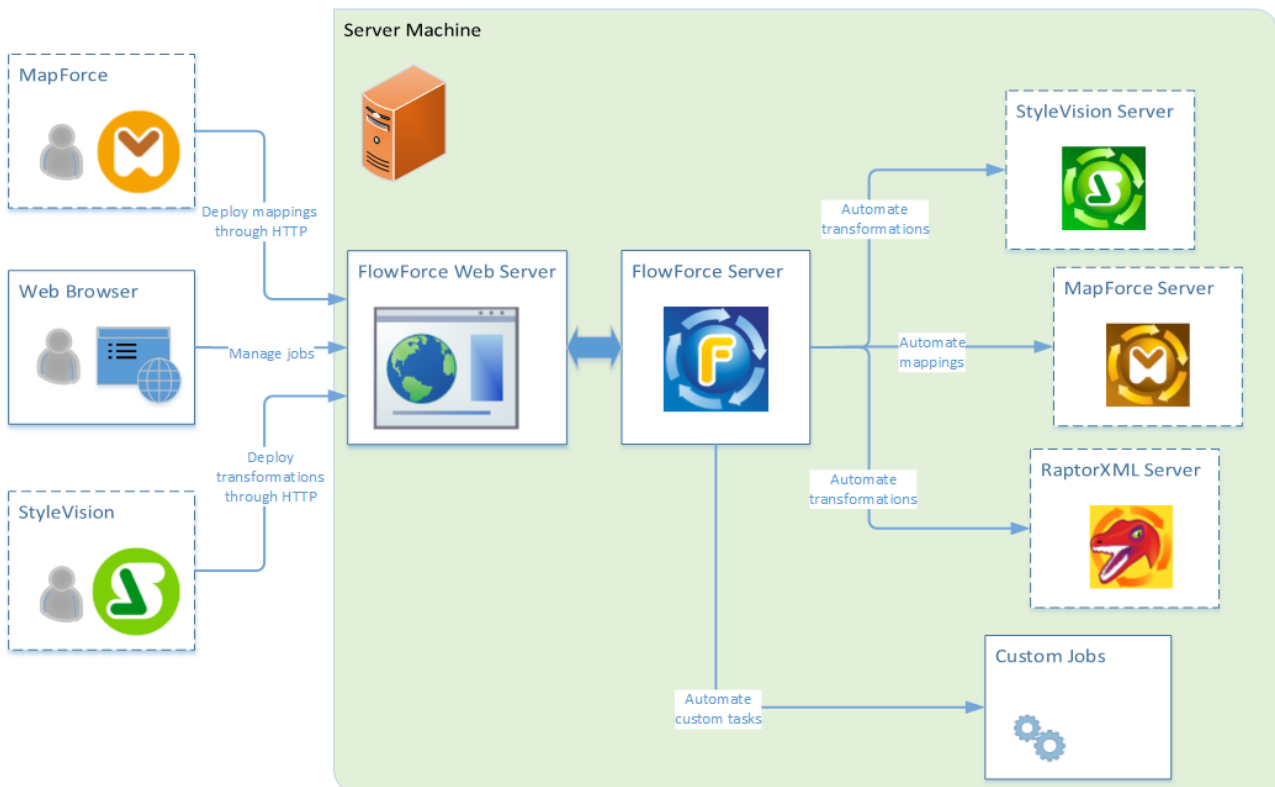
Altova website: [Workflow Automation Tool](#)

This topic describes the FlowForce Server architecture. The architecture is illustrated in the diagram below. The components that are optionally licensed are represented with dashed borders. The FlowForce Server solution consists of two services: *FlowForce Web Server* and *FlowForce Server*. They run as separate services and can be configured, started, and stopped separately. The manner in which these two services are managed depends on the operating system family (Linux, Windows, or macOS).

FlowForce Web Server accepts and validates requests from clients (*see Clients below*) and passes these requests to FlowForce Server. FlowForce Server is the core of the FlowForce Server solution and runs as a background service without a graphical user interface. FlowForce Server continuously checks for trigger conditions, starts and monitors job execution, and logs all system- and job-related events. In addition to this, FlowForce Server listens to requests for jobs that are exposed as [Web services](#)¹⁵⁸.

FlowForce Web Server handles requests through the Web administration interface where you can create, modify, and monitor jobs, create users and roles, define privileges and permissions, and configure various FlowForce settings. FlowForce Web Server accepts HTTP (or HTTPS) connections from the clients described below. For information about terminology associated with job configuration and execution, see [Terminology](#)¹⁵.

FlowForce Server can be integrated with other Altova products. For details, see *Clients* and *Integration with Altova Server Products* below.



Clients

The clients from which FlowForce Web Server can accept requests and send them further to FlowForce Server are described below.

Web browser

A Web browser is used to configure FlowForce Server jobs and other settings. For an overview of the Web administration interface, see [Web UI Reference](#) ⁸¹.

MapForce Enterprise and Professional editions

[MapForce](#) is a data mapping desktop application in which you visually design mappings that transform your data or convert it from one format to another. Once you have created your mapping in MapForce, you can deploy it to FlowForce Server. This will enable you to convert your mapping to a flexibly configurable job. For example, you can configure a mapping job that will run at a specific time or whenever a file is added to the monitored directory.

To run jobs created from MapForce mappings, FlowForce Server calls [MapForce Server](#), whose role is to actually execute mappings and generate output files. For details about deploying MapForce mappings to FlowForce Server, see [Integration with Altova Products](#) ⁴⁰⁵.

StyleVision Enterprise and Professional editions

[StyleVision](#) is a desktop application that enables you to visually design reports and forms for XML, database, and XBRL data. StyleVision enables you to create StyleVision Power Stylesheets (i.e., SPS) that control the display and entry of data of databases, XML and XBRL documents and specify the output design of an XML document transformation. Once you have designed your SPS, you can save it as a Portable XML Form (PXF) file that packages the SPS file with its related files and deploy this PXF file to FlowForce Server. The deployed file then becomes available for use in any transformation job on the server.

To execute jobs created from deployed StyleVision transformations, FlowForce Server calls [StyleVision Server](#), whose role is to actually execute transformations and produce output files.

For more information about each product, refer to the Altova documentation page (<https://www.altova.com/documentation.html>).

Integration with Altova server products

You can integrate FlowForce Server with other Altova server products. This enables you to automate these servers' tasks with the help of FlowForce Server. On Windows, the FlowForce Server installer comprises several Altova server products that you can install in addition to FlowForce Server. On other platforms, the Altova server products need to be installed separately. For information about each server product, see the subsections below.

StyleVision Server

[StyleVision Server](#) is based on the built-in report and document generation engine developed for StyleVision. StyleVision Server uses PXF files, which contain StyleVision stylesheets with related files, to render XML, XBRL, and database data into HTML, RTF, PDF, text, and Microsoft Word files. You can automate business report and document generation by deploying PXF files to FlowForce Server. This will enable you to run data transformations as scheduled jobs.

MapForce Server

[MapForce Server](#) is based on the built-in data transformation engine developed for MapForce. MapForce Server performs data transformations, using preprocessed data mappings stored in MapForce Server Execution (MFX) files. When MapForce Server operates under the management of FlowForce Server, data mappings are executed as FlowForce Server job steps.

RaptorXML Server

[RaptorXML Server](#) is Altova's third-generation, super-fast XML and XBRL processor, optimized for the latest standards and parallel computing environments. RaptorXML is available in two editions: RaptorXML Server and RaptorXML+XBRL Server. XBRL processing is available only in RaptorXML+XBRL Server.

When RaptorXML Server is installed on the same server as FlowForce Server, RaptorXML Server's functions become available as built-in FlowForce Server functions. This means that you can create jobs that validate and check the well-formedness of XML and JSON documents, XBRL taxonomies and instance files. You can also perform XSLT transformations and execute XQuery documents. For more information, see [Integration with RaptorXML Server](#)⁴³³.

For more information about each product, refer to the Altova documentation page (<https://www.altova.com/documentation.html>).

1.3 Terminology

This topic provides information about basic concepts associated with job execution and access-control management.

Job A job is a task or a sequence of tasks that will be executed by the server. A job consists of the following parts (some of them are optional): input parameters, execution steps, triggers, credentials and various settings.

The degree of complexity of a job can vary, depending on your business needs and requirements. A job can consist of a single step (e.g., sending an email) or can be configured to perform multiple actions and to pass the result (e.g., a file) as a parameter to another job. For details about job configuration, see [Job Configuration](#)¹²⁸.

Job instance A job instance is not the same as a job. When you configure a FlowForce job on the job configuration page, you create in fact a job configuration. Every time the defined trigger criteria for a job apply, an instance of the job starts running. Every job instance has an execution result that can be successful, failed, or interrupted/unknown. For details, see [Statistics](#)⁸⁵.

Trigger When you create a job, you must specify conditions that will start the job. These conditions are known as [triggers](#)¹⁵¹. FlowForce Server continuously checks for trigger conditions and executes the job whenever a specific trigger condition is met. A job can have multiple triggers.

Step In FlowForce Server, steps define what a job must do (e.g., delete a file, execute a MapForce mapping, send an email). In its simplest form, a step is an operation with failed or successful outcome. Each step must execute [a function](#)¹⁵. You can create as many steps as required for your job and set the order in which the steps must be executed. You can also use [the result of a step](#)¹⁴⁵ in other steps.

To find out more about steps, see [Job Execution Steps](#)¹³².

Function In FlowForce Server, there are two types of functions: (i) step functions and (ii) expression functions.

A step function defines a particular operation to be performed. Each execution step must have a step function. For example, the [system/mail/send](#)²⁷⁹ function instructs FlowForce Server to send an email to the specified recipients. The following types of step functions are available:

- [System functions](#)²³⁵
- [StyleVision transformations](#)⁴²⁰
- [MapForce mappings](#)⁴¹⁸
- [A job as an execution step of another job](#)³¹⁶

Most step functions have parameters. Parameters can accept different values, including [expressions](#)¹⁷¹ and [expression functions](#)¹⁸⁰. Expression functions manipulate values supplied as arguments, for example, to join strings (see the [concat](#)²²² function). For an example of a job that uses expression functions, see Example 2 in the [send-mime](#)²⁸⁰ function.

Execution result In FlowForce Server, you can work with execution results at two levels: (i) at step level and (ii) at job level. The result of a step defines what is returned after the step has been executed (e.g., a file). You can use the step result in other execution steps. See Example 1 in the [send-mime](#)²⁸¹ function, in which the result of the second step is used in the *Message body* parameter of the last step. At job level, you must specify the return type of the execution result if you want to [cache the job result](#)¹⁴⁸. Declaring the return type of the job result might also be meaningful if you intend to use this result in other jobs.

For more information, see [Step/Job Result](#)¹⁴⁵.

Credential A credential object is a piece of data that stores authentication information such as usernames and passwords, certificates, API keys, tokens, etc. that are used to securely manage and transmit authentication details and access different services and resources.

For more information about credentials, see [Credentials](#)¹⁶².

Container FlowForce Server manages jobs, credentials, step functions, and other configuration objects in a hierarchical structure of containers. A container is similar to a folder on an operating system. Containers can have any of the following: jobs, credentials, functions, and other containers. By setting permissions on a container, you can control who can access the container's contents.

User A user is a person who logs on to FlowForce Server to create and monitor jobs, deploy MapForce mappings and StyleVision transformations, and configure various settings. The scope of actions available to users in FlowForce Server depends on the following:

- The permissions and privileges assigned to the users
- The permissions and privileges assigned to the roles that the users are members of

Role A role defines a set of privileges and permissions. It can be assigned to another role or to a user. A role's privileges automatically become the privileges of any other role or any user that the role is assigned to. A user can be assigned any number of roles. As a result, a user will have all the privileges defined in the multiple assigned roles.

Note that privileges are global, whereas permissions are defined per container.

Privilege A privilege is an activity that a user is allowed to carry out (e.g., set a password, read users and roles, stop any job, etc.). A user can be assigned zero to all of the available privileges. It is recommended to assign privileges via roles rather than to assign privileges directly to the user. The assigning of privileges and roles to a user is done by a user that has been assigned this privilege. Initially, it is the `root` user that has this privilege.

Permission Permissions are access rights and can be set for each container individually. Permissions determine which users or roles have access to that container and what kind of access each user/role has (read, write, use, no access). Permissions can be defined for containers, configuration objects, credentials, queues, services, functions, resources, and child containers. In FlowForce Server Advanced Edition, permissions can also be set for certificates and AS2 partner objects.

Password policy A password policy defines a set of minimum requirements that a user password must meet in order to be valid (e.g., a password must be at least N characters long). FlowForce Servers uses password policies to enable administrators to enforce the complexity of user passwords.

1.4 Important Paths

After installing FlowForce Server, note the following directories where important files are stored:

- Installation directory (**INSTALLDIR**)
- Application-data directory (**APPDATADIR**)
- Instance-data directory (**INSTANCEDIR**)

The tables below provide information about the default locations of these directories on different operating systems.

INSTALLDIR

FlowForce Server installation directory (INSTALLDIR)	
Linux	/opt/Altova/FlowForceServer2024/
macOS	/usr/local/Altova/FlowForceServer2024/
Windows	C:\Program Files\Altova\FlowForceServer2024\ C:\Program Files (x86)\Altova\FlowForceServer2024\

APPDATADIR

The application-data directory (*table below*) contains two configuration files (**flowforceserver.ini** and **flowforceweb.ini**) that enable you to configure global configuration settings (currently, the language used in server logs and in error messages).

FlowForce Server application-data directory (APPDATADIR)	
Linux	/var/opt/Altova/FlowForceServer2024
macOS	/var/Altova/FlowForceServer2024
Windows	C:\ProgramData\Altova\FlowForceServer2024

INSTANCEDIR

The instance-data directories shown below are default paths. You can also select your custom location of the instance-data directory via [the FlowForce Server Setup page](#) ⁴⁵.

FlowForce Server instance-data directory (INSTANCEDIR)	
Linux	/var/opt/Altova/FlowForceServer/data
macOS	/var/Altova/FlowForceServer/data
Windows	C:\ProgramData\Altova\FlowForceServer\data

Important information about INSTANCEDIR

With 2024 R2, the default path to the instance-data directory has changed and contains no year anymore. This is the new recommended naming convention, because in-place upgrades are safer and faster. 2024 R2 has

also introduced a new way of installing server instances: via the Setup page as opposed to the installation wizard. The new method provides more control over the location of your instance data. For more information, see [Configuration via Setup Page](#)⁴⁵.

2 Installation and Configuration

To be able to work with FlowForce Server, you need first to install it, configure it, and carry out various administration tasks. For information about these procedures, see the subsections below.

Part 1: Installation and licensing

The installation of FlowForce Server consists of the following procedures:

1. Installing FlowForce Server
2. Installing LicenseServer
3. Starting LicenseServer
4. Registering FlowForce Server with LicenseServer
5. Assigning a license to FlowForce Server

For more information about these procedures, see [Installation and Licensing](#)²¹. The next step is to configure the server on the FlowForce Server Setup page (*see below*).

Part 2: Configuration on Setup page

The FlowForce Server Setup page provides a centralized way of configuring and managing your server instances. The configuration of FlowForce Server on the Setup page involves the procedures described below. Some of the procedures are optional (e.g., setting up SSL encryption).

1. [Creating a new server instance](#)⁴⁶
2. [Configuring instance parameters](#)⁴⁷, which includes:
 - a. Setting ports to connect to FlowForce Server and FlowForce Web Server
 - b. Setting up SSL encryption
 - c. Configuring the default time zone
 - d. Configuring cluster-related settings (*Advanced Edition*)
3. [Installing the services](#)⁶⁰
4. [Starting the services](#)⁶¹

Alternatively, you can configure your server instance via [the configuration files and CLI](#)⁶⁴.

After you have finished installing and configuring FlowForce Server, you can [log in](#)⁸¹ and carry out various administration tasks (*see below*).

Part 3: Administration tasks

Administration tasks involve the procedures described below. These procedures are on-demand tasks that you can carry out when necessary and in any order.

- [Creating and configuring users and roles](#)⁷²
- [Configuring basic settings](#)¹¹⁸ (the default time zone, parameters for the [/system/mail/send](#)²⁷⁹ function, directory service and logging settings)
- [Backing up, restoring, migrating data](#)⁷⁵
- [Revisiting the Setup page](#)⁷² (in case you need to change ports, enable SSL, etc.)
- [Localizing FlowForce Server](#)⁷⁹

Next step

After you have completed the configuration procedures described above, you can proceed with [job configuration](#)¹²⁸.

2.1 Installation and Licensing

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

- [Setup on Windows](#) ²¹
- [Setup on Linux](#) ³⁰
- [Setup on macOS](#) ³⁷
- [Upgrade FlowForce Server](#) ⁴⁴

2.1.1 Setup on Windows

This section describes the installation and licensing of FlowForce Server on Windows systems. The setup of FlowForce Server comprises the following procedures:

1. [Installing FlowForce Server](#) ²²
2. [Installing LicenseServer](#) ²⁵
3. [Starting LicenseServer](#) ²⁷
4. [Registering FlowForce Server with LicenseServer](#) ²⁸
5. [Assigning a license to FlowForce Server](#) ²⁸
6. [Configuring a server instance via the FlowForce Server Setup page or configuration files](#) ²²

In order for FlowForce Server to work, it must be registered and licensed with an [Altova LicenseServer](#) on your local machine or on another machine on your network. If LicenseServer is already installed and running on your network, you can skip Steps 2 and 3 and proceed with licensing and instance configuration.

Note that you can postpone registration and assigning a license and carry out these tasks within the framework of or after the configuration of FlowForce Server. However, it is mandatory to register FlowForce Server first, and only then will you be able to assign a license to FlowForce Server. Note that assigning a license to FlowForce Server does not require that FlowForce Server be running; only LicenseServer must be running.

Also note that if you install FlowForce Server together with LicenseServer, LicenseServer will start automatically after the installation process has successfully finished.

System requirements (Windows)

Note the following system requirements:

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

Prerequisites

Note the following prerequisites:

- Perform installation as a user with administrative privileges.
- From version 2021 onwards, a 32-bit version of FlowForce Server cannot be installed over a 64-bit version, or a 64-bit version over a 32-bit version. You must either (i) remove the older version before

installing the newer version or (ii) upgrade to a newer version that is the same bit version as your older installation.

2.1.1.1 Install on Windows

FlowForce Server is available for installation on Windows systems. The broad installation and setup procedure is described below.

Installing FlowForce Server

To install FlowForce Server, download the installation package from the Altova Download Center (<http://www.altova.com/download.html>), run it and follow the on-screen instructions. You can select your installation language from the box in the lower left area of the wizard. Note that this selection also sets the default language of FlowForce Server. You can change the language later from the command line.

Installing LicenseServer

In order for FlowForce Server to work, it must be registered and licensed with an [Altova LicenseServer](#) on your local machine or on another machine on your network. When you install FlowForce Server on Windows systems, you can install LicenseServer together with FlowForce Server. For details, see [Install LicenseServer](#)²⁵. If LicenseServer is already installed on your network, you might be prompted to update LicenseServer to the latest version (if applicable).

The installation wizard will also suggest registering FlowForce Server with Altova LicenseServer during the installation process. Alternatively, you can do this at a later stage (from [the Setup page](#)⁴⁷ or from [the command line](#)²⁸). For information about licensing FlowForce Server, see [Assign License to FlowForce Server](#)²⁶.

Installing additional Altova server products

The FlowForce Server installer also includes installers for the products listed below.

- *Altova MapForce Server*
- *Altova StyleVision Server*
- *Altova RaptorXML Server* (the installer for Altova RaptorXML+XBRL Server is available separately)

You can always install any Altova server product separately at a later time. Standalone installers are available in the [Altova Download Center](#). After having installed all the server products you wish to integrate with FlowForce Server, it is recommended to register FlowForce Server with LicenseServer first, and then all the other Altova server products will be registered automatically. After that, you can proceed to assign licenses to all these products.

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

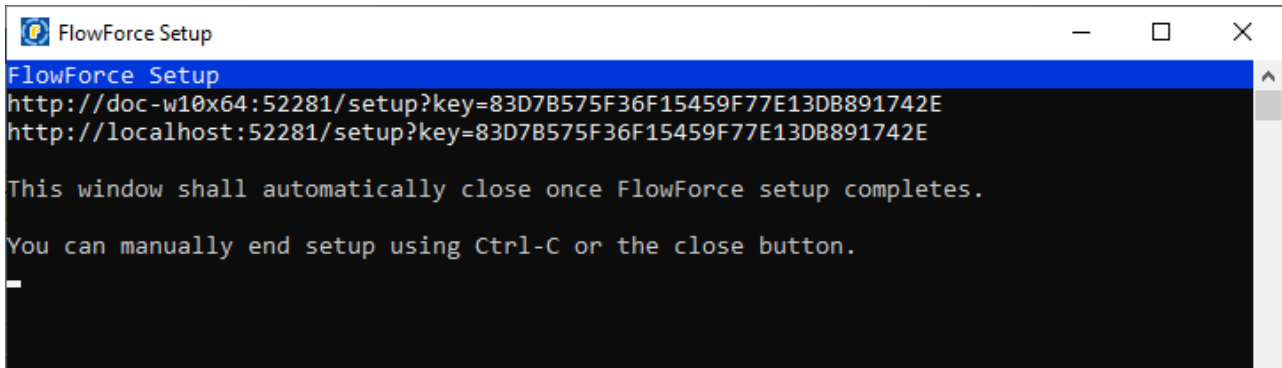
```
<ProgramFilesFolder>\Altova\FlowForceServer2024\bin\FlowForceServer.exe
```

Invoking Setup page

After the installation wizard has informed you that FlowForce Server has successfully been installed, you will need to access the FlowForce Server Setup page on which you will proceed with server configuration. The Setup page provides a centralized way of configuring and managing server instances. To access the Setup page, make sure the *Invoke FlowForce Setup* check box is selected and then click **Finish**. This will open the Setup page in a new browser window.

Alternatively, you can access the Setup page from the Start menu: Navigate to the Start menu and select **Altova FlowForce Server 2024 > FlowForce Server Setup Page**.

On Windows, together with the Setup page, a Command Prompt window also opens (*screenshot below*). This window remains open for the duration of the setup and will close automatically after you have clicked the **Finish setup** button at the bottom of the Setup page.



```
FlowForce Setup
http://doc-w10x64:52281/setup?key=83D7B575F36F15459F77E13DB891742E
http://localhost:52281/setup?key=83D7B575F36F15459F77E13DB891742E

This window shall automatically close once FlowForce setup completes.

You can manually end setup using Ctrl-C or the close button.
```

For more information about configuring FlowForce Server on the Setup page, see [Configuration via Setup Page](#) ⁴⁵.

Alternatively, you can configure your server instance via the configuration files and CLI. For details, see [Configuration via Configuration Files and CLI](#) ⁶⁴.

Installing on Windows Server Core

Windows Server Core has no GUI and must be installed via the command line. See the section [Installing on Windows Server Core](#) ²⁴ for information about how to do this.

Uninstalling FlowForce Server

Uninstall FlowForce Server as follows:

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

Evaluation license

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

2.1.1.2 Install on Windows Server Core

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

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

Important: Keep the MSI file!

Note the following points:

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

Register FlowForce Server with LicenseServer

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

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

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

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

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

Useful commands

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

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

```
start /wait msixec /i FlowForceServer.msi /q
echo %errorlevel%
```

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

```
start /wait msixec /i FlowForceServer.msi /q /L*v! <pathToInstallLogFile>
```

To modify the installation:

```
msixec /m FlowForceServer.msi
```

To repair the installation:

```
msixec /r FlowForceServer.msi
```

To uninstall FlowForce Server:

```
msixec /x FlowForceServer.msi
```

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

```
start /wait msixec /x FlowForceServer.msi /q /L*v! <pathToUninstallLogFile>
```

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

```
msixec /i FlowForceServer.msi INSTALLER_LANGUAGE=<languageCode>
```

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

2.1.1.3 Install LicenseServer

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

During the installation process of FlowForce Server, check or uncheck the option for installing LicenseServer as appropriate. Note the following points:

- If you have not installed LicenseServer yet, leave the default settings as is. The wizard will install the latest version on the computer where you are running the wizard.
- If you have not installed LicenseServer yet and want to install Altova LicenseServer on another computer, clear the check box *Install Altova LicenseServer on this machine* and choose **Register Later**. In this case, you will need to install LicenseServer separately and register FlowForce Server afterwards.

- If LicenseServer has already been installed on your computer but is a lower version than the one indicated by the installation wizard, leave the default setting (for upgrading to the newer version) as is. In this case, the installation wizard will automatically upgrade your LicenseServer version. The existing registration and licensing information will be carried over to the new version of LicenseServer.
- If LicenseServer has already been installed on your computer or network and has the same version as the one indicated by the wizard, do the following:
 - Clear the check box *Install Altova LicenseServer on this machine*.
 - Under *Register this product with*, choose the LicenseServer with which you want to register FlowForce Server. Alternatively, choose **Register Later**. Note that you can always select **Register Later** if you want to ignore the LicenseServer associations and carry on with the installation of FlowForce Server.

For information about how to register and license FlowForce Server with [Altova LicenseServer](#), see the section [License FlowForce Server](#) ²⁶.

LicenseServer versions

Note the following information about LicenseServer versions:

- Altova products must be licensed either (i) with a version of LicenseServer that corresponds to the installed FlowForce Server version or (ii) with a later version of LicenseServer.
- The LicenseServer version that corresponds to the current version of FlowForce Server is **3.14**.
- On Windows, you can install the corresponding version of LicenseServer as part of the FlowForce Server installation or install LicenseServer separately. On Linux and macOS, you must install LicenseServer separately.
- Before a newer version of LicenseServer is installed, any older one must be de-installed.
- At the time of LicenseServer de-installation, all registration and licensing information held in the older version of LicenseServer will be saved to a database on your server machine. This data will be imported automatically into the newer version when the newer version is installed.
- LicenseServer versions are backwards compatible. They will work with older versions of FlowForce Server.
- The latest version of LicenseServer available on the Altova website. This version will work with any current or older version of FlowForce Server.
- The version number of the currently installed LicenseServer is given at the bottom of the [LicenseServer configuration page](#) (all tabs).

2.1.1.4 License FlowForce Server

In order to use FlowForce Server, you must license it with Altova LicenseServer. Licensing is a two-step process:

1. **Register FlowForce Server** with LicenseServer.
2. **Assign a license** to FlowForce Server from LicenseServer. Download the latest version of LicenseServer from the [Altova website](#) and install it on your local machine or a machine on your network.

These steps are described in this section. For detailed information, see the [LicenseServer user manual](#) on the [Altova website](#).

You can register FlowForce during the installation process or at a later stage, when you configure FlowForce Server on the FlowForce Server Setup page. For details, see [Register FlowForce Server](#)²⁸. After you have registered FlowForce Server with LicenseServer, you need to [assign a license to FlowForce Server](#)²⁸. You can assign a license to FlowForce Server immediately after installation, in parallel with server configuration on the Setup page, or after you have finished configuring the server on the Setup page.

2.1.1.4.1 Start LicenseServer

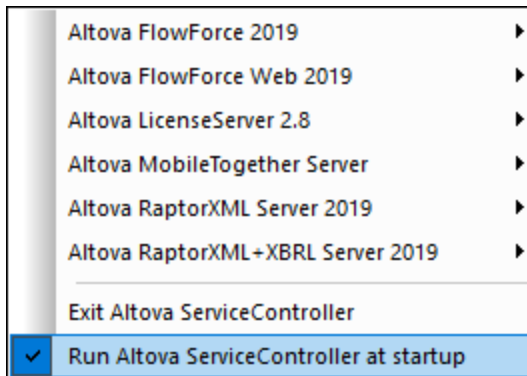
After you installed LicenseServer (alone or together with FlowForce Server), LicenseServer will automatically start after the installation process has been completed. If LicenseServer is not running, you can start it via Altova ServiceController (*details below*).

Altova ServiceController

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

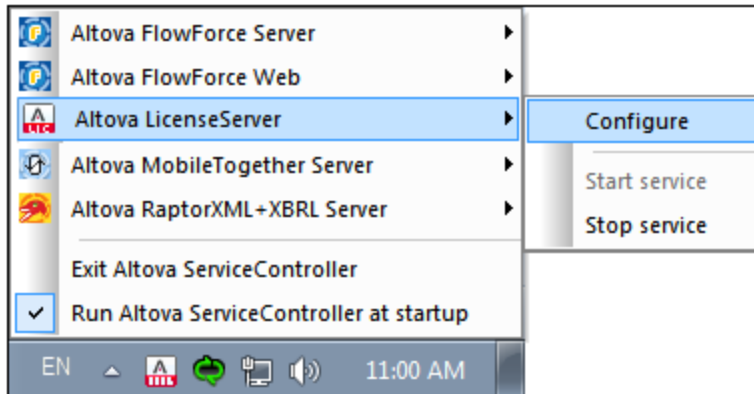


To specify that ServiceController starts automatically on logging in to the system, click the **ServiceController** icon in the system tray to display the **ServiceController** menu (*screenshot below*), and then toggle on the command **Run Altova ServiceController at Startup**. (This command is toggled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



Start LicenseServer

To start LicenseServer, click the **ServiceController** icon in the system tray, hover over **Altova LicenseServer** in the menu that pops up (see *screenshot below*), and then select **Start Service** from the LicenseServer submenu. If LicenseServer is already running, then the *Start Service* option will be disabled. You can also stop the service via ServiceController. If you install FlowForce Server together with LicenseServer, LicenseServer will automatically start after installation has been completed.



2.1.1.4.2 Register FlowForce Server

To be able to license FlowForce Server with Altova LicenseServer, you must first register FlowForce Server with LicenseServer. You can register FlowForce Server with Altova LicenseServer during [the installation process](#)²² or at a later stage (from [the Setup page](#)⁴⁵ or from the command line).

Registration from the command line

To register FlowForce Server from the command line interface, use the `licenseserver` command and supply the address of the LicenseServer machine (see *below*).

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

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

```
FlowForceServer licenseserver localhost
```

After successful registration, go to the [Client Management tab of LicenseServer's configuration page](#) or use the [assignlicense](#)³⁸⁸ command to assign a license to FlowForce Server.

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

2.1.1.4.3 Assign License to FlowForce Server

After successful registration of FlowForce Server, it will be listed in the Client Management tab of the configuration page of LicenseServer. Go there and [assign a license](#) to FlowForce Server. Alternatively, you can assign a license via the [assignlicense](#)³⁸⁸ command.

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

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

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

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

FlowForceServer and MapForceServer licensing

FlowForce Server Advanced Edition and MapForce Server Advanced Edition will run only on machines with two or more cores.

When assessing the number of cores you should license, take into account the data volume you need to process and the processing time your business environment is expected to allow for. In most scenarios, a larger number of cores means more volume of data processed in less time. Given below are a few application-specific tips:

- FlowForce Server runs as a multi-threaded application. If the number of concurrent requests to the server is big, an insufficient number of cores will lead to latency (waiting times). For example, if you are exposing jobs as Web services, there may be hundreds of concurrent requests from clients. In this case, FlowForce Server will significantly benefit from a larger number of cores.
- MapForce Server will utilize a single core at a time, per mapping. Therefore, if you need to run multiple mappings simultaneously, a larger number of cores is highly recommended. For example, when MapForce Server runs under FlowForce Server management, several mapping jobs may overlap and run concurrently, depending also on the setup. Note, however, that if the volumes processed by your mappings are extremely big, latency could still occur.

In addition to the above, note that there are various external factors that typically influence the processing volumes and times that your server is capable of handling (for example, the hardware, the current load on the CPU, memory allocation of other applications running on the server). In order to get the most accurate performance measurements, it is generally advisable to first run the tools in your environment and expose them to actual factors and data specific to your business.

Single-thread execution

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

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

Estimate of core requirements

There are various external factors that influence the data volumes and processing times your server can handle

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

2.1.2 Setup on Linux

This section describes the installation and licensing of FlowForce Server on Linux systems (Debian, Ubuntu, CentOS, RedHat). The setup of FlowForce Server comprises the following procedures:

1. [Installing FlowForce Server](#) ³¹
2. [Installing LicenseServer](#) ³²
3. [Starting LicenseServer](#) ³³
4. [Registering FlowForce Server with LicenseServer](#) ³⁴
5. [Assigning a license to FlowForce Server](#) ³⁴
6. [Configuring a server instance via the FlowForce Server Setup page or configuration files](#) ³⁶

In order for FlowForce Server to work, it must be registered and licensed with an [Altova LicenseServer](#) on your local machine or on another machine on your network. If LicenseServer is already installed and running on your network, you can skip Steps 2 and 3 and proceed with licensing and instance configuration.

Note that you can postpone registration and assigning a license and carry out these tasks within the framework of or after the configuration of FlowForce Server. However, it is mandatory to register FlowForce Server first, and only then will you be able to assign a license to FlowForce Server. Note that assigning a license to FlowForce Server does not require that FlowForce Server be running; only LicenseServer must be running.

System Requirements (Linux)

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

Prerequisites

- Perform installation either as **root** user or as a user with **sudo** privileges.
- The previous version of FlowForce Server must be uninstalled before a new one is installed.
- If you are installing FlowForce Server with other Altova server products, it is recommended that you install FlowForce Server first.
- The following libraries are required as a prerequisite to install and run the application. If the packages below are not already available on your Linux machine, run the `yum` command (or `apt-get` if applicable) to install them.

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

2.1.2.1 Install on Linux

FlowForce Server is available for installation on Linux systems. Its installation and setup procedure is described below. Perform installation either as the `root` user or as a user with `sudo` privileges.

Uninstall FlowForce Server

If you need to uninstall a previous version of FlowForce Server, do this as follows. In the Linux command line interface (CLI), you can check which Altova server products are installed with the following commands:

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

If FlowForce Server is not installed, go ahead with the installation as documented below in *Install FlowForce Server*.

If you need to uninstall an old version of FlowForce Server, do this with the following commands:

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

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

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

Download the FlowForce Server Linux package

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

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

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

Install FlowForce Server

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

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

Install FlowForce Server using the relevant command:

```
[Debian]:  sudo dpkg --install flowforceserver-2024-debian.deb
[Ubuntu]:  sudo dpkg --install flowforceserver-2024-ubuntu.deb
[CentOS]:  sudo rpm -ivh flowforceserver-2024-1.x86_64.rpm
[RedHat]:  sudo rpm -ivh flowforceserver-2024-1.x86_64.rpm
```

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

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

```
/opt/Altova/FlowForceServer2024
```

2.1.2.2 Install LicenseServer

In order for FlowForce Server to work, it must be licensed via an [Altova LicenseServer](#) on your network. On Linux systems, [Altova LicenseServer](#) will need to be installed separately. Download LicenseServer from the [Altova website](#) and copy the package to any directory on the Linux system. Install it in the same way you installed FlowForce Server (see [previous topic](#)³¹).

```
[Debian]:  sudo dpkg --install licenseserver-3.14-debian.deb
[Ubuntu]:  sudo dpkg --install licenseserver-3.14-ubuntu.deb
[CentOS]:  sudo rpm -ivh licenseserver-3.14-1.x86_64.rpm
[RedHat]:  sudo rpm -ivh licenseserver-3.14-1.x86_64.rpm
```

The LicenseServer package will be installed at the following path:

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

For information about how to register and license FlowForce Server with [Altova LicenseServer](#), see the section [License FlowForce Server](#)³³. Also see the [LicenseServer documentation](#) for more detailed information.

LicenseServer versions

Note the following information about LicenseServer versions:

- Altova products must be licensed either (i) with a version of LicenseServer that corresponds to the installed FlowForce Server version or (ii) with a later version of LicenseServer.
- The LicenseServer version that corresponds to the current version of FlowForce Server is **3.14**.
- On Windows, you can install the corresponding version of LicenseServer as part of the FlowForce Server installation or install LicenseServer separately. On Linux and macOS, you must install LicenseServer separately.
- Before a newer version of LicenseServer is installed, any older one must be de-installed.
- At the time of LicenseServer de-installation, all registration and licensing information held in the older version of LicenseServer will be saved to a database on your server machine. This data will be imported automatically into the newer version when the newer version is installed.
- LicenseServer versions are backwards compatible. They will work with older versions of FlowForce Server.
- The latest version of LicenseServer available on the Altova website. This version will work with any

- current or older version of FlowForce Server.
- The version number of the currently installed LicenseServer is given at the bottom of the [LicenseServer configuration page](#) (all tabs).

2.1.2.3 License FlowForce Server

In order to use FlowForce Server, you must license it with Altova LicenseServer. Licensing is a two-step process:

1. **Register FlowForce Server** with LicenseServer.
2. **Assign a license** to FlowForce Server from LicenseServer. Download the latest version of LicenseServer from the [Altova website](#) and install it on your local machine or a machine on your network.

These steps are described in this section. For detailed information, see the [LicenseServer user manual](#) on the [Altova website](#).

Before you register FlowForce Server, make sure that [LicenseServer is running](#)³³. Then you can register FlowForce Server with LicenseServer via [the FlowForce Server Setup page or from the command line](#)³⁴. When you have registered FlowForce Server, you can proceed to [assign a license](#)³⁴ to FlowForce Server. You can also assign a license at a later stage, for example, after you have finished configuring FlowForce Server on the Setup page.

2.1.2.3.1 Start LicenseServer

When you have installed LicenseServer, it will automatically be started. In case it is not running, you can use one of the following options to start it: (i) you can be the root user and leave out the `sudo` keyword from the commands listed below (leaving out `sudo` is optional), or (ii) you can run the `sudo` command as a normal user with the corresponding permissions for `sudo`.

Start LicenseServer

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

```
sudo systemctl start licenseserver
```

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

```
sudo systemctl stop licenseserver
```

Check status of daemons

To check if a daemon is running, run the following command, replacing `<servicename>` with the name of the daemon you want to check:

```
sudo service <servicename> status
```

2.1.2.3.2 Register FlowForce Server

To be able to license FlowForce Server with Altova LicenseServer, you must first register FlowForce Server with LicenseServer. You can register FlowForce Server with Altova LicenseServer from [the Setup page](#)⁴⁵ or from the command line.

Registration via the command line

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

```
sudo /opt/Altova/FlowForceServer2024/bin/flowforceserver licenseserver [options]
ServerName-Or-IP-Address
```

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

```
sudo /opt/Altova/FlowForceServer2024/bin/flowforceserver licenseserver localhost
```

Notice also that the location of the FlowForce Server executable is:

```
/opt/Altova/FlowForceServer2024/bin/
```

If you have installed FlowForce Server together with other Altova server products, it is recommended to register FlowForce Server with LicenseServer first. After this, all the other installed Altova server products will be registered automatically. After successful registration, go to the [Client Management tab of LicenseServer's configuration page](#) or use the `assignlicense`³⁸⁸ command to assign a license to FlowForce Server.

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

2.1.2.3.3 Assign License to FlowForce Server

After successful registration of FlowForce Server, it will be listed in the Client Management tab of the configuration page of LicenseServer. Go there and [assign a license](#) to FlowForce Server. Alternatively, you can assign a license via the `assignlicense`³⁸⁸ command.

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

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

Note: Each Altova server product license can be used for only one client machine at a time, even if the

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

FlowForceServer and MapForceServer licensing

FlowForce Server Advanced Edition and MapForce Server Advanced Edition will run only on machines with two or more cores.

When assessing the number of cores you should license, take into account the data volume you need to process and the processing time your business environment is expected to allow for. In most scenarios, a larger number of cores means more volume of data processed in less time. Given below are a few application-specific tips:

- FlowForce Server runs as a multi-threaded application. If the number of concurrent requests to the server is big, an insufficient number of cores will lead to latency (waiting times). For example, if you are exposing jobs as Web services, there may be hundreds of concurrent requests from clients. In this case, FlowForce Server will significantly benefit from a larger number of cores.
- MapForce Server will utilize a single core at a time, per mapping. Therefore, if you need to run multiple mappings simultaneously, a larger number of cores is highly recommended. For example, when MapForce Server runs under FlowForce Server management, several mapping jobs may overlap and run concurrently, depending also on the setup. Note, however, that if the volumes processed by your mappings are extremely big, latency could still occur.

In addition to the above, note that there are various external factors that typically influence the processing volumes and times that your server is capable of handling (for example, the hardware, the current load on the CPU, memory allocation of other applications running on the server). In order to get the most accurate performance measurements, it is generally advisable to first run the tools in your environment and expose them to actual factors and data specific to your business.

Single-thread execution

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

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

Estimate of core requirements

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

2.1.2.4 Configure Instance

After you have installed FlowForce Server, you will need to proceed with server configuration. Methods 1 and 2 enable you to configure the server via the FlowForce Server Setup page. The Setup page provides a centralized way of configuring and managing server instances. Method 3 enables you to configure the server via the command line and configuration files. For more information about server configuration, see the methods below.

Method 1

If you run Linux with a graphical user interface on your local machine, run the FlowForce Web server executable with the setup command as shown below:

```
sudo /opt/Altova/FlowForceServer2024/bin/flowforcewebserver setup
```

Method 2

If you want to connect to the Setup page from a browser on a different machine, run the same setup command as above and add the `--listen` option and, if necessary, with the `--key` option (see below).

After you have used Method 1 or 2 above, the terminal will display two alternative URLs for the setup page that you can copy-paste into your browser's address bar. In the event that the first URL does not work, you can use the second one. For information about server configuration on the Setup page, see [Configuration via Setup Page](#)⁴⁵.

Method 3

The third method enables you to configure your server instance via the command line and configuration files. For details, see [Configuration via Configuration Files and CLI](#)⁶⁴.

Options of the setup command

The `setup` command supports the following options:

`--listen`

By default, every time when you run the setup, the URL of the setup page is regenerated on a free random port (e.g., `http://localhost:50492/setup`). The `--listen` option enables you to specify an alternative interface/port combination to listen to (other than `localhost` or `127.0.0.1`). This might be useful if you want to access the Setup page from a browser on a different machine.

Note the following points:

- It is not recommended to make the setup run privileged with the actual data directory and bind it to an external network interface. If you intend to do that, the next option (`--key`) is useful.
- Do not use the same port as the normal (non-setup) FlowForce Web Server or FlowForce Server instance, because when the instance runs, that port will already be in use.

If the binding address (interface) is non-local, you may need to configure the operating system's firewall so as to enable access through the designated port.

Example

For example, the command `flowforcewebserver setup --listen=0.0.0.0:10008` would make the setup listen on port 10008 on all interfaces.

--key

The `--key` option enables you to set an access key for the setup page. With this option set, it is possible to save the setup page only if the correct access key is provided in the URL. The key can be any string that will be included in the URL. The setup page is not available continuously but only for the duration of the setup operation. For example, if you run a Linux command like the one below,

```
flowforcewebserver setup --listen=wild.berries.com:8015
--key=all_cats_love_fish
--datadir=/var/opt/Altova/FlowForceServer2024/data
```

the URL to connect to will be as follows:

```
http://wild.berries.com:8015/setup?key=all_cats_love_fish
```

Note that the setup page does not use HTTPS, because it used itself to configure the HTTPS parameters.

2.1.3 Setup on macOS

This section describes the installation and licensing of FlowForce Server on macOS systems. The setup of FlowForce Server comprises the following procedures:

1. [Installing FlowForce Server](#) ³⁸
2. [Installing LicenseServer](#) ³⁹
3. [Starting LicenseServer](#) ⁴⁰
4. [Registering FlowForce Server with LicenseServer](#) ⁴⁰
5. [Assigning a license to FlowForce Server](#) ⁴¹
6. [Configuring a server instance via the FlowForce Server Setup page or configuration files](#) ⁴²

In order for FlowForce Server to work, it must be registered and licensed with an [Altova LicenseServer](#) on your local machine or on another machine on your network. If LicenseServer is already installed and running on your network, you can skip Steps 2 and 3 and proceed with licensing and instance configuration.

Note that you can postpone registration and assigning a license and carry out these tasks within the framework of or after the configuration of FlowForce Server. However, it is mandatory to register FlowForce Server first, and only then will you be able to assign a license to FlowForce Server. Note that assigning a license to FlowForce Server does not require that FlowForce Server be running; only LicenseServer must be running.

System Requirements (macOS)

Note the following system requirement:

- macOS 12 or newer

Prerequisites

Note the following prerequisites:

- Perform installation as the `root` user or as a user with `sudo` privileges.
- The previous version of FlowForce Server must be uninstalled before a new one is installed.
- If you are installing FlowForce Server with other Altova server products, it is recommended to install FlowForce Server first to preserve the correct associations of FlowForce Server with other Altova server products.
- The macOS machine must be configured so that its name resolves to an IP address. This means that you must be able to successfully ping the host name from the Terminal using the command `ping <hostname>`.

2.1.3.1 Install on macOS

FlowForce Server is available for installation on macOS systems. The installation and setup procedure is described below.

Uninstall FlowForce Server

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

```
sudo launchctl unload /Library/LaunchDaemons/com.altova.FlowForceServer2024.plist
```

To check whether the service has been stopped, open the Activity Monitor in Finder and make sure that FlowForce Server is not in the list. In the Applications folder in Finder, 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 following command:

```
sudo rm -rf /usr/local/Altova/FlowForceServer2024/
```

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

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

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

Install FlowForce Server

To install FlowForce Server, follow the instructions below:

1. Download the disk image (`.dmg`) file of FlowForce Server from the Altova website (<http://www.altova.com/download.html>).
2. Click to open the downloaded disk image (`.dmg`). This causes the FlowForce Server installer to appear as a new virtual drive on your computer.
3. On the new virtual drive, double-click the installer package (`.pkg`).
4. Go through the successive steps of the installer wizard. These are self-explanatory and include one step in which you have to agree to the license agreement before being able to proceed. See also [Licensing FlowForce Server](#)⁴⁰.
5. To eject the drive after installation, right-click it and select **Eject**.

The FlowForce Server package will be installed in the folder:

```
/usr/local/Altova/FlowForceServer2024 (application binaries)
/var/Altova/FlowForceServer (data files: database and logs)
```

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

```
sudo launchctl load /Library/LaunchDaemons/com.altova.FlowForceServer2024.plist
```

Installing LicenseServer and other Altova server products

The virtual drive also enables you to [install LicenseServer](#)³⁹ and, if necessary, other Altova server products: StyleVision Server, MapForce Server, and RaptorXML Server (the installer for Altova RaptorXML+XBRL Server is available separately). You can always install any Altova server product separately at a later time. Standalone installers are available in the [Altova Download Center](#).

2.1.3.2 Install LicenseServer

In order for FlowForce Server to work, it must be licensed via an [Altova LicenseServer](#) on your network. The LicenseServer installation package is available on the virtual drive you mounted [in the previous step](#)³⁸. To install LicenseServer, double-click the installer package included on the virtual drive and follow the on-screen instructions. You will need to accept the license agreement for installation to proceed.

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

The LicenseServer package will be installed in the following folder:

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

For information about how to register FlowForce Server with [Altova LicenseServer](#) and license it, see [Licensing on macOS](#)⁴⁰.

LicenseServer versions

Note the following information about LicenseServer versions:

- Altova products must be licensed either (i) with a version of LicenseServer that corresponds to the installed FlowForce Server version or (ii) with a later version of LicenseServer.
- The LicenseServer version that corresponds to the current version of FlowForce Server is **3.14**.
- On Windows, you can install the corresponding version of LicenseServer as part of the FlowForce Server installation or install LicenseServer separately. On Linux and macOS, you must install LicenseServer separately.
- Before a newer version of LicenseServer is installed, any older one must be de-installed.
- At the time of LicenseServer de-installation, all registration and licensing information held in the older version of LicenseServer will be saved to a database on your server machine. This data will be imported automatically into the newer version when the newer version is installed.
- LicenseServer versions are backwards compatible. They will work with older versions of FlowForce Server.

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

2.1.3.3 License FlowForce Server

In order to use FlowForce Server, you must license it with Altova LicenseServer. Licensing is a two-step process:

1. **Register FlowForce Server** with LicenseServer.
2. **Assign a license** to FlowForce Server from LicenseServer. Download the latest version of LicenseServer from the [Altova website](#) and install it on your local machine or a machine on your network.

These steps are described in this section. For detailed information, see the [LicenseServer user manual](#) on the [Altova website](#).

Before you register FlowForce Server, make sure that [LicenseServer is running](#)⁴⁰. Then you can register FlowForce Server with LicenseServer via [the FlowForce Server Setup page or from the command line](#)⁴⁰. When you have registered FlowForce Server, you can proceed to [assign a license](#)⁴¹ to FlowForce Server. You can also assign a license at a later stage, for example, after you have finished configuring FlowForce Server on the Setup page.

2.1.3.3.1 Start LicenseServer

When you have installed LicenseServer, it will automatically be started. In case it is not running, you must have administrator (root) privileges to be able to start it. If you are logged in as `root`, you can leave out the `sudo` keyword from the command listed below.

Start LicenseServer

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

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

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

2.1.3.3.2 Register FlowForce Server

To be able to license FlowForce Server with Altova LicenseServer, you must first register FlowForce Server with LicenseServer.

You can register FlowForce Server with Altova LicenseServer from [the Setup page](#)⁴⁵ or from the command line.

Registration via the command line

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

```
sudo /usr/local/Altova/FlowForceServer2024/bin/FlowForceServer licenseserver [options]
ServerName-Or-IP-Address
```

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

```
sudo /usr/local/Altova/FlowForceServer2024/bin/FlowForceServer licenseserver localhost
```

Notice also that the location of the FlowForce Server executable is:

```
/usr/local/Altova/FlowForceServer2024/bin/
```

If you have installed FlowForce Server together with other Altova server products, it is recommended to register FlowForce Server with LicenseServer first. After this, all the other installed Altova server products will be registered automatically. After successful registration, go to the [Client Management tab of LicenseServer's configuration page](#) or use the `assignlicense`³⁸⁸ command to assign a license to FlowForce Server.

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

2.1.3.3.3 Assign License to FlowForce Server

After successful registration of FlowForce Server, it will be listed in the Client Management tab of the configuration page of LicenseServer. Go there and [assign a license](#) to FlowForce Server. Alternatively, you can assign a license via the `assignlicense`³⁸⁸ command.

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

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

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

client machine.

FlowForceServer and MapForceServer licensing

FlowForce Server Advanced Edition and MapForce Server Advanced Edition will run only on machines with two or more cores.

When assessing the number of cores you should license, take into account the data volume you need to process and the processing time your business environment is expected to allow for. In most scenarios, a larger number of cores means more volume of data processed in less time. Given below are a few application-specific tips:

- FlowForce Server runs as a multi-threaded application. If the number of concurrent requests to the server is big, an insufficient number of cores will lead to latency (waiting times). For example, if you are exposing jobs as Web services, there may be hundreds of concurrent requests from clients. In this case, FlowForce Server will significantly benefit from a larger number of cores.
- MapForce Server will utilize a single core at a time, per mapping. Therefore, if you need to run multiple mappings simultaneously, a larger number of cores is highly recommended. For example, when MapForce Server runs under FlowForce Server management, several mapping jobs may overlap and run concurrently, depending also on the setup. Note, however, that if the volumes processed by your mappings are extremely big, latency could still occur.

In addition to the above, note that there are various external factors that typically influence the processing volumes and times that your server is capable of handling (for example, the hardware, the current load on the CPU, memory allocation of other applications running on the server). In order to get the most accurate performance measurements, it is generally advisable to first run the tools in your environment and expose them to actual factors and data specific to your business.

Single-thread execution

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

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

Estimate of core requirements

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

2.1.3.4 Configure Instance

After you have installed FlowForce Server, you will need to proceed with server configuration. To configure the server on macOS, you can select any of the methods below.

Method 1

The first method enables you to configure your server instance via the FlowForce Server Setup page. To access this page, do one of the following:

- Open the **Finder**, navigate to **Applications**, and double-click the **FlowForce Server 2024** icon.
- Or open the terminal and use the following command to enter the setup mode:

```
sudo /usr/local/Altova/FlowForceServer2024/bin/FlowForceWebServer setup
```

After you have used the command above, the terminal will display two URLs for the Setup page that you can copy-paste into your browser's address bar. In the event that the first URL does not work, you can use the second one.

For information about server configuration on the Setup page, see [Configuration via Setup Page](#)⁴⁵.

Method 2

The second method enables you to configure your server instance via the command line and configuration files. For details, see [Configuration via Configuration Files and CLI](#)⁶⁴.

Options of the setup command

The `setup` command supports the following options:

`--listen`

By default, every time when you run the setup, the URL of the setup page is regenerated on a free random port (e.g., `http://localhost:50492/setup`). The `--listen` option enables you to specify an alternative interface/port combination to listen to (other than `localhost` or `127.0.0.1`). This might be useful if you want to access the Setup page from a browser on a different machine.

Note the following points:

- It is not recommended to make the setup run privileged with the actual data directory and bind it to an external network interface. If you intend to do that, the next option (`--key`) is useful.
- Do not use the same port as the normal (non-setup) FlowForce Web Server or FlowForce Server instance, because when the instance runs, that port will already be in use.

If the binding address (interface) is non-local, you may need to configure the operating system's firewall so as to enable access through the designated port.

Example

For example, the command `flowforcewebserver setup --listen=0.0.0.0:10008` would make the setup listen on port 10008 on all interfaces.

`--key`

The `--key` option enables you to set an access key for the setup page. With this option set, it is possible to save the setup page only if the correct access key is provided in the URL. The key can be any string that will be included in the URL. The setup page is not available continuously but only for the duration of the setup operation. For example, if you run a Linux command like the one below,

```
flowforcewebserver setup --listen=wild.berries.com:8015
```

```
--key=all_cats_love_fish
--datadir=/var/opt/Altova/FlowForceServer2024/data
```

the URL to connect to will be as follows:

```
http://wild.berries.com:8015/setup?key=all_cats_love_fish
```

Note that the setup page does not use HTTPS, because it used itself to configure the HTTPS parameters.

2.1.4 Upgrade FlowForce Server

When you upgrade to a newer version of FlowForce Server, the license of your previous version will be used automatically for the newer version if, during installation:

- the new version is registered with the same LicenseServer as the one with which the previous version of FlowForce Server was registered;
- you accept the license agreement of FlowForce Server.

The simplest way to carry over a license from the previous version of FlowForce Server to the newer version is to let the installation process implement the required steps. The relevant steps during the installation process are listed below in the order in which they occur:

1. Let the installer register the new version of FlowForce Server with the LicenseServer that holds the license used by the older version of FlowForce Server.
2. Accept the license agreement of FlowForce Server. (If you do not accept the agreement, the new version will not be installed.)

Note: If you do not register FlowForce Server with the correct LicenseServer during the installation process, you will need to register and license FlowForce Server manually with your alternative LicenseServer.

Important information about instance-data directory

With 2024 R2, the default path to the instance-data directory has changed and contains no year anymore. This is the new recommended naming convention, because in-place upgrades are safer and faster. 2024 R2 has also introduced a new way of installing server instances: via the Setup page as opposed to the installation wizard. The new method provides more control over the location of your instance data. For more information, see [Configuration via Setup Page](#)⁴⁵.

2.2 Configuration via Setup Page

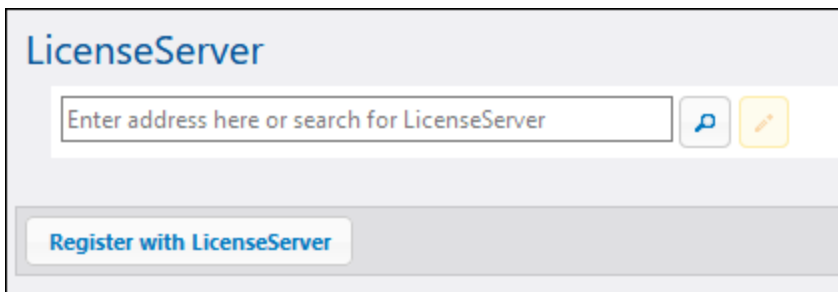
After you have installed FlowForce Server, you must configure it. This section explains how to configure it via the FlowForce Server Setup page. Alternatively, you can use [configuration files](#)⁶⁴ to configure FlowForce Server.

Overview of Setup page

The Setup page provides a centralized way of managing and configuring server instances. The Setup page consists of two sections: *LicenseServer* and *Instances*.

LicenseServer

FlowForce Server must be registered with Altova LicenseServer (see [Altova LicenseServer](#)). If you do not specify a LicenseServer host during installation, you will need to do this via the Setup page, by entering the address or host name of the machine where Altova LicenseServer runs (*LicenseServer field shown below*). This can be the address of the local machine if LicenseServer is installed locally or a network address. After entering the relevant address/host name, click **Register with LicenseServer**.



The registration of FlowForce Server with LicenseServer is also possible during the installation process (Windows) or from the command line (Windows, Linux, macOS). The details are provided in the following topics:

- [Register FlowForce Server \(Windows\)](#)²⁸
- [Register FlowForce Server \(Linux\)](#)³⁴
- [Register FlowForce Server \(macOS\)](#)⁴⁰

Instances

The *Instances* section of the Setup page enables you to carry out the following actions:

- Create a new server instance
- Add an existing instance
- Have an overview of existing server instances
- Configure instance parameters
- Install and uninstall the services (the same tasks can be carried out via the [install](#)³⁹⁵ and [uninstall](#)⁴⁰² commands, respectively)
- Upgrade the database to the latest version (fulfills the same function as the [upgradedb](#)⁴⁰³ command)
- Migrate the instance directory to a new location (fulfills the same function as the [migratedb](#)³⁹⁷ command)
- Reduce the size of the database files (fulfills the same function as the [compactdb](#)³⁸⁹ command)
- Reset the password of the `root` user to the default value (fulfills the same function as the [resetpassword](#)³⁹⁹ command)

For information about creating, configuring, and installing a server instance, see *Configuration Procedures* below. Importantly, before compacting the database files, upgrading the database, and migrating your instance data, you must [stop](#)⁶¹ the FlowForce Server and FlowForce Web Server services.

Note that you can have multiple server instances, but only one server instance can be running at a given time.

How to access Setup page

Depending on your operating system, the instructions on how to access the Setup page vary:

- [Windows](#)²²
- [Linux](#)³⁶
- [macOS](#)⁴²

Configuration procedures

The configuration of FlowForce Server on the Setup page involves the procedures described below. Some of the procedures are optional (e.g., setting up SSL encryption).

1. [Creating a new server instance](#)⁴⁶
2. [Configuring instance parameters](#)⁴⁷, which includes:
 - a. Setting ports to connect to FlowForce Server and FlowForce Web Server
 - b. Setting up SSL encryption
 - c. Configuring the default time zone
 - d. Configuring cluster-related settings (*Advanced Edition*)
3. [Installing the services](#)⁶⁰
4. [Starting the services](#)⁶¹

After you have finished configuring FlowForce Server, you can [log in](#)⁸¹ and carry out various [administration tasks](#)⁷².

2.2.1 Create New Server Instance

After you have [opened the Setup page](#)⁴⁶, you can proceed to create a new server instance as follows:

1. Click **New Instance**.
2. Enter the folder where instance data will be stored. This can be any directory; note, however, that you must select your local drive for instance-data storage. Alternatively, you can opt for [the default location](#)¹⁸ of the instance-data directory.
3. Click **Initialize New Instance**.

As soon as you add an instance, the current state as well as various configuration options will appear. The state will inform you whether the server version is compatible and whether the FlowForce Server and FlowForce Web Server services are installed and running.

Next step

The next step is to configure various parameters, such as ports, SSL encryption, the default time zone. To configure these parameters, click the **Configure Parameters** button on the Setup page, which will open a separate page. For details, see [Configure Instance Parameters](#)⁴⁷.

2.2.2 Configure Instance Parameters

The FlowForce Server Setup page enables you to configure various network settings, including the interface and port on which FlowForce Server and FlowForce Web Server should listen. Most of the settings from the Setup page can also be defined by means of [configuration files](#)⁶⁴. The settings defined on the Setup page will be preserved when you install a new minor version of FlowForce Server. If you install a new major version, the settings will be preserved only if you opted to migrate your data from the previous major version during installation.

To get access to the network settings, click **Configure Parameters** on the Setup page. The screenshots below illustrate the connection settings for FlowForce Web Server and FlowForce Server, respectively.

FlowForce Web Server

Unencrypted Connection

Enabled:

Bind address: All interfaces (0.0.0.0) other: Port:

Host name:

SSL Encrypted Connection

Enabled:

Bind address: All interfaces (0.0.0.0) other: Port:

Host name:

Certificate file:

Private Key file:

Certificate Chain file:

Settings

Default time zone:

FlowForce Server

Unencrypted Connection

Enabled:

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

Host name:

SSL Encrypted Connection

Enabled:

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

Host name:

Certificate file:

Private Key file:

Certificate Chain file:

Master Instance Encrypted Connection

Enabled:

Bind address: All interfaces (0.0.0.0) other: Port:

The available settings are listed below.

Unencrypted connection enabled

Select this check box to enable plain HTTP (unencrypted) connections to FlowForce Web Server/FlowForce Server.

Bind address

FlowForce Web Server

On Windows, the FlowForce Web Server administration interface is available by default on all network interfaces on port 8082. On Linux and Mac OS, the port number is chosen randomly during installation. To specify a custom address other than `Local only` or `All interfaces`, enter it in the *Other* text box.

FlowForce Server

The default setting for FlowForce Server accepts only requests from the same machine (`127.0.0.1`) on port 4646, through an unencrypted connection. If you intend to start jobs as Web services via plain HTTP from remote machines, select `All interfaces (0.0.0.0)` from the *Bind address* combo box.

If the binding address (interface) is non-local, you may need to configure the operating system's firewall so

as to enable access through the designated port.

Port

Specifies the TCP port on which FlowForce Web Server/FlowForce Server should listen. The port must not be already in use.

Host name

FlowForce Web Server

The *Host name* field, if non-empty, sets a fixed host name that is used for the binding. It sets the name of the machine running FlowForce Web Server, and other machines on the network will use this name to connect to it. FlowForce automatically detects the appropriate host name to use. If you set this field explicitly, then automatic detection will be overridden. Depending on the network configuration in your organization, you may need to use a value such as `somehost` or `somehost.example.org`.

The host name associated with a binding is used for SSL (see [Enable SSL for FlowForce Server/Web Server](#)⁵³) and by Altova ServiceController* on Windows. If SSL is enabled, the host name must match the *Common Name* property of the certificate.

Setting a host name is meaningful if the bind address is not local (i.e., when the *Bind address* field is set to something other than `Local (127.0.0.1)`).

FlowForce Server

The field *Host name* designates the host name bound to the interface where FlowForce Server listens for connections from clients that access jobs exposed as Web services. Setting a host name is meaningful when the *Bind address* field is not set to `Local (127.0.0.1)`. Depending on the network configuration in your organization, you may need to use a value such as `somehost` or `somehost.example.org`.

The host name associated with a binding is used for SSL (see [Set Up SSL Encryption](#)⁵⁰). If SSL is enabled, the host name must match the *Common Name* property of the certificate. The host name is also used by Altova ServiceController* on Windows. If the host name is not set, FlowForce automatically detects the first appropriate host name to be used by Altova ServiceController.

If the host name is configured, the FlowForce Web interface can show clickable links to navigate to jobs exposed as Web services, including links in the [Active Triggers and Services](#)⁸⁷ section of the Home page. Also, the **Call Web Service** button becomes available in the *Service* section of the job configuration page. Clicking this button enables you to call the Web service in a new browser window. For more information, see [Jobs as Web Services](#)¹⁵⁸.

* *Altova ServiceController* is an application that enables you to conveniently start, stop, and configure Altova services on Windows systems.

SSL Encrypted Connection

SSL (Secure Sockets Layer) is an encryption security protocol that encrypts data transmitted between a client and a server. In FlowForce Server, you can encrypt the following HTTP connections with SSL certificates:

- The connection between a browser and FlowForce Web Server
- The connection between a Web service consumer (e.g., a client application) and the FlowForce Server service

- The internal connection between FlowForce Web Server and FlowForce Server

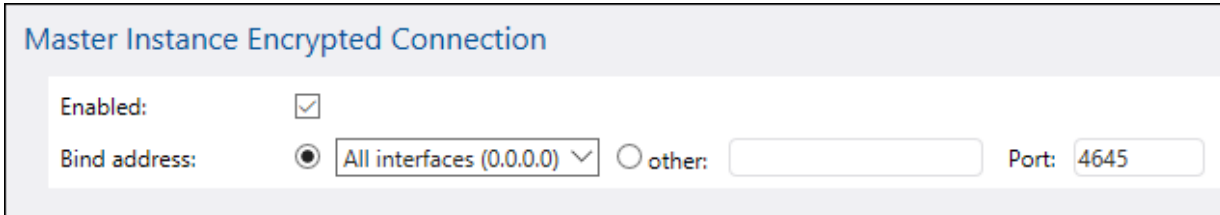
For more details, see [Set Up SSL Encryption](#) ⁵⁰.

Settings

In addition to encrypted and unencrypted connections, you can also set the default time zone for FlowForce Web Server. You can also set the default time zone through the [Administration](#) ¹¹⁸ page.

Master Instance Encrypted Connection (Advanced Edition)

The settings shown in the screenshot below must be configured if FlowForce Server is a master instance in a cluster of multiple machines that run FlowForce Server.



Master Instance Encrypted Connection

Enabled:

Bind address: All interfaces (0.0.0.0) other:

Port:

When you have finished defining the network settings and other parameters, click **Save Changes**. This action will redirect you to the main Setup page. The next step is to [install the services](#) ⁶⁰.

2.2.3 Set Up SSL Encryption

SSL (Secure Sockets Layer) is an encryption security protocol that encrypts data transmitted between a client and a server. In FlowForce Server, you can encrypt the following HTTP connections with SSL certificates:

- The connection between a browser and FlowForce Web Server
- The connection between a Web service consumer (e.g., a client application) and the FlowForce Server service
- The internal connection between FlowForce Web Server and FlowForce Server

For the first two connections, you need an SSL certificate and a private key corresponding to that certificate. For security reasons, you might want to use a separate SSL certificate and private key for each connection. If you want to use the same certificate and private key for both connections, this requires that both FlowForce Server and FlowForce Web Server have the same fully qualified domain name (FQDN). For example, if FlowForce Web Server listens on `https://somehost:8083`, then FlowForce Server should listen on `https://somehost:4647`. Note that you can always change the port later; only the host name is important in this case.

For the last connection, there is no need for a third certificate and private key pair—you can use the same SSL certificate as for FlowForce Server. In this case, FlowForce Web Server acts as an HTTP client for FlowForce Server.

FlowForce Server Advanced Edition

If you use FlowForce for exchanging AS2 data, you can also use SSL certificates to sign or encrypt data as part of the AS2 service.

SSL encryption procedures

If you need to encrypt communications with the SSL protocol, follow the instructions below. In this example, we have used the open-source [OpenSSL toolkit](#) to set up SSL encryption. The steps listed below, therefore, need to be carried out on a computer on which [OpenSSL](#) is available. [OpenSSL](#) typically comes pre-installed on most Linux distributions and on macOS machines. It can also be [installed on Windows computers](#). For download links to installer binaries, see the [OpenSSL Wiki](#).

1. Generate a private key

SSL requires that a private key be installed on FlowForce Server. This private key will be used to encrypt all data sent to clients. To create the private key, use the following OpenSSL command:

```
openssl genrsa -out private.key 2048
```

The command above creates a file called `private.key`, which contains your private key. Note where you save the file. You will need the private key (i) to generate the Certificate Signing Request (CSR) and (ii) to be installed on FlowForce Server (see *Step 7 below*). The value `2048` refers to the 2048-bit size of the private key, which is the minimum encryption strength normally accepted by a certification authority.

Private key requirements

Because FlowForce Server runs unattended, enabling SSL requires that the certificate's private key be *unencrypted*, which means it must not be protected with a password. Otherwise, the private key cannot be used by FlowForce Server. For this reason, the file that stores the private key must have restricted access and be accessible only to entitled personnel in your organization.

To check whether the private key is password-protected or unencrypted, open the private key file using a text editor or the command line. An *encrypted* private key begins with the following lines:

```
-----BEGIN RSA PRIVATE KEY-----  
Proc-Type: 4, ENCRYPTED  
DEK-Info: AES-256-CBC,DFC3FAD546517ED6336CFF72AA23F6C7
```

To decrypt the private key, you can use the following OpenSSL command:

```
openssl rsa -in enc.key -out dec.key
```

Note also the following requirements:

- The private key must be in PEM (Privacy Enhanced Mail) format. The file extension of PEM files is usually `.pem`, but it can also be `.key`, `.cert`, `.cer`, or `.crt`.
- The private key must be stored securely.

2. Create a Certificate Signing Request (CSR)

A Certificate Signing Request (CSR) is sent to a certificate authority (CA), such as [VeriSign](#) or [Thawte](#), to request a public key certificate. The CSR is based on your private key and contains

information about your organization. Create a CSR with the following OpenSSL command (which provides the private-key file, `private.key`, that was created in Step 1, as one of its parameters):

```
openssl req -new -nodes -key private.key -out my.csr
```

During the generation of the CSR, you will need to give information about your organization (*listed below*). This information will be used by the certificate authority to verify your company's identity.

- Country
- Locality (the city where your business is located)
- Organization (your company name). Do not use special characters - these will invalidate your certificate.
- Common Name (the DNS name of your server). This must exactly match your server's official name, that is, the DNS name that client apps will use to connect to the server.
- A challenge password. Keep this entry blank.

3. Buy an SSL certificate

In the next step, you need to purchase an SSL certificate from a recognized certificate authority (CA), such as [VeriSign](#) or [Thawte](#). For the rest of these instructions, we follow the VeriSign procedure. The procedure with other CAs is similar.

1. Go to the [VeriSign website](#).
2. Click **Buy SSL Certificates**.
3. Different types of SSL certificates are available. For FlowForce Server, Secure Site or Secure Site Pro certificates are sufficient. EV (extended verification) is not necessary, since there is no "green address bar" for users to see.
4. Proceed through the sign-up process and fill in the information required to place your order.
5. When prompted for the CSR (*created in Step 2: Create a Certificate Signing Request above*), copy and paste the contents of the `my.csr` file into the order form.
6. Pay for the certificate with your credit card.

Obtaining public key certificates from an SSL certificate authority (CA) typically takes two to three business days. Please take this into account when setting up your FlowForce Server.

Alternative method: Create a self-signed SSL certificate

Alternatively, if FlowForce Server runs on a private network, you can configure your own SSL root certification authority (provided you are entitled to do this in your organization). No browser or operating system trusts such an authority by default. Therefore, you will need to configure each machine (or browser, depending on the case) that connects to FlowForce Server to trust your self-signed root certificate. Otherwise, the browser will still display warnings, or the Web service call will not be successful. For more information, see [Create Self-Signed SSL Certificates](#)⁵⁵.

4. Receive the public key from CA

Your certificate authority will complete the enrollment process over the next two to three business days. During this time, you might get emails or phone calls to check whether you are authorized to request an SSL certificate for your DNS domain. Please work with the authority to complete the process.

After the authorization and enrollment process has been completed, you will get an email containing the public key of your SSL certificate. The public key will be in plain text form or attached as a `.cer` file.

5. Save the public key to a file

For use with FlowForce Server, the public key must be saved in a `.cer` file. If the public key was supplied as text, copy-paste all the lines from

```
--BEGIN CERTIFICATE--  
...  
--END CERTIFICATE--
```

into a text file that we will call `mycertificate.cer`.

6. Save CA's intermediate certificates to a file

When you sign a certificate with a certificate authority, you will receive a single intermediate certificate or intermediate certificates (primary and secondary) that form the chain of trust between your server and the certificate authority. If you receive a primary certificate and a secondary certificate, you must combine them into a single file (the so-called *Certificate Chain File*), as shown in the instructions below.

1. Using a text editor such as Notepad, create a new text file. In our example, we have called it `intermediate.pem`. You can choose another file name and extension.
2. Open each intermediate certificate in a text editor and copy-paste its contents into `intermediate.pem`. Importantly, the certificate text must be copied in reverse order: The secondary intermediate certificate goes first; the primary one goes second (*code listing below*).

```
--BEGIN CERTIFICATE--  
... (secondary intermediate certificate) ...  
--END CERTIFICATE--  
--BEGIN CERTIFICATE--  
... (primary intermediate certificate) ...  
--END CERTIFICATE
```

3. Save the changes. You will need `intermediate.pem` on the FlowForce Setup page later.

7. Enable SSL for FlowForce Server/Web Server

The instructions below show you how to enable SSL for FlowForce Web Server (the service which drives the Web administration interface of FlowForce) and FlowForce Server (the service responsible for exposing Web services created from FlowForce jobs to HTTP(S) clients).

Note: If you created self-signed certificates, each client browser must be configured to trust your self-signed certificate authority. See [Import Root Certificate](#)⁵⁷ for more information.

To enable SSL for FlowForce Server/Web Server, follow the instructions below:

1. Open the FlowForce Server [Setup page](#)⁴⁵ and click **Configure Parameters**.
2. Depending on your needs, navigate to the settings of FlowForce Web Server or FlowForce Server.

3. Select the *Enabled* check box in the SSL Encrypted Connection section.
4. Select All Interfaces (0.0.0.0) in the *Bind address* drop-down list. This value means that FlowForce Server/FlowForce Web Server will be accessible externally, not only from the current machine.
5. Enter the host (domain) name and port where FlowForce Server/Web Server should listen to SSL-encrypted connections. The domain name entered in the *Host name* field must correspond to the SSL certificate's Common Name. The port must not be in use. Depending on the case, you can also enter a different IP address. If you enter an IP address in the *Other* field without entering a host name, this IP address must correspond to the SSL certificate's Common Name.
6. Enter the path to the certificate in the *Certificate File* field. The certificate must be in PEM (Privacy Enhanced Mail) format. The file extension of PEM files is usually *.pem*, but it can also be *.key*, *.cert*, *.cer*, or *.crt*. The certificate must be issued for the domain name on which FlowForce Server is running.
7. Enter the path to the private key file in the *Private Key File* field. The private key must be in PEM (Privacy Enhanced Mail) format. The file extension of PEM files is usually *.pem*, but it can also be *.key*, *.cert*, *.cer*, or *.crt*. The private key must be stored securely. In order for the private key to be usable in FlowForce, it must not be password protected.
8. Enter the path to the certificate chain file in the *Certificate Chain File* field. If there is no intermediate certificate, you can leave this field empty. If there are several intermediate certificates available, then you must combine all of them into the so-called *Chain File* (see *Save CA's Intermediate Certificates to a File (Step 6) above*).
9. Click **Apply settings and restart FlowForce services** at the bottom of the Setup page.

Optionally, clear the *Enabled* check box under *Unencrypted Connection*. Note that this will make FlowForce Server/Web Server unavailable through plain HTTP, so you should take this step only after the SSL encrypted connection starts working. Instead of disabling the HTTP connection completely, you may want to restrict it to local connections only (the *Local only* option in the *Bind address* drop-down list).

Note the following points:

- The browser (or connecting client) will display warnings if the Common Name (CN) of the SSL certificate does not correspond to the domain name or IP address where FlowForce Server runs.
- If you are using self-signed certificates, the browser (or connecting client) will display warnings if you have not added your CA root certificate to the operating system's certificate store or to the browser's certificate store (see [Import Root Certificates](#)⁵⁷)

8. Test SSL communication

You can now use any SSL testing tool to check whether secure communication with your server via HTTPS is working properly. This will tell you (i) whether the public key certificate file was properly constructed with the intermediate trust chain in Step 6, and (ii) whether your server can be reached properly through the firewall.

2.2.3.1 Create Self-Signed SSL Certificates

This demo shows you how to create self-signed SSL certificates for FlowForce Server running on a private network. Note that this demo is intentionally simplified and not suitable for use in production. Your organization will likely have specific security policies concerning SSL certificates and could use SSL tools other than the ones described below.

Creating self-signed SSL certificates involves the following procedures:

1. Creating a root certificate
2. Creating a FlowForce certificate
3. Importing the root certificate

For more information about each step, see the subsections below.

Prerequisites

This example makes use of the OpenSSL toolkit (<https://www.openssl.org/>) to generate self-signed certificates. Note that OpenSSL is an open source library and may need to be compiled before you can use it at the command line. The compilation and installation instructions for OpenSSL vary for each operating system. [OpenSSL](#) typically comes pre-installed on most Linux distributions and on macOS machines. You can quickly check if OpenSSL is installed by typing the command below (it displays the current OpenSSL version):

```
openssl version
```

OpenSSL can also be [installed on Windows computers](#). For download links to installer binaries, see the [OpenSSL Wiki](#).

Step 1: Create root certificate

The instructions below explain how to create a root certificate. The root certificate will be used to sign the server certificate (Step 2 below).

1. Create a directory that will store all certificates used in this demo (e.g., `C:\secure`). This will be the working directory for all subsequent OpenSSL commands. Then change to this directory from the command line:

```
cd C:\secure
```

2. For this demo, we will be issuing certificates with OpenSSL extensions. To make this possible, find the `openssl.cnf` file of your OpenSSL distribution and copy it to the working directory created in the previous step.
3. Create the root key that acts as your certificate authority's (CA) private key. Be aware that the root private key is the most sensible piece of your public key infrastructure, so it must always be generated and stored in a secure environment (in this demo, it is stored in `C:\secure`).

```
openssl genrsa -aes256 -out root.key 2048
```

When prompted, type a password to protect the root key. You will subsequently need this password to sign certificate requests.

4. Create the root certificate that is the public certificate of your certificate authority. The command below generates a self-signed certificate for the private key created above, with a validity of 3650 days. Notice that the `-config` parameter points to the `openssl.cnf` file in the same directory. The `-extensions` parameter refers to the `v3_ca` extension (section) defined in `openssl.cnf`.

```
openssl req -config openssl.cnf -extensions v3_ca -x509 -new -nodes -key root.key -
sha256 -days 3650 -out root.pem
```

When prompted, enter information about your organization, for example:

```
Country Name (2 letter code) [AU]: AT
State or Province Name (full name) [Some-State]: .
Locality Name (eg, city) []: Vienna
Organization Name (eg, company) [Internet Widgits Pty Ltd]: MyCompany Ltd
Organizational Unit Name (eg, section) []: IT
Common Name (eg, YOUR name) []: Demo CA
Email Address []: test@example.org
```

You can fill in the required fields as applicable to your organization. For the `Common Name` field, enter the name of your self-signed certificate authority (`Demo CA` in this example).

Step 2: Create FlowForce certificate

The next step is to create the actual certificate that will be used for SSL encryption (by FlowForce Server, FlowForce Web Server, or both). The FlowForce certificate will be signed with the root certificate that was created in Step 1.4. Follow the instructions below:

1. Create the private key, using the OpenSSL command below. The private key accompanies your self-signed certificate used by FlowForce.

```
openssl genrsa -out flowforce.key 2048
```

The private key must be in PEM (Privacy Enhanced Mail) format. The file extension of PEM files is usually `.pem`, but it can also be `.key`, `.cert`, `.cer`, or `.crt`. In order for the private key to be usable in FlowForce, it must not be password-protected. The private key must be stored securely.

2. Open the working `openssl.cnf` file and add the following section to it:

```
[ server_cert ]
# Extensions for server certificates (`man x509v3_config`).
basicConstraints = CA:FALSE
nsCertType = server
nsComment = "OpenSSL Generated Server Certificate"
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer:always
keyUsage = critical, digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth
subjectAltName=DNS:server.my.domain.com
```

Make sure to change the `subjectAltName` (Subject Alternative Name) so that it corresponds to the FQDN (fully qualified domain name) of the machine where FlowForce Server runs. In this example, it is set to `server.my.domain.com`. Specifying a subject alternative name is required by Google Chrome 58 and later; otherwise, your self-signed certificate will generate a `NET::ERR_CERT_COMMON_NAME_INVALID` error (see the [Chrome Help page](#)).

3. Create a Certificate Signing Request (CSR), using the command shown below. Notice that the `-config` parameter points to the `openssl.cnf` file edited previously. The `-extension` parameter refers to the `server_cert` extension defined in `openssl.cnf`.

```
openssl req -config openssl.cnf -extensions server_cert -new -nodes -key
flowforce.key -out flowforce.csr
```

4. When prompted, enter information about your organization, for example:

```
Country Name (2 letter code) [AU]: AT
State or Province Name (full name) [Some-State]: .
Locality Name (eg, city) []: Vienna
Organization Name (eg, company) [Internet Widgits Pty Ltd]: MyCompany Ltd
Organizational Unit Name (eg, section) []: IT
Common Name (eg, YOUR name) []: server.my.domain.com
Email Address []: test@example.org
```

For the `Common Name` field, make sure to enter the FQDN (fully qualified domain name) of the host machine where FlowForce Server runs. Leave the challenge password field empty when prompted.

5. Sign the FlowForce certificate with the root certificate. Note that, in a production environment, the root certificate does not normally sign server certificates directly; instead, intermediate certificates are used. The command below signs the `flowforce.csr` certificate request against the root certificate created previously and creates the `flowforce.crt` file (which is the server certificate required in FlowForce Server):

```
openssl x509 -extfile openssl.cnf -extensions server_cert -req -in flowforce.csr -CA
root.pem -CAkey root.key -CAcreateserial -out flowforce.crt -days 365 -sha256
```

Summary

After taking Step 1 and Step 2, you must have the following certificates and keys:

- **root.key:** This is your certificate authority's (CA) private key. Store this file in a secure place; if this key becomes compromised, then anyone can generate browser-trusted certificates on your behalf.
- **root.pem:** This is the public certificate of your certificate authority. You will need to install (import) this certificate into the trusted certificates store of each machine (or browser) that needs to access FlowForce securely (Step 3 below).
- **flowforce.key:** This private key accompanies your self-signed certificate used by FlowForce (see next item).
- **flowforce.crt:** This is your self-signed certificate that will be used by FlowForce Server, FlowForce Web Server, or both.

Step 3: Import root certificate

When you create your own certificate authority (CA), the root certificate is self-signed; therefore, no browser will trust it by default. In order for an HTTP client (such as a browser) to trust your self-signed certificate, the certificate must be imported as follows:

- Into the operating system's trusted certificate store if the browser uses it. On Windows, for example, Google Chrome and Microsoft Edge use the operating system's certificate store, while Mozilla Firefox

uses its own store. On Linux, Google Chrome and Mozilla Firefox use their own certificate stores (see next point). On Mac, Safari uses the operating system's certificate store (Keychain Access).

- Into the trusted certificates store of the browser itself.

Notes

- The self-signed certificate must be imported for each client machine (or browser, if applicable) that will access FlowForce Server.
- When you enable SSL encryption between FlowForce Web Server and FlowForce Server, it is not sufficient to import the certificate into the browser. Your self-signed root CA certificate must also be trusted by the operating system.

Depending on your operating system, the instructions on how to import the root certificate vary. For details, see the subsections below.

Linux

On Linux, you can import a trusted certificate into the system's certificate store as shown below. Perform the following steps only if you are sure of the authenticity of the certificate you want to trust.

On Debian and Ubuntu, follow the steps below:

1. Copy the certificate file of the Web server to the following directory.

```
sudo cp /home/downloads/server_cert.crt /usr/local/share/ca-certificates/
```

2. Update the certificate store as follows:

```
sudo update-ca-certificates
```

On CentOS, follow the steps below:

1. Install the `ca-certificates` package:

```
yum install ca-certificates
```

2. Enable the dynamic certificate authority configuration feature:

```
update-ca-trust enable
```

3. Copy the server certificate to the following directory:

```
cp server_cert.crt /etc/pki/ca-trust/source/anchors/
```

4. Use the following command:

```
update-ca-trust extract
```

For cases in which you need to access the server only through the browser, it is sufficient to import the certificate into the browser certificate store. The exact instructions will vary for each browser. For

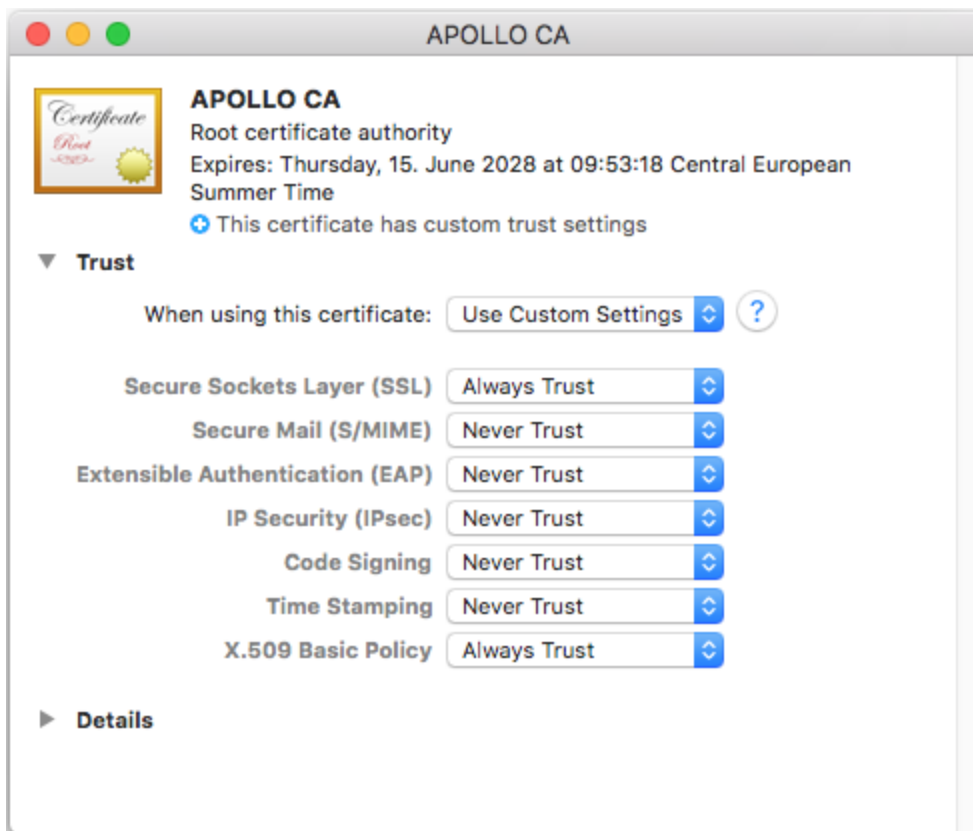
example, in Firefox 59.0.2, you can do this as follows:

1. Under **Options | Privacy & Security**, click **View Certificates**.
2. On **Authorities** tab, click **Import** and browse for the root certificate file created previously.
3. When prompted, select **Trust this CA to identify websites** and click **OK**.

Mac

On macOS, you can import a trusted certificate into Keychain Access as follows:

1. Run Keychain Access.
2. Click **System** and then click **Certificates**.
3. Open the **File** menu and click **Import Items**.
4. Browse for the trusted certificate and click **Open**.
5. Enter the Keychain Access password when prompted and then click **Modify Keychain**.
6. Double-click the certificate, expand the *Trust* section, and select **Always Trust**.



Windows

On Windows, you can manage certificates in the Microsoft Management Console (MMC) snap-in, either for your user account or for the computer account.

To open the Certificates snap-in for the *current Windows user*, run the following command in the command line:

```
certmgr.msc
```

To open the Certificates snap-in for the *computer account*, take the steps below:

1. Run `mmc` in the command line.
2. Go to the **File** menu of the MMC and click **Add/Remove Snap-in**.
3. Click **Certificates** and then click **Add**.
4. Select **Computer account** and click **Next**.
5. Select **Local computer** and then click **Finish**.

On Windows, you can import a trusted certificate into the system certificates store as follows:

1. Open the Windows certificate store *for the computer account*, see .
2. Expand the *Trusted Root Certification Authorities* folder of the *Certificates (Local Computer)* tree, right-click **Certificates**, select **All Tasks | Import** and follow the Certificate Import Wizard.

For more information, see the article [Import a Certificate on the Microsoft website](#).

Next step

After taking Steps 1-3, you can now enable SSL for FlowForce Server, FlowForce Web Server, and for the HTTP connection between them, which is described in Step 7 of the section [Set Up SSL Encryption](#)⁵³.

2.2.4 Install and Start Services

After you have configured [all the necessary parameters](#)⁴⁷, you can go on to install the services, start them, and finish the setup.

Install services

To install the services, click **Install Services** on the Setup page. As a result, the **Install FlowForce Services** dialog box will appear (*see below*). This dialog box gives an overview of the services to be installed and the location of the instance-data directory.

Most of the data in this dialog is for information purposes only. However, you can set the way the services will start after the reboot of the machine. On Windows, you can also select the logon user (*screenshot below*). You can also copy the start type and logon details from FlowForce Server to FlowForce Web Server by clicking the **Copy Parameters** button.

Install FlowForce services

Data directory:

FlowForce Server

Service name:

Display name:

Description:

Start type:

Logon user: Local System Other user

FlowForce Web Server

Service name:

Display name:

Description:

Start type:

Logon user: Local System Other user

After installation: Start services

After checking all the details and, possibly, modifying the start type, click **Install Services**.

Start services

As soon as you click **Install Services** in the **Install FlowForce Services** dialog box, the information about the instance will be updated and logged. By default, the services will start automatically after installation. In case the services have not started yet, click their respective **Start** buttons on the Setup page. The states of both services will switch to *Running*.

Alternative ways of starting services

Besides being able to start the services via the Setup page, you can also use some alternative ways. Depending on the operating system, the instructions vary (see *details below*).

Start services on Windows

To start the FlowForce Server and FlowForce Web Server services, click the [ServiceController](#)⁸¹ icon in the system tray, hover over the corresponding service in the menu that pops up, and then select **Start Service** from the service's submenu. If the services are already running, their respective *Start Service* options will be disabled. You can also stop the services via ServiceController. Note that you will be able to start FlowForce Server and FlowForce Web Server from ServiceController *only after* you have configured your server instance on the FlowForce Server Setup page. You can also start and stop FlowForce Server via the Setup page.

Start services on Linux

Since the FlowForce Server solution consists of two services, FlowForce Server and FlowForce Web Server, you need to start both of them. To start FlowForce Server, use the following command:

```
sudo systemctl start flowforceserver
```

To start FlowForce Web Server, use the following command:

```
sudo systemctl start flowforcewebserver
```

If at any time you need to stop the services, replace `start` with `stop` in the command above:

```
sudo systemctl stop flowforceserver
```

Note that you can start FlowForce Server and FlowForce Web Server *only after* you have configured your server instance on the FlowForce Server Setup page. You can also start and stop FlowForce Server via the Setup page.

Start services on macOS

Since the FlowForce Server solution consists of two services, FlowForce Server and FlowForce Web Server, you need to start both of them. To start FlowForce Server, use the following command:

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

To start FlowForce Server, use the following command:

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

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

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

Note that you can start FlowForce Server and FlowForce Web Server *only after* you have configured it on the FlowForce Server Setup page. You can also start and stop FlowForce Server via the Setup page.

Finish setup

The last step is to finish the setup by clicking **Finish Setup**. After you have finished setting up your server instance, you will need to [log on to the Web UI](#)⁸¹, where you can proceed with [administration tasks](#)⁷² and [job configuration](#)¹²⁸.

Information for Windows users

After you have finished the configuration of FlowForce Server on the Setup page, the FlowForce Server and FlowForce Web Server services will become available in [Altova ServiceController](#)⁸¹.

2.3 Configuration via Configuration Files and CLI

Besides being able to configure FlowForce Server via the [FlowForce Server Setup page](#)⁴⁵, you can also configure it via the configuration files and CLI. This section describes the [contents of the FlowForce Server application-data and instance-data directories](#)⁶⁴ and explains how to use the configuration files and the CLI to set up FlowForce Server.

The configuration of FlowForce Server via the configuration files and the CLI consists of the procedures listed below. Some of the procedures are optional (e.g., setting SSL encryption). The steps below assume that you have already [installed FlowForce Server](#)²¹.

1. [Creating a new FlowForce database via the CLI](#)³⁹⁴
2. [Configuring instance parameters via the configuration files, which includes](#)⁶⁷:
 - a. Setting ports to connect to FlowForce Server and FlowForce Web Server
 - b. Setting up SSL encryption
 - c. Configuring the default time zone
 - d. Configuring cluster-related settings (*Advanced Edition*)
3. [Installing the FlowForce Server and FlowForce Web Server services via the CLI](#)³⁹⁵
4. [Starting the services](#)⁶¹

Note that to be able to use the FlowForce Server functionality, you must register and license FlowForce Server with Altova LicenseServer. For information about registration and licensing, see the instructions corresponding to your operating system:

- [Licensing on Windows](#)²⁶
- [Licensing on Linux](#)³³
- [Licensing on macOS](#)⁴⁰

After you have finished configuring your service instance, you can proceed to [log on to the Web UI](#)⁸¹ and carry out [administration tasks](#)⁷².

2.3.1 Overview of Configuration Files

This topic provides an overview of configuration files stored in the application-data and instance-data directories. The paths to the application-data directory and instance-data directories depend on your operating system.

Application-data directory

The application-data directory (*table below*) contains two configuration files (`flowforceserver.ini` and `flowforcweb.ini`) that enable you to configure global configuration settings (currently, the language used in server logs and in error messages).

FlowForce Server application-data directory (APPDATADIR)	
Linux	<code>/var/opt/Altova/FlowForceServer2024</code>
macOS	<code>/var/Altova/FlowForceServer2024</code>
Windows	<code>C:\ProgramData\Altova\FlowForceServer2024</code>

Instance-data directory

The instance-data directory stores data generated by FlowForce Server and its users. This can be data related to jobs, triggers, system functions, server logs, and other files. The instance-data directory also contains two `.ini` configuration files. Administrators can edit the `.ini` configuration files with a text editor, as an alternative to changing settings from the [Web Administration Interface](#) ⁸¹, the [Setup page](#) ⁴⁷, and from the [Command Line Interface](#) ³⁸³.

The instance-data directories shown below are default paths. You can also select your custom location of the instance-data directory via [the FlowForce Server Setup page](#) ⁴⁵.

FlowForce Server instance-data directory (INSTANCEDIR)	
Linux	<code>/var/opt/Altova/FlowForceServer/data</code>
macOS	<code>/var/Altova/FlowForceServer/data</code>
Windows	<code>C:\ProgramData\Altova\FlowForceServer\data</code>

List of installed FlowForce instances

The `flowforceinstances.json` file contains a list of FlowForce instances that are managed by [the FlowForce Server Setup page](#) ⁴⁵. The table below shows the path to this file on different operating systems. This file is for information purposes only.

File with the list of FlowForce instances (INSTANCELIST)	
Linux	<code>/var/opt/Altova/SharedBetweenVersions</code>
macOS	<code>/var/Altova/SharedBetweenVersions</code>
Windows	<code>C:\ProgramData\Altova\SharedBetweenVersions</code>

Contents of APPDATADIR

The list below summarizes the configuration files in the application-data directory.

`flowforceserver.ini` Stores global configuration settings of FlowForce Server (currently, the language used in server logs and in error messages).

`flowforceweb.ini` Stores global configuration settings of FlowForce Web Server (currently, the language of the Web administration interface).

Note: Do not confuse this file with the `flowforceweb.ini` file stored in the `INSTANCEDIR` directory (*described below*).

Contents of INSTANCEDIR

The list below describes the contents of the instance-data directory.

<code>cache.db</code>	This database file stores data related to the caching feature of FlowForce (see Cache Job Results ¹⁴⁸).
<code>state.db</code>	This database file stores the volatile (that is, not configured) state of FlowForce.
<code>ffweb.log</code> <code>flowforce.log</code>	These files store the log of messages from FlowForce Web Server and FlowForce Server, respectively. This is applicable to Windows and macOS systems. On Debian, Ubuntu and CentOS or later, the log is written to the system log (<code>/var/log/syslog</code>).
<code>flowforce.db</code>	This database file stores the FlowForce Server object system, user data, active jobs, roles, and others.
<code>flowforcelog.db</code>	This database file stores the FlowForce Server logs.
<code>flowforce.ini</code>	This configuration file defines the port and listening interfaces of FlowForce Server. See also Instance Parameters in Configuration Files ⁶⁷ .
<code>flowforceweb.ini</code>	This configuration file defines the port and listening interfaces of FlowForce Web Server. See also Instance Parameters in Configuration Files ⁶⁷ .
<code>files</code>	Stores files associated with deployed functions.
<code>logs</code>	Contains captured output from job execution steps.
<code>tmp</code>	Stores temporary files.
<code>tools</code>	When other Altova server products (such as MapForce Server or StyleVision Server) are installed alongside FlowForce Server, this directory stores <code>.tool</code> files which enable FlowForce Server to work with these products. If this directory is empty, you can manually copy any tool files from the installation directory of the relevant product to this directory. You may need to edit a <code>.tool</code> file to set environment variables that may be required to execute MapForce mappings and StyleVision transformations (see Environment Settings ⁴³⁷).

2.3.2 Instance Parameters in Configuration Files

This topic describes structure of the `.ini` configuration files.

Structure of `.ini` files

The `flowforce.ini` file has the following structure:

```
[Listen]
active=1
host=127.0.0.1
port=4646
hostname=

[ListenSSL]
active=1
SSL=1
host=0.0.0.0
port=4647
hostname=

[SSL]
certificate=/path/to/certificate.crt
private_key=/path/to/private_key.key
certificate_chain=/path/to/certificate_chain

[Master]
host=0.0.0.0
port=4645
active=1
```

The `flowforceweb.ini` file has the following structure:

```
[Listen]
active=1
host=0.0.0.0
port=8082
hostname=example.domain.org

[ListenSSL]
active=1
SSL=1
host=0.0.0.0
port=8083
hostname=example.domain.org

[SSL]
certificate=path/to/certificate.crt
private_key=path/to/private_key.key
certificate_chain=/path/to/certificate_chain

[FlowForce]
```

```

host=127.0.0.1
port=4646
hostname=

```

The `.ini` files are organized into sections. For details, see the subsections below.

[Listen]

The `[Listen]` section defines the HTTP connection settings. You can define multiple `[Listen]` sections. Each `[Listen]` section must begin with `Listen` (e.g., `[ListenSSL]`). The available parameters are listed below.

active (Optional) Activates or deactivates the current `[Listen]` section. The valid values are 0 (disabled) and 1 (enabled). For example, `active=1` means that HTTP connections are enabled.

host Specifies the network bind address of FlowForce (Web) Server (e.g., `127.0.0.1`). This can be an IPv4 or IPv6 address. Use `0.0.0.0` to listen on all interfaces. For local access only, use `127.0.0.1`.

port Specifies the port on which FlowForce (Web) Server will listen. Make sure to specify the port that is not in use yet.

If the binding address (interface) is non-local, you may need to configure the operating system's firewall so as to enable access through the designated port.

hostname The *Host name* field, if non-empty, sets a fixed host name that is used for the binding. It sets the name of the machine running FlowForce Web Server, and other machines on the network will use this name to connect to it. FlowForce automatically detects the appropriate host name to use. If you set this field explicitly, then automatic detection will be overridden. Depending on the network configuration in your organization, you may need to use a value such as `somehost` or `somehost.example.org`.

The host name associated with a binding is used for SSL (see [Enable SSL for FlowForce Server/Web Server](#)⁵³) and by Altova ServiceController* on Windows. If SSL is enabled, the host name must match the *Common Name* property of the certificate.

* *Altova ServiceController is an application that enables you to conveniently start, stop, and configure Altova services on Windows systems.*

max_request_body_size This option enables you to specify the maximum size, in bytes, of HTTP requests to FlowForce Server/Web Server (e.g., `max_request_body_size=500000000`). The default limit is around 100 MB (100,000,000 bytes). You may need to set this option explicitly in the following situations:

1. If you call FlowForce Web services exposed as jobs and the HTTP request body is larger than the default limit.
2. If you deploy mappings from MapForce to FlowForce Server and the input files are larger than the default limit.

For Case 1 above, the option must be set only in the `flowforce.ini` file. For Case 2, the option must be set in both `flowforce.ini` and `flowforceweb.ini` files.

[ListenSSL]

This section enables you to configure the encrypted HTTPS connection settings. Besides all the parameters listed in the [Listen] section above, the [ListenSSL] section also has the `ssl` parameter that can be used to enable SSL support. The valid values are 0 (disabled) and 1 (enabled). If you enable SSL, the [SSL] section is required (see below).

[SSL]

The [SSL] section defines the SSL/HTTPS connection settings. This section is required if the HTTPS interface is active (see [ListenSSL] above). The available parameters are listed below.

<code>certificate</code>	Specifies the absolute path of the certificate file in PEM format.
<code>private_key</code>	Specifies the absolute path of the private key file in PEM format.
<code>certificate_chain</code>	(Optional) The path to the certificate chain file.

[FlowForce]

The [FlowForce] section applies only to FlowForce Web Server (`flowforceweb.ini`). This section enables you to configure the connection settings between FlowForce Web Server and FlowForce Server. This section must match the [Listen] section in `flowforce.ini`. The available parameters are described below.

<code>host</code>	Specifies the IP address or host name of FlowForce Server. If FlowForce Server is not bound to all interfaces, this value must be the same as in the [Listen] section of <code>flowforce.ini</code> ; otherwise, the value of the <code>host</code> parameter will be <code>127.0.0.1</code> . If SSL is enabled, this value must match the <i>Common Name</i> property of the certificate configured in the [SSL] section of <code>flowforce.ini</code> .
<code>port</code>	Specifies the TCP port on which FlowForce Web Server will connect to FlowForce Server. This value must be the same as in the corresponding [Listen] or [ListenSSL] section of <code>flowforce.ini</code> . If SSL is enabled on this port, the <code>host</code> and <code>hostname</code> (or just the <code>host</code> if the <code>hostname</code> is not present) must match the <i>Common Name</i> property of the certificate configured in the [SSL] section of <code>flowforce.ini</code> .
<code>hostname</code>	If non-empty, this field sets a fixed host name that is used by other machines on the network to connect to FlowForce jobs that are exposed as Web services ¹⁵⁸ . You may need to use a value like <code>somehost</code> or <code>somehost.example.org</code> , depending on the network configuration in your organization.

The host name is also used by Altova ServiceController*. If you do not set the host name, FlowForce detects automatically the first appropriate host name that will be used by Altova ServiceController.

If the hostname is configured, the FlowForce Web interface may show clickable links to navigate to jobs exposed as Web services, including links in the [Active Triggers and Services](#)⁸⁷ section of the Home page. Also, the **Call Web Service** button becomes available in the Service section of the job configuration page. This button enables you to call a Web service in a new browser window.

* *Altova ServiceController is an application that enables you to conveniently start, stop, and configure Altova services on Windows systems.*

- certificate** (Optional) Specifies the server certificate that will be accepted by FlowForce Server. If no certificate is given, the system root CA certificates will be used to verify the server certificate. If present, this value must match the certificate that FlowForce Server is using (specified in the `flowforce.ini` file).
- ssl** (Optional) Enables SSL support for the connection between FlowForce Web Server and FlowForce Server. The valid values are 0 (disabled) and 1 (enabled).

[FlowForceWeb]

This section applies only to FlowForce Web Server (the `flowforceweb.ini` file).

- timezone** Specifies the default time zone of FlowForce Web Server (e.g., `timezone=Europe/Berlin`). You can also configure the default time zone in the [Settings on the Administration page](#)¹¹⁸.

[Master] (Advanced Edition)

This section applies only to the `flowforce.ini` file. It is relevant when multiple FlowForce Server instances run in a cluster, and the current instance is the master instance

- active** Enables encrypted connection to this master instance. The valid values are 0 (disabled) and 1 (enabled).
- binding address** Specifies the binding address of the master FlowForce Server instance. Use `0.0.0.0` to listen on all interfaces.
- port** The port on which this master instance listens to requests from worker instances.

Encrypted connections between FFS and FFW

To enable encrypted connections between FlowForce Server and FlowForce Web Server, you need to configure `flowforce.ini` and `flowforceweb.ini` as shown below.

`flowforce.ini`

```
[ListenSSL]
active=1
ssl=1
host=0.0.0.0
port=4647
hostname=server.my.domain.com

[SSL]
certificate=C:\secure\flowforce.crt
private_key=C:\secure\flowforce.key
certificate_chain=
```

The port value 4647 refers to the default port for encrypted connections. The paths in the [SSL] section point to the locations of the certificate and private key for FlowForce Server.

`flowforceweb.ini`

```
[FlowForce]
host=127.0.0.1
port=4647
ssl=1
certificate=C:\secure\flowforce.crt
```

The host in this case is 127.0.0.1, because the communication between FFS and FFW is local. The port points to the port where FFS accepts encrypted connections, i.e., 4647. The `certificate` parameter refers to the local path of the FFS certificate file (or the path of the common certificate of FFS and FFW if they are using the same).

2.4 Administration Tasks

Administration tasks comprise the following procedures:

- [Creating and configuring users and roles](#) ⁷²
- [Configuring basic settings](#) ¹¹⁸ (the default time zone, parameters for the [/system/mail/send](#) ²⁷⁹ function, directory service and logging settings)
- [Backing up, restoring, migrating data](#) ⁷⁵
- Revisiting the Setup page (in case you need to change ports, enable SSL, etc.)
- [Localizing FlowForce Server](#) ⁷⁹

The administration procedures are on-demand tasks that you can carry out when necessary.

Revisit the Setup page

In case you need to revisit the Setup page, for example, to change ports, enable SSL, or configure a new server instance, you can do this at any time. For more information, see the instructions corresponding to your operating system:

- [Access the Setup page on Windows](#) ²²
- [Access the Setup page on Linux](#) ³⁶
- [Access the Setup page on macOS](#) ⁴²

2.4.1 Define Users and Roles

A user account is defined by a login name and password and has a set of access rights associated with it. Users access FlowForce Server for administrative purposes or as end users.

Access rights are determined by the privileges a user is granted. A user receives privileges in the following ways: (i) privileges inherited from roles the user is a member of and (ii) privileges assigned directly to the user. A role is defined by a set of privileges. A role is assigned privileges directly and/or inherits the privileges of another role that it is a member of. Privileges themselves are access rights to the various administrative functions and services of FlowForce Server. Examples of privileges are as follows: the right to override security settings, to set a user's own password, to stop any job.

Through the use of roles, user privileges can be defined in a hierarchical way. For example, the `SimpleAdmin` role has the `Stop any job` privilege. If `AdvancedAdmin` is a member of `SimpleAdmin`, `AdvancedAdmin` inherits the right to stop any job, regardless of the user who created this job, and could additionally be assigned the `Maintain users, roles and privileges` privilege. The hierarchical chain can then be further extended.

About users

A user is a person who logs on to FlowForce Server to create and monitor jobs, deploy MapForce mappings and StyleVision transformations, and configure various settings. The scope of actions available to users in FlowForce Server depends on the following:

- The permissions and privileges assigned to the users
- The permissions and privileges assigned to the roles that the users are members of

Two special users are predefined:

- The `root` user is the initial administrator user. By default, it has all permissions and privileges available in the system. Its initial name-password combination is `root-root`. The password can be changed at any time.
- The `anonymous` account is for anonymous users that access services exposed via the HTTP service interface (see [Jobs as Web Services](#)¹⁵⁸). It cannot be used for logging in to the Web UI, and it has no initial password.

For more information about how to create, edit, import, and delete users, see [Users](#)¹⁰⁹.

About roles

A role defines a set of privileges and permissions. It can be assigned to another role or to a user. A role's privileges automatically become the privileges of any other role or any user that the role is assigned to. A user can be assigned any number of roles. As a result, a user will have all the privileges defined in the multiple assigned roles.

Note that privileges are global, whereas permissions are defined per container.

The following roles are predefined:

- The `authenticated` role is automatically assigned to every user *except* the `anonymous` account.
- The `all` role is automatically assigned to every user *including* the `anonymous` user.





For more information about how to create, edit, import, and delete roles, see [Roles](#)¹¹².








About privileges

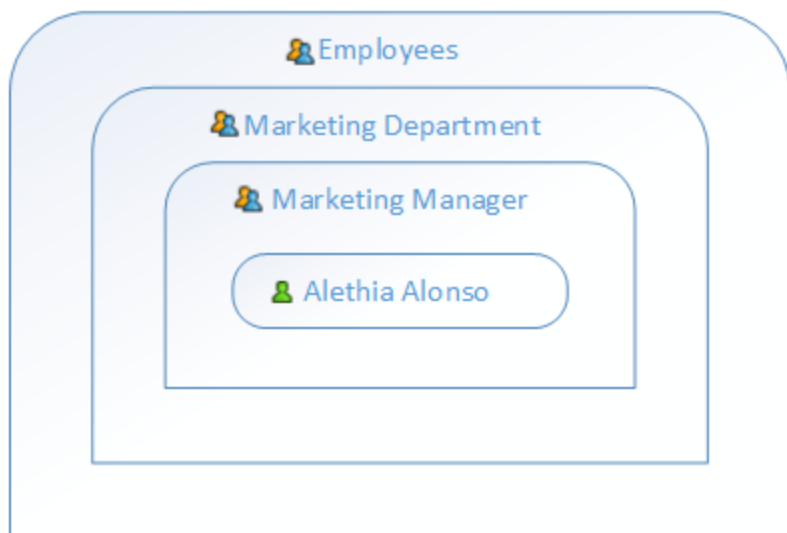
A privilege is an activity that a user is allowed to carry out (e.g., set a password, read users and roles, stop any job, etc.). A user can be assigned zero to all of the available privileges. It is recommended to assign privileges via roles rather than to assign privileges directly to the user. The assigning of privileges and roles to a user is done by a user that has been assigned this privilege. Initially, it is the `root` user that has this privilege.

Inheritance

You can assign privileges directly to a user (e.g.,  Alethia Alonso) or to a particular role (e.g.,  Marketing Manager). It is recommended to assign privileges to roles rather than to individual users, because it simplifies the maintenance and management of privileges in the long term.

You can model the hierarchy of your organization in FlowForce Server, by assigning roles to other roles. The diagram below illustrates a sample organization, for which three roles and one user have been defined. The  Employees role contains a role called  Marketing Department. This means that the privileges and permissions granted to the  Employees role will automatically be inherited by the users belonging to the  Marketing Department role.

The  Marketing Department role contains the  Marketing Manager role. In this case, the  Marketing Manager role will inherit all the privileges from the  Marketing Department and  Employees roles. A user called  Alethia Alonso is the marketing manager, and she has been assigned the  Marketing Manager role. This implies that she will inherit all the privileges from the broader roles.



Assigning a privilege

To assign a privilege to a user or role, click a user or role of interest in the Users or Roles tab (Administration page), respectively, and select the privilege(s) you wish to assign. The available privileges are summarized in the table below.

<i>Define execution queues</i>	This privilege allows defining settings related to job execution queues ¹⁶⁶ .
<i>Maintain global settings</i>	This privilege allows changing the FlowForce Server global settings (the time zone and mail server settings) on the Settings page. This is an administrative privilege and should only be granted to FlowForce Server administrators.
<i>Maintain users, roles and privileges</i>	This privilege allows adding, editing, and deleting the following data: users, roles, privileges, and passwords. This is an administrative privilege and should only be granted to FlowForce Server administrators. By default, only the <code>root</code> user has this privilege.
<i>Override security</i>	Users with this privilege can change container permissions without having the write 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 the <code>root</code> user has this privilege.
<i>Read users and roles</i>	By default, users can see only their own user accounts and any roles they are members of. When users are granted this privilege, they can see all existing users and roles. By default, only the <code>root</code> user has this privilege.
<i>Retrieve sensitive data</i>	This privilege allows retrieving and viewing passwords as plain text. By default, only the <code>root</code> user has this privilege.

<i>Set own password</i>	This privilege allows changing their password. Users who do not have this privilege need to have their password set by a FlowForce Server administrator. By default, the <code>authenticated</code> role has this privilege, which means that every user account, except for <code>anonymous</code> , also has this privilege.
<i>Stop any job</i>	This privilege allows stopping any running FlowForce Server job, regardless of the user who created it.
<i>View unfiltered log</i>	By default, users can see log entries related to configurations to which they have read access. Users with this privilege can read all log entries, including those not associated with a specific configuration. By default, only the <code>root</code> user has this privilege.

The tab [Administration | Reports | Privileges Report](#)¹¹⁴ provides a list of all privileges, with each privilege being listed together with all the users/roles that have that privilege.

2.4.2 Back Up, Restore, and Migrate Data

This section explains how to perform a backup in FlowForce Server, restore data, and copy FlowForce Server data from a previous [instance-data directory](#)¹⁸ to the current one.

2.4.2.1 Backup

This topic explains how to back up data in FlowForce Server. There are two possible options:

- From the Web administration interface. This type of backup includes only configuration data: jobs, credentials, deployed MapForce mappings and StyleVision transformations, resources. It does not include application settings or users. Any FlowForce Server user can import and export configuration data if their permissions on the respective object allow it.
- Administrative backup of the application data directory. This approach requires access to the FlowForce Server [instance-data directory](#)¹⁸ on the machine where FlowForce Server is installed. The application data directory includes all the data from the previous point as well as users and roles, including users and roles imported from a Directory Service such as Active Directory. The application data directory also includes application-level settings, such as email or LDAP server settings, password policies

Note: This topic does not cover backup and recovery of data external to FlowForce Server, such as files or directories that are input/output to jobs, FlowForce resources or local file-based databases. You will need to back up this data separately. It is recommended to keep all such external data (if possible) in the same directory for easier backup and maintenance.

Useful tips

In case you want to migrate data to a new machine in the future or restore it from a backup, the tips below will help you carry over data more easily:

- It is recommended to configure LicenseServer to have a fallback second server. For details, see the LicenseServer documentation (<https://www.altova.com/documentation>).
- It is recommended that all jobs should use standalone (not inline) [credentials](#)¹⁶². If you are using local (inline) credentials in jobs, all such jobs will have to be edited on a new server machine to match the user credentials linked to that operating system. By contrast, if you are using standalone credentials, you will only need to edit the standalone credentials on the new server machine.
- If you are running mapping functions deployed from MapForce, consider referring to file and folder paths and databases using [resources](#)⁴²³ instead of absolute references.
- As an alternative to creating and maintaining users and roles directly in FlowForce Server, you might want to use Windows Active Directory or another LDAP Server with support for Directory Services. For details, see [Changing the Directory Service Settings](#)¹¹⁹.

Partial backup from the Web administration interface

To perform a backup of selected objects, log in to the FlowForce Web administration interface, and use [the Export functionality](#)²⁸⁸. To restore data, use [the Import functionality](#)²⁹².

Note: You can import configuration data into a FlowForce Server instance that is of the same or later version than the one from where data was exported. Importing configuration data into an earlier version of FlowForce Server may work but should be avoided.

Backup of all FlowForce application data

The backup of all application data involves creating a copy of the FlowForce Server database ([INSTANCEDIR](#)¹⁸) in a safe location from which you can later restore it, if necessary. To save time and disk space, you will want the `INSTANCEDIR` directory to be as compact as possible. You can achieve this by performing the following optional steps *before* the actual backup:

1. Archive the old log records by creating a job that runs the [archive-log](#)²⁸⁴ function.
2. Delete old log records by creating a job that runs the [truncate-log](#)²⁸⁵ function.
3. Delete unused files by creating a job that runs the [cleanup-files](#)²⁸⁴ function.
4. Run the FlowForce Server executable with the [compactdb](#)³⁸⁹ function.

You can now proceed with the actual backup as follows:

1. Stop both the FlowForce Server and FlowForce Web Server services. Depending on your operating system, the instructions vary. For details, see Set Up FlowForce Server.
2. Create a copy of the `INSTANCEDIR` in a safe directory (preferably on a different machine or disk). By convention, we will call this copy `INSTANCEDIR_BACKUP` in subsequent steps (see [Data Restoration and Migration](#)⁷⁶).

The `private.db` file in the `INSTANCEDIR` contains sensitive information, such as passwords and private keys. Ensure that the backup is stored in a secure location.

2.4.2.2 Data Restoration and Migration

This topic explains how to restore data in FlowForce Server. It also provides information about data migration, which allows copying FlowForce Server data from a previous [instance-data directory](#)¹⁸ to the current one. If necessary, it also upgrades the FlowForce database to the latest version. The [migratedb](#)³⁹⁷ command, which is used to migrate data, can be invoked to copy application data from one folder to another. Running this

command may be useful when you want to migrate FlowForce Server to a new machine or when you need to restore the application data directory from a backup.

If you only need to upgrade the FlowForce database version to the latest one, it is sufficient to run [upgradedb](#)⁴⁰³.

Data restoration

If the `INSTANCEDIR_BACKUP` (which is the copy of the `INSTANCEDIR`⁷⁶) is of the same version and on the same machine as the currently running FlowForce Server, you can restore data as follows:

1. If FlowForce Server services are running, stop them.
2. Rename the `INSTANCEDIR`, for example, to `temp_data`.
3. Copy the `INSTANCEDIR_BACKUP` to `INSTANCEDIR`.
4. Start both the FlowForce Web Server and FlowForce Server services.

You can also restore backups that originate from another machine and perhaps have an older database version. The steps below could be useful, for example, if you want to migrate FlowForce data to a new server, or if a hardware failure has occurred.

Note that you can restore data on a machine that runs the same or a different operating system. In the latter case, note that all the paths used in jobs may not be valid on the new operating system, in which case they will need to be updated manually. Importantly, credentials that are tied to operating system user accounts, that is, credentials where the *Allow usage for job execution* option is enabled, may no longer be valid on a new machine, in which case they will need to be updated manually.

To restore data to a new FlowForce Server installation or version, follow the instructions below:

1. Install FlowForce Server and any of the following, if applicable: MapForce Server, StyleVision Server and RaptorXML Server. If you need to install LicenseServer as well, you can select it as part of the FlowForce Server installation (Windows only). On other platforms, you will need to install LicenseServer separately.
2. Log on to the LicenseServer Web administration interface and deregister all the products from the old machine. Then register all the products from the new machine with LicenseServer. This step can also be performed after migration.
3. If FlowForce Server services are running, stop them.
4. Rename the `INSTANCEDIR`, for example, to `temp_data`.
5. Run the `migratedb` command by supplying `INSTANCEDIR` as `--datadir` and `INSTANCEDIR_BACKUP` as `--olddatadir` (see examples below).

Windows

```
FlowForceServer migratedb
  --datadir=C:\ProgramData\Altova\FlowForceServer2024\data
  --olddatadir=C:\transfer\backup_data
```

CentOS

```
sudo ./flowforceserver migratedb
  --datadir=/var/opt/Altova/FlowForceServer2024/data
  --olddatadir=/home/chang/backups/data
```

6. Start (in this order) the FlowForce Server and FlowForce Web Server services.


Data migration

This subsection provides information about data migration on Windows, Linux, and macOS.

Windows

On Windows, you do not typically need to migrate configuration data manually. When you install a new major version of FlowForce Server, and a previous major version is already installed, the installation wizard prompts you to migrate the configuration data.

Should you need to migrate configuration data manually, follow the instructions below:

1. Ensure that Altova ServiceController  is running in the system notification area. Otherwise, [start](#) ²⁷ the Altova ServiceController.
2. [Stop](#) ⁶¹ the FlowForce Server service and the FlowForce Web Server service.
3. Delete the FlowForce Server data folder installed by the 2024 installation wizard.
4. At the command prompt, run the FlowForce executable with the [migratedb](#) ³⁹⁷ command, for example:

```
"C:\Program Files(x86)\Altova\FlowForceServer2024\bin\FlowForceServer.exe" migratedb
--datadir=C:\ProgramData\Altova\FlowForceServer2024\data
--olddatadir=C:\ProgramData\Altova\FlowForceServer2022\data
```

5. [Start](#) ⁶¹ the FlowForce Server Web and the FlowForce Server services.

Linux

Before migrating data on Linux, take the following steps:

1. [Uninstall](#) ³¹ the previous version of FlowForce Server. Note that deinstallation does not remove the application data directory. For more information, see [Important Paths](#) ¹⁸. The path to the application data directory depends on the major version of FlowForce Server (for example, `/var/opt/FlowForceServer2022`).
2. [Install](#) ³¹ FlowForce Server 2024. This creates a new application data directory with the default configuration data (for example, `/var/opt/FlowForceServer2024`).

To migrate data to FlowForce Server 2024, follow the instructions below:

1. Stop the FlowForce Web Server service if it is running:

```
sudo systemctl stop flowforcewebserver
```

2. Stop the FlowForce Server service if it is running. Use the same command as above but replace `flowforcewebserver` with `flowforceserver`.
3. Remove or rename the NEW data directory created during the installation:

```
sudo rm -rf /var/opt/Altova/FlowForceServer2024/data
```

4. Migrate the EXISTING data by running the [migratedb](#) ³⁹⁷ command available in the command-line interface of FlowForce Server. For example:

```
sudo /opt/Altova/FlowForceServer2024/bin/flowforceserver migratedb
--olddatadir=/var/opt/Altova/FlowForceServer2022/data
```

```
--datadir=/var/opt/Altova/FlowForceServer2024/data
```

5. Start the FlowForce Web Server service:

```
sudo systemctl start flowforcewebserver
```

6. Start the FlowForce Server service. Use the same command as above but replace `flowforcewebserver` with `flowforceserver`.

macOS

Note the following prerequisites:

- FlowForce Server 2024 must be installed (see [Installation on macOS](#)³⁸).
- Perform data migration as a user with administrative (root) privileges.

To migrate data to FlowForce Server 2024, follow the instructions below:

1. Stop the FlowForce Server service:

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

2. Stop the FlowForce Web Server service:

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

3. Remove or rename the data directory that was created during the installation:

```
sudo rm -rf /var/Altova/FlowForceServer2024/data
```

4. Run the [migratedb](#)³⁹⁷ command:

```
sudo /usr/local/Altova/FlowForceServer2024/bin/FlowForceServer migratedb
--olddatadir=/var/Altova/FlowForceServer2022/data
--datadir=/var/Altova/FlowForceServer2024/data
```

5. Start the FlowForce Server service:

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

6. Start the FlowForce Web Server service:

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

2.4.3 Localize FlowForce Server

FlowForce Server is delivered with support for the following languages: English, French, German, Spanish, and Japanese. To set any of these languages as the default language, use FlowForce Server's [setdeflang](#)⁴⁰⁰ command. To create a localized version of FlowForce Server in a different language, follow the instructions below:

1. Generate an XML file containing the resource strings, by using the [exportresourcestrings](#)³⁹² command.
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 braces, 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 [INSTALLDIR](#)¹⁸\bin folder. Your DLL file will have a name in the following format: `FlowForceServer2024__lc.dll`. The `_lc` part of the name contains the language code. For example, in `FlowForceServer2024_de.dll`, the `de` part is the language code for German (Deutsch).
5. Run the [setdeflang](#)⁴⁰⁰ 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.

3 Web UI Reference

The FlowForce Server Web administration interface allows you to administer the server and configure jobs.

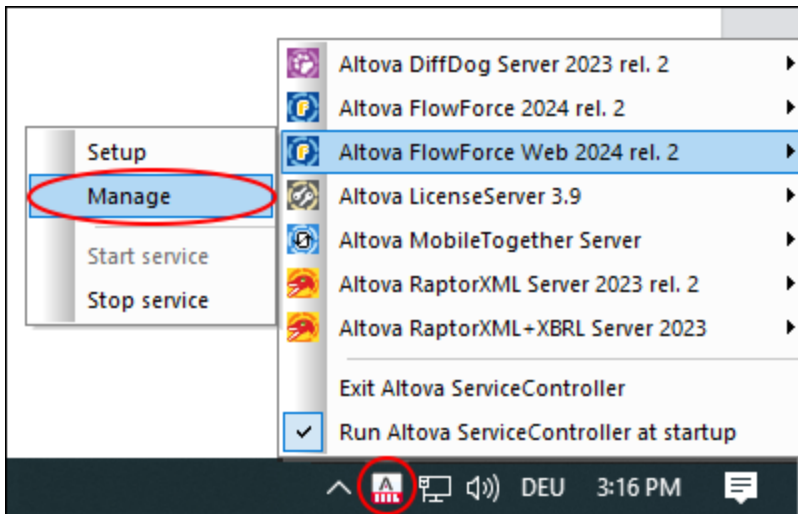
Log on to FlowForce Server

To manage FlowForce Server (create jobs, add users, etc.), you need to log on to the Web Administration Interface. Before logging in, you must make sure that both LicenseServer and FlowForce Server are running.

Accessing the logon page

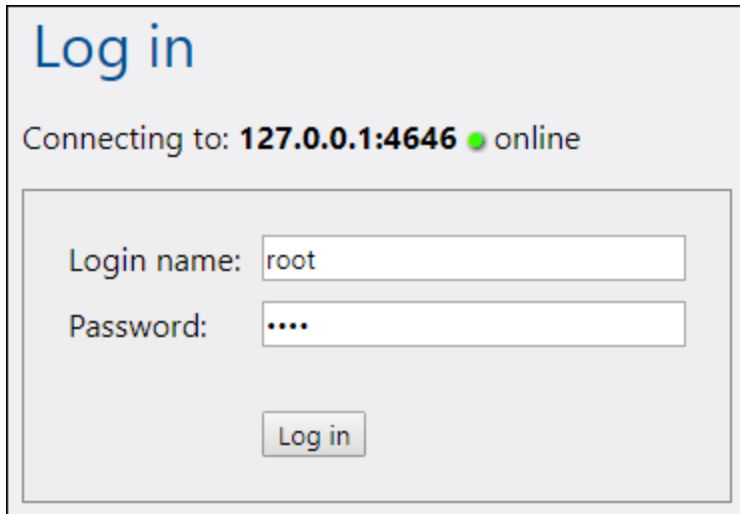
You can access the Web Administration interface at the [configured HTTP\(S\) address and port](#)⁴⁷ (e.g., `http://localhost:8082`) or through Altova ServiceController. ServiceController is an application that enables you to conveniently start, stop, and configure Altova services on Windows systems.

To access the logon page through Altova ServiceController, click the **ServiceController** icon in the system tray, hover over **Altova FlowForce Web** in the menu that pops up (*screenshot below*), and then select **Manage** from the FlowForce Web submenu. Clicking **Manage** opens the logon page where you can enter your credentials (*see subsections below for details*).



Logging in with default username/password

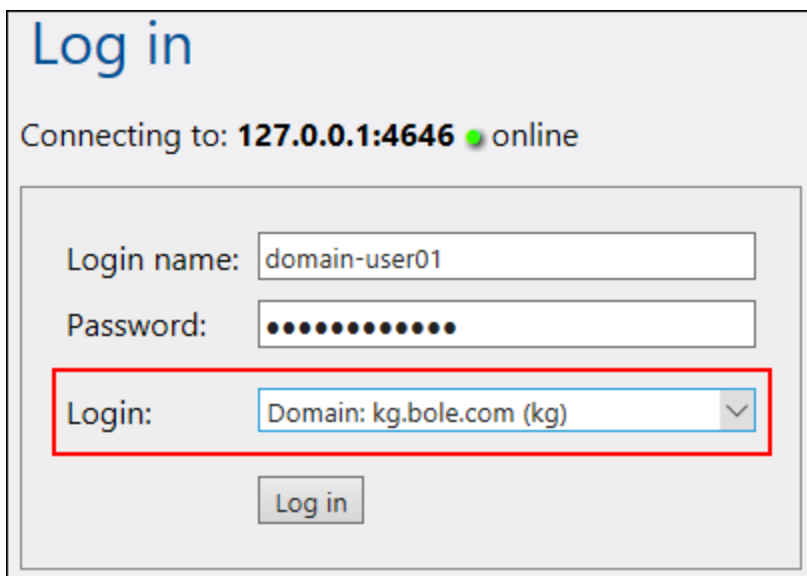
By default, after a fresh installation of FlowForce Server, you can log on with the username `root` and the password `root` (*screenshot below*). For security reasons, make sure to change the default root password immediately after the first login to FlowForce Server.



The screenshot shows a login page titled "Log in". Below the title, it says "Connecting to: 127.0.0.1:4646 online" with a green dot. The login form contains two input fields: "Login name:" with the text "root" and "Password:" with four dots. Below these fields is a "Log in" button.

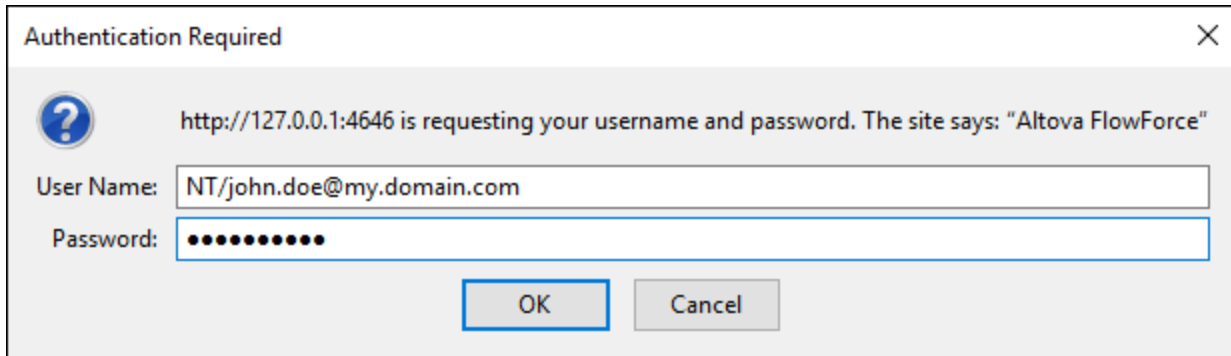
Logging in as a domain user

If [authentication with a Directory Service provider](#)¹¹⁹ (such as Active Directory) has been configured, domain users can also log on to FlowForce. In this case, the login page includes an additional drop-down list in which you can select a domain (*screenshot below*). To use standard HTTP authentication instead of Directory Service authentication, select *Directly* from the *Login* drop-down list.



The screenshot shows a login page titled "Log in". Below the title, it says "Connecting to: 127.0.0.1:4646 online" with a green dot. The login form contains three input fields: "Login name:" with the text "domain-user01", "Password:" with ten dots, and "Login:" with a dropdown menu showing "Domain: kg.bole.com (kg)". A red box highlights the "Login:" field. Below these fields is a "Log in" button.

Clients that access Web services exposed by FlowForce Server (typically, at a URL like `http://localhost:4646/service/SomeService`) may also use Active Directory authentication as an alternative to HTTP authentication. For Active Directory authentication to be possible, the username must be prefixed with `NT/` and must include the domain name, for example, `NT/john.doe@my.domain.com` (*screenshot below*). See also [Jobs as Web Services](#)¹⁵⁸.



Overview of Web UI

The following pages are available in the Web administration interface:

- Home
- Configuration
- Log
- Administration
- Help

Note: Access to resources and actions available from the Web administration interface is driven by a user access control mechanism. This means that you can access and modify configuration data as long as your assigned permissions allow it. Similarly, you can perform actions (and see the corresponding menu items) if you have been granted the corresponding privilege.

Note about browsers

We recommend using FlowForce Server with the following browsers: Chrome, Edge, and Firefox. If you are experiencing problems with your browser, try updating it to the latest version.

Home

As soon as you log on to the Web UI, you will see the Home page. This page displays the latest [statistics and charts](#)⁸⁵, the list of [running jobs](#)⁸⁵, and the list of [active timers](#)⁸⁷.

Configuration

The Configuration page displays the currently defined FlowForce containers, jobs, credentials, and functions. To view the contents of and further information about any object, click the corresponding record.

The following containers are available by default:

- /public
- /system
- /RaptorXML (if you have licensed RaptorXML Server)

For more information about containers, see [Overview of Containers](#)⁹⁵. From the Configuration page, you can also manage containers, jobs, credentials, and functions, and set permissions on containers if you have the relevant access rights.

Log

Opens the [Log View page](#)¹⁰⁴ that shows log entries, including server-related and job-related messages.

Administration

The Administration page enables you to perform actions related to server configuration and user management.

The Administration page consists of the following tabs:

- *Users*: Enables you to [set up and manage user accounts](#)¹⁰⁹.
- *Roles*: Enables you to create, delete, and manage roles. For more information, see [Roles](#)¹¹².
- *Password Policies*: Enables you to define [password-complexity rules](#)¹¹⁶.
- *Reports*: Enables you to view reports on currently assigned user privileges.
- *Settings*: Enables you to define the default time zone, mail server, and settings that let you integrate FlowForce Server with Active Directory or an LDAP-compliant server. For more information, see [Settings](#)¹¹⁸.
- *Cluster*: Enables you to distribute execution of jobs across multiple instances of FlowForce Server (*Advanced Edition*).

Note: Cross-system clusters are not supported, which means that a worker-master connection cannot be established between different OS platforms (e.g., between Linux and Windows).

Help

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

3.1 Home

The Home page is the main page that provides information about executed and running jobs, their outcome and statuses. If multiple FlowForce Server instances are configured to run as a cluster, you can also monitor the cluster members (*Advanced Edition*).

In this section

The section is organized into the following topics:

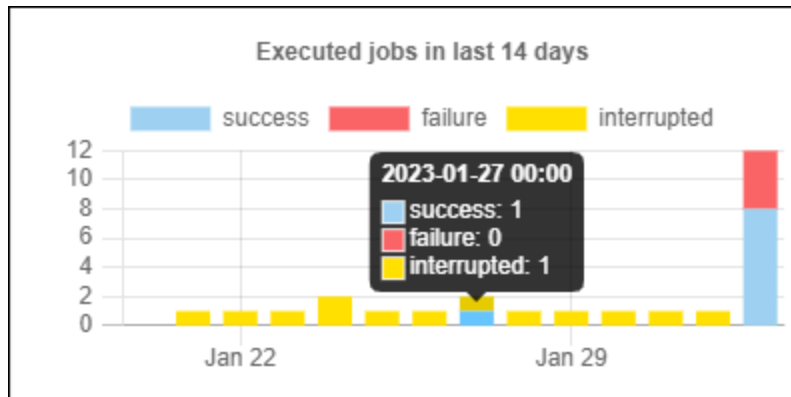
- [Job Info on Home Page](#) ⁸⁵
- [Job Statuses](#) ⁸⁹

3.1.1 Job Info on Home Page

This topic describes job monitoring data available on the **Home** page. The **Home** page has the following sections: *Statistics (Advanced Edition)*, *Running Jobs* and *Active Timers (see below)*.

Statistics (Advanced Edition)



The *Statistics* section of the **Home** page displays jobs executed in the last 14 days, 24 hours, and 60 minutes. Each of the charts contains bars colored according to the job execution result: *success*, *failure*, and *interrupted*. When you move the mouse over a specific bar on the chart, a tooltip appears with detailed information about the respective time period. For example, in the chart below, the tooltip indicates that one job instance was executed successfully and one job instance was interrupted on January the 27th at 00:00.



To find out more about a particular piece of information on a chart, double-click the bar of interest in any of the charts. This displays the **Log** page, with the log pre-filtered for the given minute, hour, or day. There may be slight differences between the statistics displayed in charts and the exact log details tracked by the log. To see a more detailed statistical report, click the link **Show more statistics** located under the first chart.

Running Jobs

The *Running Jobs* section displays up to 10 currently running jobs (*see screenshot below*).

Running Jobs				
Instance ID	Job	Activation Time	Last Action 	Status
 Show all jobs (1) , including 1 recently finished , 0 starting and 0 running jobs				

The *Running Jobs* section contains the following columns:

- *Instance ID*: When a job instance starts, a unique ID is assigned to it. The instance ID helps you track the execution status of each job instance on the **Log** page. You can click the instance ID inside the table. This redirects you to the **Log** page where you can view the details of the selected job instance. If you would like to use the job's instance ID in a job (e.g., to create unique file names), this is possible with the help of the [instance-id](#)²³² expression function.
- *Job*: This column shows the path where you can see the configuration of this job instance.
- *Activation Time*: Indicates the date and time when the job instance started running.
- *Last Action*: The date and time of the last execution status.
- *Status*: The job status as it was when the page was last refreshed. To find out more about job instance statuses, see [Job Statuses](#)⁸⁹.

Stop jobs

You can stop any currently running job if your user account (or any roles that your user account is a member of) has the [Stop-any-job](#)⁷² privilege. Stopping jobs that are still running may cause data corruption and should be done only exceptionally. To stop a running job, take the following steps:




1. Click **Home**. Any currently running jobs are displayed in the *Running Jobs* section.
2. Click **Stop job**. FlowForce Server will ask whether you want to stop the running instance. Click **OK**.

Stopping the running instance may take several minutes depending on the job type. During this time interval, the job status changes to *Aborting* or *Aborting after step N*. As soon as the job instance stops running, the status changes to *Aborted* or *Aborted after step N*. If the job instance still cannot be stopped, click **Force stop job** to stop it forcefully.

All jobs

When you click **Show all jobs** in the *Running Jobs* section, a new page called **Recent and Running Jobs** opens (see *screenshot below*). The table on this page displays all the running and any recently finished jobs, including jobs that failed. Such jobs are displayed only for a short time (approximately 1-2 minutes) after their execution has finished. You can always check the full history of each job instance on the **Log** page. For more information, see [Log](#)¹⁰⁴. The **Recent and Running Jobs** page is not refreshed automatically. To get the latest

status of all jobs, click the  button (**Reload Grid**).


Recent and Running Jobs					
Instance ID	Job	Activation Time	Last Action	Status	
17	 /public/Examples/MyTask	2021-02-02 17:18:00	2021-02-02 17:18:29	Finished successfully after step 1	
18	 /public/Examples/MyTask	2021-02-02 17:19:00	2021-02-02 17:19:29	Finished successfully after step 1	
19	 /public/Examples/MyTask	2021-02-02 17:20:00	2021-02-02 17:20:00	Running step 1	Stop job

[Show all recent and running jobs](#)

If multiple FlowForce Server instances run as a cluster, the grid includes additional details about the cluster members running each job instance (*Advanced Edition*).

Recently finished jobs

To see finished jobs, click **recently finished** in the *Running Jobs* section (see screenshot below). Jobs remain in this list for 90 seconds.

Home	Configuration	Log	Administration	Help
Recently Finished Jobs (show all Recent and Running Jobs)				
Instance ID	Job	Activation Time	Last Action	Status
3	 /public/Examples/HelloWorld	2021-09-01 15:35:58	2021-09-01 15:35:58	Aborted

Starting jobs


To see jobs that are about to run, click **starting**.

Running jobs

To see currently running jobs, click **running**.

Active timers

The *Active Timers* section (see screenshot below) displays up to 10 jobs scheduled to run via [timer triggers](#) ¹⁵²

Active Timers		
Next run ↕	Job	Info
2021-08-25 10:29:00	 /public/Test/Test1	Fire (as in Europe/Berlin) starting 2021-08-25 10:29:00.

[Show all 2 active triggers and services](#)

Show active triggers and services

To view the full list of active triggers and services, click **Show all active triggers and services** (see *screenshot above*). This opens the **Active Triggers and Services** page that displays the table with the following columns:

- *Type*: Indicates [the type of trigger](#)¹⁵¹. The watch trigger refers to a file system trigger or an HTTP trigger. The *Info* column provides additional details about the job (see *details below*).
- *Job*: Specifies the path of the job where the trigger or the service is defined. Click the link to open the job's configuration page.
- *Next run*: Applies to watch triggers only. This column indicates when the trigger will run next.
- *Info*: Provides additional information about jobs running as Web services. For watch and timer triggers, this column summarizes the current configuration of the trigger.
- *Service URL*: Specifies the URL where the Web service is accessible. This applies only to jobs [running as Web services](#)¹⁵⁸.

Additional information about file triggers


In the *Info* column on the **Active Triggers and Services** page, file triggers might have additional information, as shown in the screenshot below (*red rectangle*).

Info
Checking directory 'C:\Test\Test.txt' for content change starting 2022-04-29 14:21:00 ending 2022-04-29 14:25:00. Total files watched: 1. New files: 0, currently examining: 0, waiting for settle period: 0.
Fire (as in Europe/Berlin) every Mon, Tue, Wed, Thu, Fri, Sat and Sun starting 2022-02-10 18:05:00.
Service URL: http://127.0.0.1:4646/service/HelloWorld

Additional information about file triggers may contain the following messages:



- *Total files watched* indicates the number of files seen in the directory at the last scan.
- *New files* shows the number of files that did not exist before the last scan and have not been checked yet.
- *Currently examining* shows the number of files checked. New files have priority with regard to checking. Then old (already known) files will be checked.
- *Waiting for settle period* displays the number of files that have been checked and found changed, but these files are waiting for settle time (settle time has been configured to be not zero).

If a trigger encounters an error, this error might be shown as a third line in red (e.g., *Error: path must be absolute*). Triggers with an error have a red background for the whole row.

The **Active Triggers and Services** table is not refreshed automatically. Click the  button (**Reload Grid**) to refresh the page.

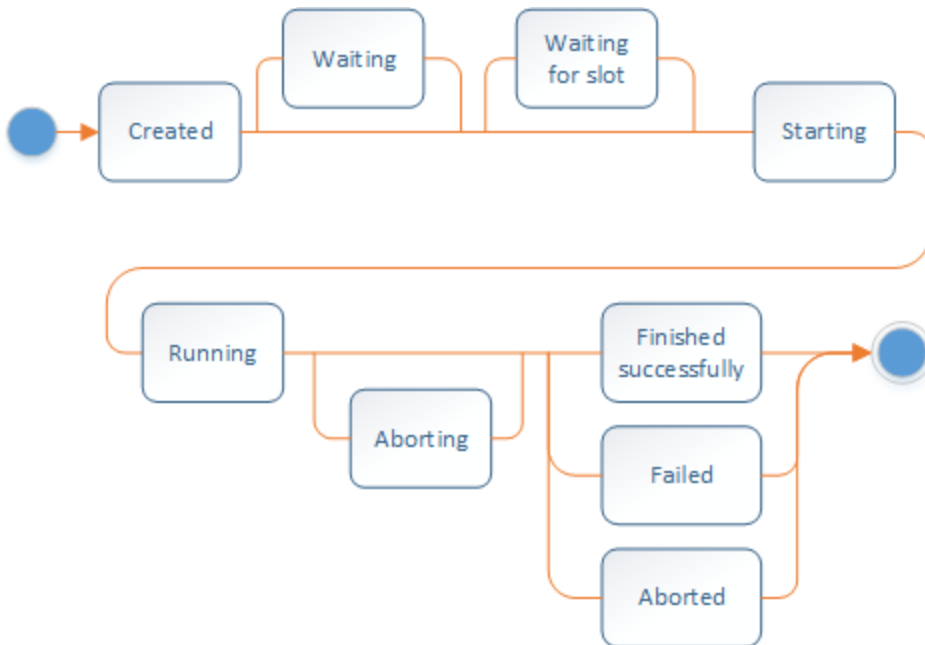
3.1.2 Job Statuses

Across its lifetime, a job instance has various statuses, as shown in the *Status* column in the **Running Jobs** table below. To find out more about job statuses, see the subsection below.

Running Jobs					
Instance ID	Job	Activation Time	Last Action	Status	
29	 /public/MyTask	2019-02-05 12:26:00	2019-02-05 12:26:29	Finished successfully after step 2	
30	 /public/MyTask	2019-02-05 12:27:00	2019-02-05 12:27:00	Running step 2	Stop job

Job instance stages

The diagram below illustrates how a job instance changes from one state to another across its lifetime. It is assumed that no loss of FlowForce Server service or network interruptions have occurred. Note that some of the statuses take a very short time span and will not normally be visible in the user interface.



Job statuses can be divided broadly into two types: *created* and *finished*. Each of these types is further divided into different statuses (see below).

Created

The *Created* status is the first state the job is in before any other action takes place. This status is abstract (i.e. it cannot be entered) and cannot be observed. The *Created* status is a superset of the following statuses: *Starting*, *Waiting*, *Waiting for slot*, *Running*, and *Aborting* (see details below).

☐ Starting

If the execution queue has an opening and the instance is not delayed for some reason, it proceeds to the *Starting* status. The *Starting* status has a short time frame and lasts while the instance starts up. Then the job instance usually switches to the *Running* status.

☐ Waiting

If the instance is delayed, it receives the *Waiting* status.

☐ Waiting for slot

If the job instance is ready to run, but the execution queue is currently full, this job instance switches to the *Waiting for slot* status. An execution queue has a limited number of slots. Therefore, only the specified number of job instances can be executed in parallel in the same queue. For details, see [Queue Settings](#)¹⁶⁶. Any further instances arriving for that queue will wait until a slot becomes available.

☐ Running

Indicates that the job instance is currently running and will stay in this state until the execution is complete or until some external event occurs that ends the execution prematurely. Except for a very brief time window at the beginning, this status has a step number associated with it. Therefore, the instance gets the *Running step {step}* status. Job instances can also have the following statuses: *Running postponed steps* and *Running postponed step {step}*. To find out more about postponed steps, see [Postponed Steps](#)¹⁴⁰.

☐ Aborting

A job instance switches to this status when the user cancels a job. It might take FlowForce Server some time to process the request. The *Aborting* status acknowledges the receipt of this request. Note that the job instance may actually be able to complete successfully before it switches to the *Aborted* state. If this happens, the job will be reported as having finished successfully. If the previous status had a step number, the *Aborting after step {step}* status would be shown instead of *Aborting*.

Finished

The *Finished* status is abstract (i.e. it cannot be entered) and includes the following statuses: *Finished successfully*, *Failed*, *Aborted*, *Interrupted*, *Superseded*, *Lost connection*, *Synchronizing*, *Untracked*, *Recovering* (see details below).

☐ Finished successfully

This is a final state which indicates that the job has finished successfully. The status *Finished successfully after step {step}* additionally indicates that the successful completion is associated with a particular step number.

Failed

The execution of the job instance has failed. This is a final status and there will be no further attempts to run the job instance. The *Failed after step {step}* status additionally indicates that the failure is associated with a step number.

Aborted

This status indicates that a user has stopped the job although it may also happen indirectly after an unexpected shutdown. This is a final state that indicates that at least some part of the job has not finished. If the previous status had a step number, the *Aborted after step {step}* status would be shown instead of *Aborted*.

Interrupted

The execution of the job instance has been interrupted. This is a more forceful variation of the *Aborted* state. The job instance cannot be restarted. Therefore, it should be treated as failed. To avoid data inconsistency, it is recommended to check the outcome manually.

Superseded

This status means that the job instance has not executed anything and that some other instance might have run instead of it. This status can only appear before the *Starting* status when, for example, the `triggerfile` has changed again during the settle time specified by the [Wait N seconds for settle](#)¹⁵⁵ option. The *Superseded* status is not a critical condition.

Lost connection (*Advanced Edition*)

This status applies when multiple FlowForce instances run as a cluster. This status indicates that the master machine has lost connection to the worker machine. When the connection is lost, FlowForce Server does not know whether the instance is still running. When the worker connection is reestablished, the instance switches to the *Synchronizing* status.

Synchronizing (*Advanced Edition*)

This status applies when multiple FlowForce instances run as a cluster. In a clustered setup, the master machine gets the current progress of job instances from the worker machines. When the worker connection is reestablished, the instance starts synchronizing and FlowForce is trying to get the latest status from the worker.

Untracked

An instance gets the *Untracked* status when FlowForce crashes or is terminated while the instance is still running. You have to abort such a job manually by clicking the **Stop job** button in the *Running Jobs* list; otherwise, it stays in that list until the next FlowForce service restarts. When you stop an untracked job, it goes to the *Aborted* state and remains in the list for some time (currently about 90 seconds).

Recovering

When an instance has become untracked, FlowForce Server will switch on the *Recovering* state before the job instance can proceed.

3.2 Configuration

The Configuration page enables you to create, modify and manage jobs, configure job-related settings and permissions.

3.2.1 Permissions and Containers

This section includes the following topics:

- [How Permissions Work](#) ⁹²
- [Understanding Containers](#) ⁹⁵
- [Creating, Renaming, and Moving Containers](#) ⁹⁷
- [Viewing Container Permissions](#) ⁹⁹
- [Changing Container Permissions](#) ¹⁰⁰
- [Restricting Access to the /public Container](#) ¹⁰²

3.2.1.1 How Permissions Work

Permissions are access rights and can be set for each container individually. Permissions determine which users or roles have access to that container and what kind of access each user/role has (read, write, use, no access). Permissions can be defined for containers, configuration objects, credentials, queues, services, functions, resources, and child containers. In FlowForce Server Advanced Edition, permissions can also be set for certificates and AS2 partner objects.

FlowForce checks container permissions when users interact with containers. For example, users can view or change the contents of a container only if they have been granted the required permissions. Permissions are not evaluated upon job execution; therefore, any permission changes will not apply retroactively to existing jobs.

For each FlowForce Server container, you can set the following permission types.

Container

The "Container" permissions define what users can do with objects in the current container.

Inherit	Provides to the user the same access rights to this container as those defined on the parent container.
Read	Grants the user rights to list the contents of the container.
Read, Write	Grants the user rights to list the contents of the container and to create or delete objects in the container. Note: To successfully create a new configuration object, or delete an existing one, users must be granted both the Container - Read, Write permission and the Configuration - Read, Write permission.

No access	Denies the user the right to enter the container (more specifically, the container appears to the user as disabled).
------------------	--

Configuration

The "Configuration" permissions define what a user can do with configuration objects (namely, jobs and credentials) in the current container.

Inherit	Provides to the user the same configuration object–related rights as those defined on the parent container.
Read	Grants the user rights to view details about configuration objects within the container (such as the execution steps or triggers of a job).
Read, Write	Grants the user rights to modify any configuration object within the container (for example, edit the trigger of a job). Note: To successfully create a new configuration object, or delete an existing one, users must be granted both the Container - Read, Write permission and the Configuration - Read, Write permission.
No access	Denies the user the right to view the details of any configuration objects within the container (more specifically, configuration objects appear to the user as disabled).

Credential

This permission defines what a user can do with [Credentials](#)¹⁶² defined in this container.

Inherit	Provides to the user the same credential–related rights as those defined on the parent container.
Use	Grants the user rights to reuse any credentials defined in this container.
No access	Denies the user the right to reuse credentials defined in this container.


Queue

This permission defines what a user can do with queues defined in this container.

Inherit	Provides to the user the same queue rights as those defined on the parent container.
Use	Grants the user rights to assign a job to any queue defined in this container.
No access	Denies the user the right to assign a job to queues defined in this container.

Service

The "Service" permission defines access to a job exposed as a Web service, via the HTTP request interface.

Inherit	Provides to the user the same service–related rights as those defined on the parent container.
Use	<p>Grants the user rights to access the service and thus execute the job via the request interface.</p> <div style="border: 1px solid black; background-color: #e6f2ff; padding: 5px;"> <p>Notes</p> <ul style="list-style-type: none"> • Service permission checks skip any container hierarchy checks. Therefore, if granted Use permission, users may use the service without having Read access to the container in which the corresponding job is defined. • If you grant Use permission to user  anonymous, the service becomes publicly available and does not require authentication. </div>
No access	Denies the user the right to access the job as a Web service.

Function

In addition to jobs, credentials, and other configuration data, a container may contain functions. These include built-in FlowForce functions, RaptorXML functions, and MapForce mappings or StyleVision transformations deployed to FlowForce.

When a FlowForce user creates a job, some execution step in their job may refer to functions from the same container, or from a different one. The "Function" permission defines whether users can invoke (refer to) functions from the container where the permission is defined.

For example, let's assume that an administrator has deployed various MapForce mappings to a FlowForce container called "Restricted". The administrator can then decide if users should be able to refer to functions in this container, by changing the "Function" permission. More specifically, any user or role who has the **Function - Use** permission on container "Restricted" can refer to functions from this container (i.e., select them from a drop-down list when they create an execution step). On the contrary, users or roles with the **Function - No Access** permission will not be able to select any function from the "Restricted" container.

If an administrator revokes users' access to functions after they had already used the function in a job, those users won't be able to run the job any longer. The job configuration page displays in this case a message with the text "You don't have permission to use the selected function".

Inherit	Provides to the user the same function–related rights as those defined on the parent container.
Use	Grants the user rights to call (refer to) any function defined inside the container.
No access	Denies the user rights to call (refer to) any function defined inside the container.

Resources

This permission defines what a user can do with [Resources](#)⁴²³ defined in this container.

Inherit	Provides to the user the same resource-related rights as those defined on the parent container.
----------------	---

Use	Grants the user rights to reuse (refer to) any resources defined in this container.
No access	Denies the user the right to reuse (refer to) any resources defined in this container.

Security

The security permission controls access to permissions of any child containers defined in the current container.

By default, users are permitted to read only permissions applicable to them (that is, any permissions assigned to themselves or any role they are a member of). However, users who have the *Read users and roles* privilege can read all permission entries.

Inherit	Provides to the user the same security–related rights as those defined on the parent container.
Read Security	Grants the user rights to view the permissions of any child of the container.
Read and Write Security	Grants the user rights to change the permissions of any child of the container.
No access	Denies the user rights to view the permissions of any child of the container.

3.2.1.2 Overview of Containers

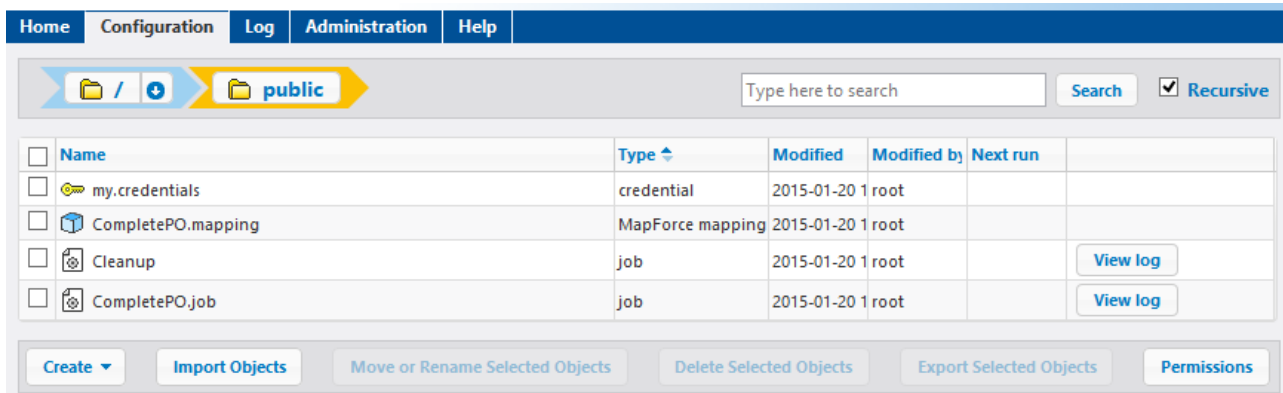
FlowForce Server manages jobs, credentials, step functions, and other configuration objects in a hierarchical structure of containers. A container is similar to a folder on an operating system. Containers can have any of the following: jobs, credentials, functions, and other containers. By setting permissions on a container, you can control who can access the container's contents.

The top-level container in FlowForce Server is the root (/) container. By default, the root container contains the following predefined FlowForce Server containers.

/public	<p>The /public container is the default location where any FlowForce user can create jobs and credentials. It is by default empty and accessible to any FlowForce user. The /public container serves as default location in the following cases:</p> <ul style="list-style-type: none"> • When you deploy mappings from MapForce to FlowForce Server. • When you deploy transformations from StyleVision to FlowForce Server. <p>You can, however, deploy mappings or transformations to a different container, if required.</p>
/RaptorXML	<p>This container is present if you licensed RaptorXML Server. It stores the validation and other functions specific to RaptorXML Server.</p>

/system	The system container contains the FlowForce Server system functions. It is not recommended to make changes to this container.
----------------	--

You can navigate through containers from the Web administration interface, by clicking on a container to view its contents. The following screen shot shows a sample **/public** container that contains several configuration objects.








Sample FlowForce container

To go back to any container in the hierarchy, use the breadcrumb-style navigation available at the top of the page.

You can also search objects either within the current container including children objects (if the *Recursive* check box is checked) or only within the current container (if the *Recursive* check box is unchecked).

Containers contain objects such as jobs, deployed MapForce mappings or StyleVision transformations, functions, credentials. When you open a container, the following information is available about its objects:

Property	Description
<i>Name</i>	Specifies the name of the object on the file system. Note that, when you create a new object, the name must not be already in use.
<i>Type</i>	<p>Specifies the object type (such as credential, job, or function). You can also identify the object type by its accompanying icon:</p> <ul style="list-style-type: none">  Credential  Function (includes built-in functions, MapForce mappings and StyleVision transformations)  Job  Container  Missing configuration object. You may see this icon when you attempt to import into FlowForce Server data that has unresolved dependencies, see Handling

Property	Description
	Missing Dependencies ²⁹² .
<i>Date modified</i>	Specifies the date and time when the object was created or last modified.
<i>Modified by</i>	Specifies the name of the user who modified the object.
<i>Next run</i>	For jobs scheduled to run with time triggers, this column specifies the date and time of the next run, as defined in the job settings.
<i>View log</i>	For jobs, this button provides quick access to the execution log of the corresponding job.

Provided you have [permissions](#)⁹² to do so, you can create any number of additional containers to store your custom FlowForce server data (for example, one for each department). Alternatively, you can store data in the **/public** container, which by default is available to any authenticated user. If necessary, it is possible to restrict access to the **/public** container (see [Restricting Access to the /public Container](#)¹⁰²).

You can also move, rename, and delete any containers where you have the relevant permissions.

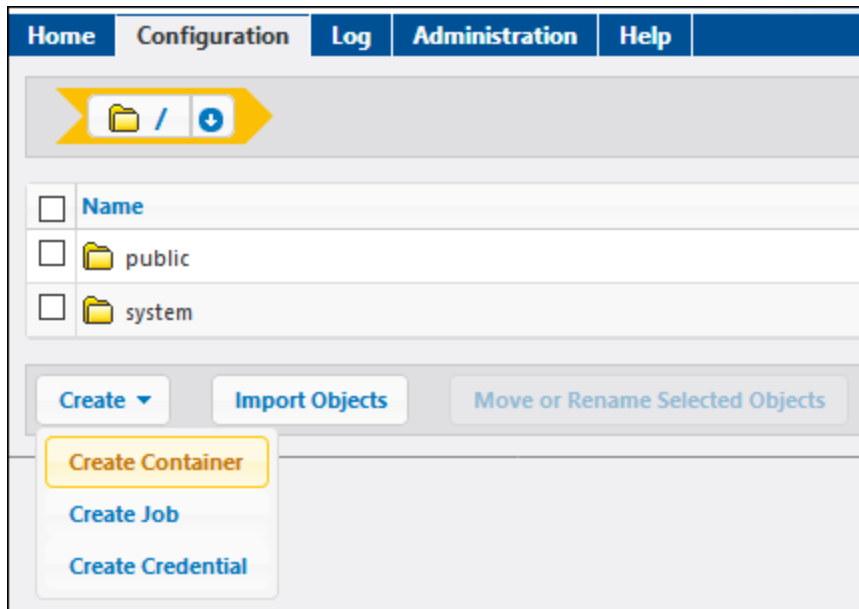
3.2.1.3 Create/Rename/Move Containers

You can create, rename and move containers if you (or any roles you are member of) have the *Container / Read, Write* permission (see also [How Permissions Work](#)⁹²).

Note: It is not recommended to modify the contents of the **/RaptorXML** and **/system** containers, which are provided by FlowForce Server by default.

To create a container:

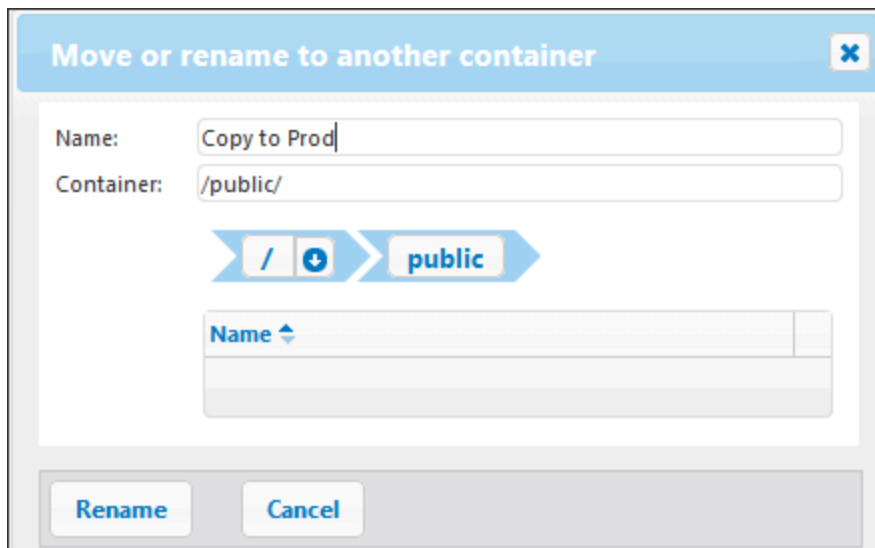
1. Click **Configuration**.
2. Click an existing container under which you want to create a new container. If you want to create the container at the top level of the hierarchy, omit this step.
3. Click the **Create Container** button located in the lower left part of the page.



4. Enter the name of the container. The following name restrictions apply:
 - It must not be empty
 - It must not begin or end with space characters
 - It can contain letters, digits, single space, underscore (_), dash (-), and full stop (.) characters.
5. Click **Save**.

To rename a container:

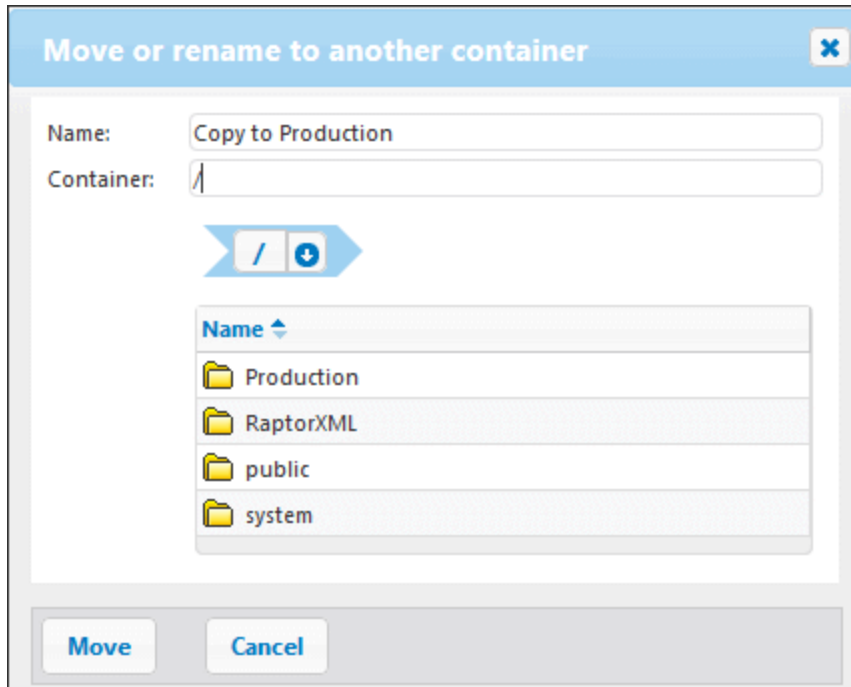
1. Click **Configuration**, and then navigate to the container you want to rename.
2. Select the check box next to the container, and click **Move or Rename Selected Object**.



3. Enter the name of the container in the Name box, and then click **Rename**.

To move a container:

1. Click **Configuration**, and then navigate to the container you want to move.
2. Click the **Move or Rename Selected Objects** button located in the lower left part of the page.



3. Select the container's destination by doing one of the following:
 - o Enter the path in the Container text box.
 - o Use the interactive navigation controls to reach the destination container.
4. Optionally, set the new name of the container by typing it in the Name box.
5. Click **Move**.

To move multiple containers:

- Click the check boxes next to them, and then follow the same logic as for moving a single container.

To select or deselect all objects in the container:

- Click the topmost check box.

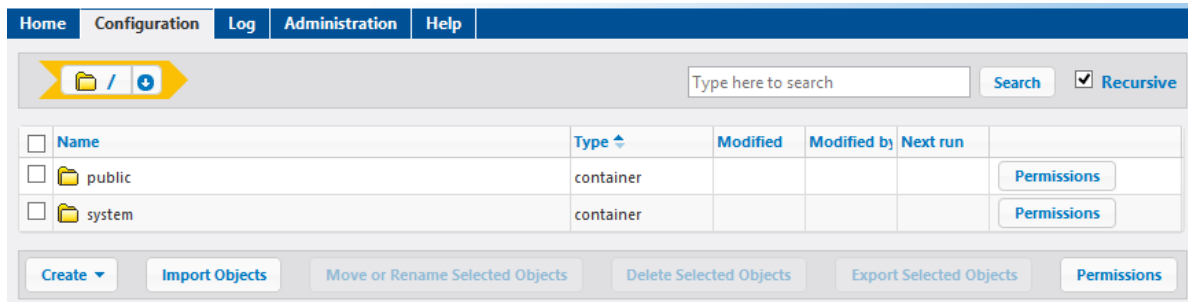
3.2.1.4 Container Permissions

You can view the permissions of containers where you have the relevant permissions to do so (see also [How Permissions Work](#)⁹²). By default, you can see your own permissions with respect to the container. If you are member of any role, you can also see the permissions available to roles of which you are member. If you have

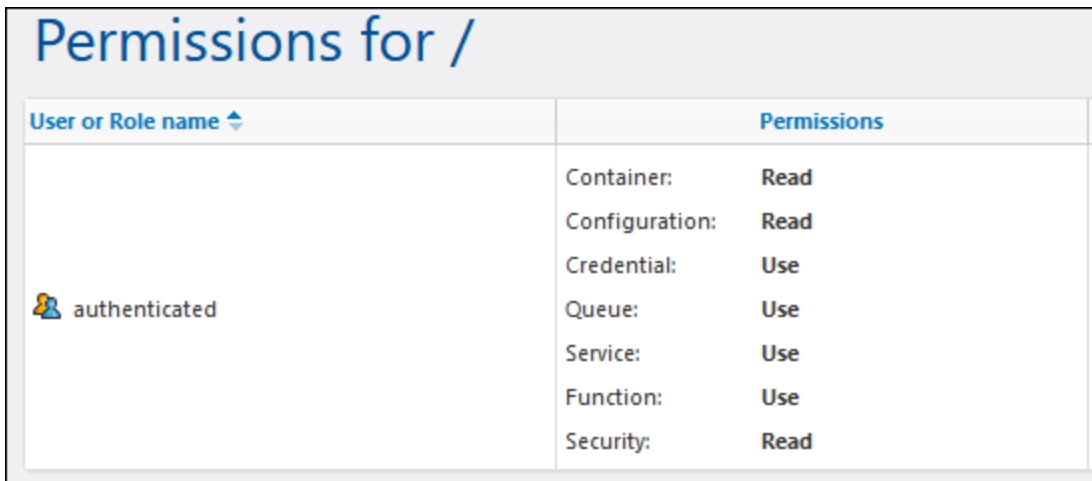
the privilege *Read users and roles*, you can also see the permission of other users and roles with respect to the container.

To view the permissions of a container:

1. Click **Configuration**.
2. Do one of the following:
 - o Click the **Permissions** button adjacent to the container record.
 - o Enter the container, and then click the **Permissions** button available in the lower right corner of the page.



The *User and Role name* column displays any users and roles whose permissions you have rights to see. The *Permissions* column displays what permission types are available to this particular user or role with respect to the container. For example, the image below illustrates the default permissions available to role **authenticated** for the root (/) container.



For the description of each permission type, see [How Permissions Work](#)⁹².

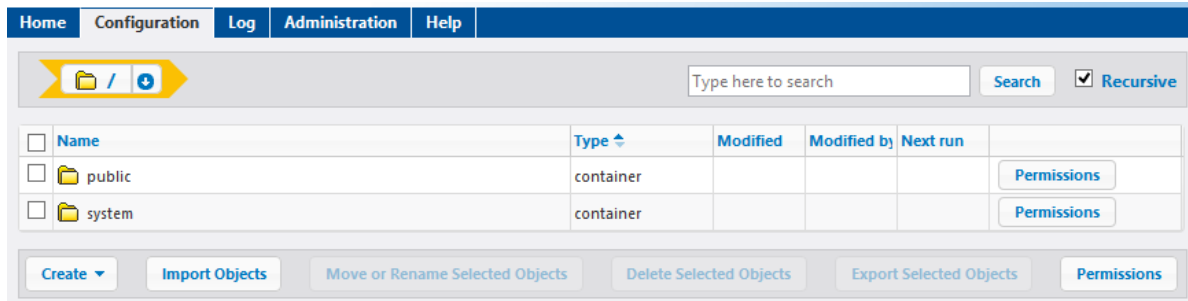
3.2.1.5 Setting Container Permissions

You can change permissions of containers where the following is true:

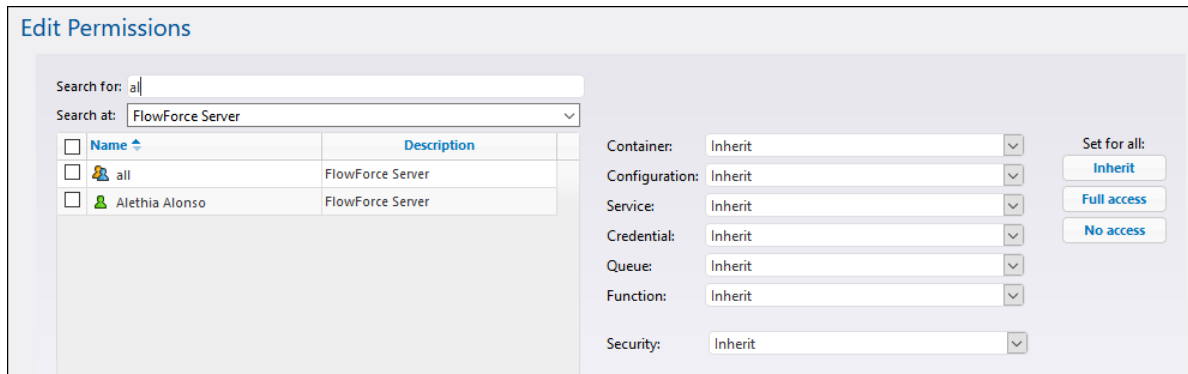
- You (or any roles you are member of) have the *Security / Read and Write Security* permission on the parent container relative to the one where you want to change permissions. For example, to change the permission of container "Jobs" which is a child of container "Marketing", you must have the permission *Security / Read and Write Security* on container "Marketing" (see [How Permissions Work](#)⁷²).
- You (or any roles you are member of) have been granted the privilege *Override Security* (see [How Privileges Work](#)⁷²).

To change the permissions of a container:

1. Click **Configuration**.
2. Do one of the following:
 - Click the **Permissions** button adjacent to the container record.
 - Enter the container, and then click the **Permissions** button available in the lower right corner of the page.













3. Do one of the following:
 - To change the permissions of any of the listed users and roles, click the **Change** button next to the relevant user or role.
 - To add permissions for any users and roles that are not listed, click **Add Permissions**.
4. In the **Edit Permissions** section, search for the user or role whose permissions you want to change, and select the check box next to it. You can either search for users created in FlowForce Server, or, if Directory Service is enabled, for domain users. For more information about importing domain users into FlowForce Server, see [Users](#)¹⁰⁹.




5. Change each relevant group of permissions as required. For the description of each permission type, see [How Permissions Work](#)⁹². If you want to modify all permission types with a single click, use the **Inherit**, **Full access**, and **No access** buttons.
6. Click **Save Changes**.

3.2.1.6 Restrict Access to the /public Container


The **/public** container (located under the top-level root container) is available by default in FlowForce Server. It acts as a location accessible to any FlowForce Server user and a location where any FlowForce Server user can store their data, without any predefined permissions. Therefore, by default, the **/public** container has the following permissions.

Permissions for /public			
User or Role name	Permissions		
 authenticated	Container:	Read, Write	Change
	Configuration:	Read, Write	
	Credential:	Use	
	Queue:	Use	
	Service:	Use	
	Function:	Use	
	Security:	Read inherited from  /	
 root	Container:	Read, Write inherited from  authenticated	Change
	Configuration:	Read, Write inherited from  authenticated	
	Credential:	Use inherited from  authenticated	
	Queue:	Use inherited from  authenticated	
	Service:	Use inherited from  authenticated	
	Function:	Use inherited from  authenticated	
	Security:	Read, Write inherited from  /	

Default permissions of the /public container


This means that, by default, any FlowForce Server user who is member of the  **authenticated** role can do the following:

- Add, modify, and delete objects inside the **/public** container (namely, jobs, credentials, or other containers)
- Reuse any credentials available in the **/public** container
- Access as a Web service any job located in the **/public** container, provided that the job was configured to be available as a Web service
- Refer to any function available in the **/public** container
- Read the permissions assigned to the **/public** container

Note: These permissions may also be inherited by any containers that are children of the **/public** container. Normally, any new container inherits the permissions of the parent container; however, permissions may have been overridden by the  **root** user, or by other users with relevant privileges.

You can restrict access to the **/public** container, if required. Note, however, that the [job configuration examples](#)²⁹⁴ included in this documentation assume the existence of the /public container.

To restrict access to the /public container:

1. Revoke permissions on this container from the  **authenticated** role (see [Setting Container Permissions](#)¹⁰⁰).
2. Create a new role and assign this role to all users who require permissions to the **/public** container (see [Roles](#)¹¹²).
3. Assign to the new role only the required permissions (again, see [Setting Container Permissions](#)¹⁰⁰).

3.3 Log

The log provides information about all system activities and job-related events.

3.3.1 Job Info in Log

You can view details about all kinds of events on the Log View page. To access the Log View page, click the **Log** menu. You can also access the log from other locations where the **View log** button is displayed (e.g., on each job's configuration page).

Note: By default, you can view the log of any jobs where you have read-only access. To view the global log of all jobs and events in FlowForce Server, your user account must have the [View unfiltered log](#)⁷² privilege.

Log view

Navigate to:

Date from:

Date to:

Filter by:

Object path:

Instance ID:

Minimum severity:

[Show](#) [Reset Filter](#)

Date	Severity	User	Instance ID	Message
<div style="display: flex; justify-content: space-between; margin-bottom: 5px;"> ⬆ Show 25 more records ⬆ Show 100 more records ⬆ Show 500 more records </div>				
2023-03-15 13:27:07	INFO	root	463	Finished job execution: job /public/Examples/CheckTemperature in queue /public/Examples/CheckTemperature
2023-03-15 13:27:07	INFO	root	463	Starting execution of job /public/Examples/CheckTemperature in queue /public/Examples/CheckTemperature
2023-03-15 13:27:07	INFO			Service /CheckTemperature of /public/Examples/CheckTemperature invoked
2023-03-15 13:26:47	INFO	root	462	Finished job execution: job /public/Examples/ListDirContents in queue /public/Examples/ListDirContents
2023-03-15 13:26:47	INFO	root	462	Starting execution of job /public/Examples/ListDirContents in queue /public/Examples/ListDirContents
2023-03-15 13:26:47	INFO			Service /ListDir.txt of /public/Examples/ListDirContents invoked
2023-03-15 13:26:41	INFO	root		Configuration for /public/Examples/ListDirContents changed.
2023-03-15 13:25:50	ERROR	root	461	Job execution failed: job /public/Examples/ListDirContents in queue /public/Examples/ListDirContents
2023-03-15 13:25:50	INFO	root	461	Starting execution of job /public/Examples/ListDirContents in queue /public/Examples/ListDirContents
2023-03-15 13:25:49	INFO			Service /ListDir.txt of /public/Examples/ListDirContents invoked
2023-03-15 13:25:44	ERROR	root	460	Job execution failed: job /public/Examples/ListDirContents in queue /public/Examples/ListDirContents
2023-03-15 13:25:44	INFO	root	460	Starting execution of job /public/Examples/ListDirContents in queue /public/Examples/ListDirContents
2023-03-15 13:25:43	INFO			Service /ListDir.txt of /public/Examples/ListDirContents invoked
2023-03-15 13:25:41	INFO	root	459	Finished job execution: job /public/Examples/HelloWorld in queue /public/Examples/HelloWorld
2023-03-15 13:25:41	INFO	root	459	Starting execution of job /public/Examples/HelloWorld in queue /public/Examples/HelloWorld
2023-03-15 13:25:41	INFO			Service /HelloWorld of /public/Examples/HelloWorld invoked
2023-03-15 11:48:32	INFO	ffserver		FlowForce scheduling engine startup complete.
2023-03-15 11:48:32	INFO	ffserver		Starting up FlowForce scheduling engine...
2023-03-15 11:48:32	INFO	flowforce		Read FlowForce configuration from:
2023-03-15 11:46:19	INFO	ffserver		All jobs have been terminated.
2023-03-15 11:46:19	INFO	ffserver		Beginning termination of running jobs.
2023-03-15 11:46:19	INFO	ffserver		FlowForce scheduling engine stop complete.
2023-03-15 11:46:19	INFO	ffserver		Stopping FlowForce scheduling engine...
2023-03-14 20:24:55	INFO	ffserver		FlowForce scheduling engine startup complete.
2023-03-14 20:24:55	INFO	ffserver		Starting up FlowForce scheduling engine...

⬇ Show 25 more records
⬇ Show 100 more records
⬇ Show 500 more records

[Export](#)
[View Exported Log or Log for Instance...](#)
[Copy Permalink to Clipboard](#)

The subsections below describe filter options, the main parts of the log view table, and export/copy options.

Filters

You can filter log entries using the following criteria:

- *Date from*: Includes only events after this date.
- *Date to*: Includes only events before this date. If you set both the *Date from* and *Date to* filters, up to 1000 records within that range will be shown. To view additional records, click the **Show N more records** buttons. The most recent records are always shown first.
- *Object path*: Shows events configured at the selected path. You can select the path to some specific FlowForce object (e.g., a job or credential record).
- *Instance ID*: This option is useful when you want to see all the log entries related to one specific instance ID.
- *Minimum severity*: This option helps filter log entries based on severity. The severity statuses are explained below.

After changing any filters, click the **Show** button or the **Enter** key to apply the filters. The **Reset Filter** button clears all filters and refreshes the log. Clicking the **Show** button without any filters set also refreshes the log.

About minimum severity

The following severity statuses are available: *Verbose*, *Info*, *Warning*, and *Error*. The *Info* status is the default severity type.

The *Verbose* status can be useful for troubleshooting [file system triggers](#)¹⁵⁴. When you select the *Verbose* status, you will get detailed information about the job: e.g., the start and end of scanning of the directory and so on. To be able to use this status, you must specify the following parameter in [the .ini file](#)⁶⁷:

```
[Experimental]
trigger.verbose = 1
```

When you select the *Verbose* status, the log will show the following severity types: *Verbose*, *Info*, *Warning*, and *Error*. The *Info* status includes information messages, warnings, and errors. When you select *Warning*, only warnings and errors will be shown in the log. If you are interested in the most critical messages, select *Error*.

Log table

The log view table has the following columns: *Date*, *Severity*, *User*, *Instance ID*, and *Message*.

- *Date*: Indicates the date and time of an event.
- *Severity*: Indicates the severity of an event. You can filter messages by severity (see above). The default severity type is *Info*.
- *User*: This can be a FlowForce service, a Python security service, or a specific FlowForce user.
- *Instance ID*: Each run of a job produces a unique job instance whose ID is shown in the *Instance ID* column. To find out more about a specific instance, click the link displayed in the *Instance ID* column. For details, see [Instance Log](#)¹⁰⁶. Note that some logged events do not have an ID, because it is not applicable (e.g., events related to changes in job configuration).
- *Message*: Provides information about each log entry. Note that some log entries may be truncated, because the default maximum length of a log entry has been exceeded. To change the length of log entries, see [Logging Settings](#)¹²².

To load older records, click **Show N more records**. To resize any column in the grid, click any of the vertical bars delimiting the column headings, and, holding the left mouse button pressed, drag to the left or right.

Export, copy, and view log details

To export the log to a file on the disk, click **Export**. All records that are currently visible on the page will be exported as a JSON file.

To view the previously exported log entries (.json file) or [logged instance](#)¹⁰⁷ (a .zip archive or a .json file extracted from that .zip archive), click the **View Exported Log or Log for Instance** button. When you view the exported log or instance log, some links in the records (e.g., instance ID links, links to jobs) might lead to an error page, because these links are not in the exported file.

The **Copy Permalink to Clipboard** button copies the current URL of the log view to the clipboard, including any selected parameters (e.g., `?id=2773968&limit=25`). This is useful if you want to quickly load the same information later onto the page. For example, you can paste the permanent URL into another browser's address bar or send it to someone else so that they can see the same log.

3.3.2 Instance Log

The Log for Instance N page (*screenshot below*) provides detailed information about a specific job instance. You can open this page in one of the following ways:

- Click the instance ID link of a record on the [Log View](#)¹⁰⁴ page.
- Click the instance ID link of a record in the [Recent and Running](#)⁸⁵ section of the Home page.
- Click **View Log** from the job configuration page to go to the [Log View](#)¹⁰⁴ page. Then click the instance ID link of a record.

Log for instance 159

Created instance: 2022-06-28 18:05:00
 Finished instance: 2022-06-28 18:05:01
 Job Name: [/public/Examples/ValidateSchema](#)
 Queue Name: [/public/Examples/ValidateSchema](#)
 Number of steps executed: 1
 Postponed Steps Failures: 0
 Current State: Finished successfully

[Expand all steps](#) [Collapse all steps](#)

Date	Message
2022-06-28 18:05:00	Starting instance 159.
2022-06-28 18:05:00	Starting execution of job /public/Examples/ValidateSchema in queue /public/Examples/ValidateSchema
2022-06-28 18:05:00	Running instance 159 locally.
2022-06-28 18:05:00	<ul style="list-style-type: none"> Execute main job /public/Examples/ValidateSchema <ul style="list-style-type: none"> Execute function /RaptorXMLBRL/valany <ul style="list-style-type: none"> Execute function /RaptorXMLBRL/2023/valany
2022-06-28 18:05:01	Finished job execution: job /public/Examples/ValidateSchema in queue /public/Examples/ValidateSchema

[Export](#) [View Exported Log or Log for Instance...](#)

Reported data

The instance log can report the following categories of data:

- Messages related to the execution of job instances, grouped by step. These include:
 - Messages related to the execution of built-in functions and mappings
 - Results of steps that run the [compute](#)²³⁷ and [compute-string](#)²³⁹ functions
 - Error messages that lead to *retry* in the *Execute with success/failure* handler step or to job failure
- Information about elapsed time after step execution.
- Iterations of for-each steps.
- Information about how many times the job has been retried. For details, see [Retry on Error](#)¹³⁷.
- Information about streams produced by executing mappings or by the [commandline](#)²⁸⁵ function.

Export and view log entries

The **Export** button creates a `.zip` archive of all data associated with the current log instance.

To view the previously exported [main log](#)¹⁰⁴ (`.json` file) or logged instance (a `.zip` archive or a `.json` file extracted from that `.zip` archive), click the **View Exported Log or Log for Instance** button. When you view

the exported log or instance log, some links in the records (e.g., instance ID links, links to jobs) might lead to an error page, because these links are not in the exported file. When you have finished viewing the instance log loaded from the file, click **Close Exported Log View**.

Expand/collapse all steps

FlowForce Server allows you to show or hide information about an instance, which can help you get detailed information about this instance or only a general overview of the instance, respectively (see *red rectangle in screenshot above*).

3.4 Administration



The Administration page enables you to create and manage users, roles, and password policies. You can also get an overview of privileges, configure various settings, and manage clusters (*Advanced Edition*).

3.4.1 Users


This topic explains how to create and import users and reset the root password. This topic also provides information about default users, domains and domain trusts.

Default users

The following special users are predefined in FlowForce Server.

 root	This user is the initial, top-level FlowForce Server administrator. By default, it has all permissions and privileges available in the system.
 anonymous	This 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 as Web services (see Exposing Jobs as Web Services ¹⁵⁸).

The built-in users cannot be deleted, although it is possible to change their privileges.

Note: The `root` user can change any privileges and permissions, including own permissions and privileges. Take extra caution when logged in as  **root** and editing root privileges, since you may unintentionally lose your own access to the system. In the event that this happens, see [Resetting the Root Password](#).

To get a global view of all currently assigned privileges, use privilege reports.

In addition to creating FlowForce Server users, you can import domain user accounts and roles from Windows Active Directory or an LDAP Directory Service provider. When the *Allow any domain users to log in* setting is enabled in the [Directory Service settings](#)¹¹⁹, users from configured domains are able to log on to FlowForce Server even if you have not explicitly imported their accounts into the FlowForce Server database. To ensure that domain users log on to FlowForce Server only if their account has been explicitly imported by an administrator, clear the *Allow any domain users to log in* check box and import the domain users, as shown below.

Note: The local machine accounts are not part of Active Directory. Therefore, they cannot be imported into FlowForce Server.

Create users

A user is a person who logs on to FlowForce Server to create and monitor jobs, deploy MapForce mappings and StyleVision transformations, and configure various settings. The scope of actions available to users in FlowForce Server depends on the following:

- The permissions and privileges assigned to the users
- The permissions and privileges assigned to the roles that the users are members of

To add a FlowForce Server user:

1. Click **Administration**, and then click **Users**.
2. Click **Create User**.
3. Fill in the required fields.

<i>User name</i>	Enter the name of the user. The following restrictions apply: <ul style="list-style-type: none"> • It must not be empty • It must not begin with or end with spaces • The allowed characters are letters, digits, underscore (_), dash (-), and full stop (.)
<i>Password</i>	Enter the user's password.
<i>Re-type password</i>	Re-type the user's password.
<i>Change password on next login</i>	If you select this check box, the user will be prompted to change password on next login.

4. Optionally, grant the required [privileges](#)¹¹² to the user. Note that you can grant privileges to users either directly from this page, or by assigning to them a role which already has some privileges. To simplify user maintenance, it is recommended to use the latter approach (see *Create Roles* below and [Assign Roles to Users](#)¹¹³).
5. Click **Save**.


To rename a user:

1. Click **Administration**, and then click **Users**.
2. Click the user record you want to edit.
3. Enter the new name in the **User name** text box, and then click **Save**.


Notes:

- When a user name is changed, the currently assigned user password remains unchanged.
- If you are changing your own name (provided that you have this privilege), the changed name becomes effective as soon as you click **Save**, and is visible in the top right area of the page.

Reset root password

In the event that you forgot or lost the password of the  **root** user account, you can reset it to the default value from the command line interface (see the command [resetpassword](#)³⁹⁹).

To perform root password reset, it is assumed that you have access to the operating system where FlowForce is running, including FlowForce binaries and data files. This is the same kind of access required when installing FlowForce or when migrating to a new FlowForce version or server manually.

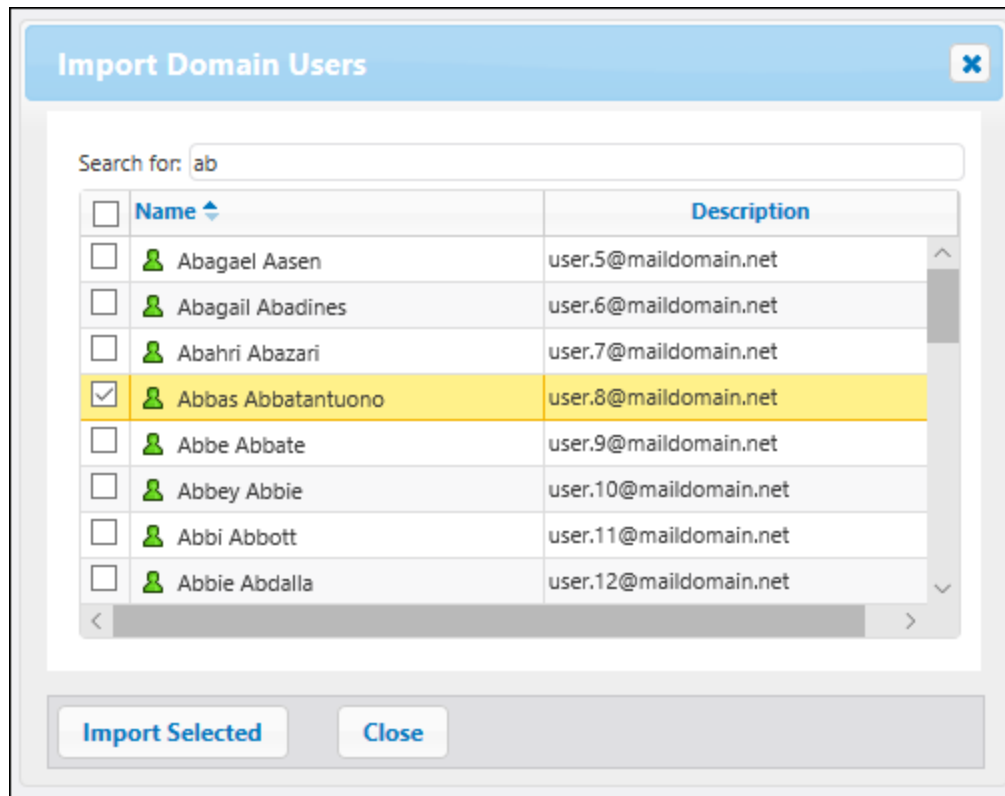
When you perform a password reset, the privileges of the  **root** user will also be restored to the default value (that is, all the privileges will be granted).

Performing a root password reset does not affect any FlowForce users except the  **root** user.

Import domain users

To import domain user accounts into FlowForce Server, take the following steps:

1. Open the Settings tab of the Administration page, select the *Enable* check box in the *Directory Services* section, and configure your preferred Directory Service provider, as described in [Changing the Directory Service Settings](#) ¹¹⁹.
2. Open the Users tab on the Administration page.
3. Click **Import Domain Users**, which opens the **Import Domain Users** dialog shown below.



4. If applicable, select the domain of choice from the **Context** drop-down list.
5. In the *Search for* text box, start typing the name of the user account you want to import. Partial searches are valid: For example, if you enter a value such as *ad*, the accounts *Administrators*, *Admanager*, and *Admin* are retrieved from the LDAP server or Active Directory and shown in the webpage dialog. In the case of Active Directory, FlowForce Server uses the Ambiguous Name Resolution (ANR) search algorithm that allows you to specify complex search conditions in a single clause. For example, you can retrieve the account of a person named Jim Smith by typing *ji sm*. See the Microsoft documentation for further information about Ambiguous Name Resolution in Active Directory.
6. Select the records that you you want to import and click **Import Selected**. Waiting time increases if the domain is not local.

Domains and domain trusts

You can see the list of available domains on the login page and in the following sections of the Administration page: (i) in the dialog box **Import Domain Users** in the **Users** tab, (ii) in the dialog box **Import Domain Roles** in the **Roles** tab, and (iii) in the **Settings** tab. Currently, only the following domains are visible in FlowForce Server: the domain with the machine on which FlowForce Server is installed and any domains from the same

[forest](#) to which this machine belongs. However, other trusted domains connected via the [external. forest. realm and shortcut trusts](#) are not supported and cannot be seen in the list of available domains in FlowForce Server.

Note: To run a job, you can use any user credentials accepted by Windows. In this case, Windows will take care of the external trusts.

3.4.2 Roles





This topic explains how to create, import, and assign roles.





A role defines a set of privileges and permissions. It can be assigned to another role or to a user. A role's privileges automatically become the privileges of any other role or any user that the role is assigned to. A user can be assigned any number of roles. As a result, a user will have all the privileges defined in the multiple assigned roles.

Note that privileges are global, whereas permissions are defined per container.

Default Roles

The following special roles are predefined in FlowForce Server.

 authenticated	This role includes all users who are authenticated using an existing user name and password. Every FlowForce Server user except user  anonymous is a member of this role. By default, this role has the <i>Set own password</i> privilege.
 all	This role includes all FlowForce Server users, including user  anonymous . By default, this role has no privileges.

Since the roles  **authenticated** or  **all** are built-in, you cannot explicitly assign these roles to users or revoke them from users. The membership of the built-in roles is automatically managed by FlowForce Server. Every time when you add a new user, FlowForce Server automatically assigns to the new user both the role  **authenticated** and the role  **all**.

Create roles



To add a FlowForce Server role:

1. Click **Administration**, and then click **Roles**.
2. Click **Create Role**.
3. Enter the role name (for example, "Administrator").
4. Under **Privileges**, select the privileges that must be assigned to the role (for the description of available privileges, see [Privileges](#)¹¹⁴).
5. Click **Save**.

To rename a role:

1. Click **Administration**, and then click **Roles**.
2. Click the record you want to edit.
3. Enter the new role name in the **Role name** text box, and then click **Save**.

Notes

- The members of a role do not change when the role is renamed.
- The default roles all and authenticated cannot be changed.





Import domain roles








To import domain roles into FlowForce Server, take the following steps:

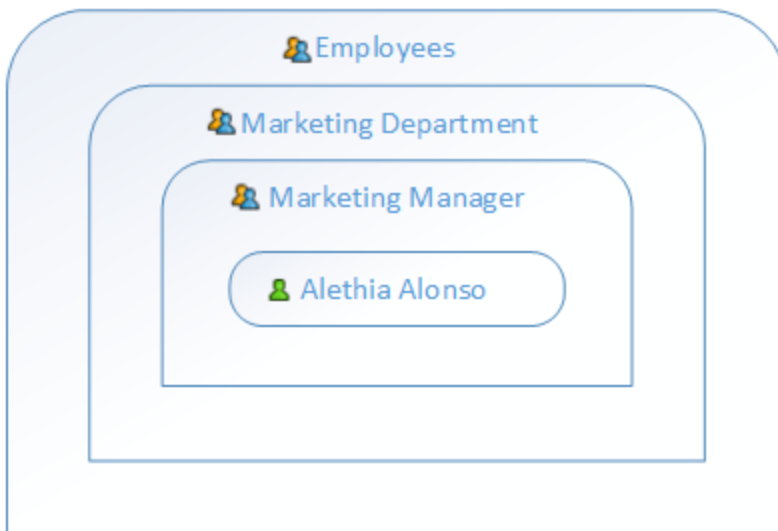
1. Click **Roles** in the **Administration** menu.
2. Click **Import Domain Roles**.
3. Follow the steps 4-6 above.

Assign roles to users and roles

You can assign privileges directly to a user (e.g., Alethia Alonso) or to a particular role (e.g., Marketing Manager). It is recommended to assign privileges to roles rather than to individual users, because it simplifies the maintenance and management of privileges in the long term.

You can model the hierarchy of your organization in FlowForce Server, by assigning roles to other roles. The diagram below illustrates a sample organization, for which three roles and one user have been defined. The Employees role contains a role called Marketing Department. This means that the privileges and permissions granted to the Employees role will automatically be inherited by the users belonging to the Marketing Department role.

The Marketing Department role contains the Marketing Manager role. In this case, the Marketing Manager role will inherit all the privileges from the Marketing Department and Employees roles. A user called Alethia Alonso is the marketing manager, and she has been assigned the Marketing Manager role. This implies that she will inherit all the privileges from the broader roles.

**Assign roles to users**

To assign one or more roles to a user:

1. Click **Administration**, and then click **Users**.
2. In the list of users, click the record you want to edit.
3. Under **Roles available**, select the roles that must be assigned to the user, and then click **Assign**.

To revoke one or more roles from a user:

1. Click **Administration**, and then click **Users**.
2. In the list of users, click the record you want to edit.
3. Under **Roles assigned to user '<user name>'**, select the roles that must be revoked from the user, and then click **Remove**.

To assign a role to multiple users:



1. Click **Administration**, and then click **Roles**.
2. In the list of roles, click the record you want to edit.
3. Under **Users/Roles available**, select the users that must be assigned the role, and then click **Assign**.

To revoke a role from multiple users:

1. Click **Administration**, and then click **Roles**.
2. In the list of roles, click the record you want to edit.
3. Under **Members of role '<role name>'**, select the users from whom the role must be revoked, and then click **Remove**.

Assign roles to other roles

To assign a role to another role:

1. Click **Administration**, and then click **Roles**.
2. In the list of roles, click the role you want to assign to another role (for example, if you want the role  **Marketing Department** to inherit privileges from the role  **Employees**, click "Employees").
3. Under **Users/Roles available**, select the role to be assigned, and then click **Assign**.

3.4.3 Reports

As a FlowForce Server administrator, you might find it difficult to keep track of privileges assigned to each and every role or user, especially when the number of users and roles increases. To help you get a quick overview of all privileges currently assigned to users and roles, FlowForce Server provides the following reports:
















































- Privileges Report
- Privileges by User Report

To view these reports, open the Reports tab of the Administration page.

Privileges Report

This report lists the FlowForce Server privileges. For each privilege, you can see the users who have been granted that privilege or inherited it by virtue of their roles.





































Privileges Report

Maintain global settings	 root	granted to  root
Maintain users, roles and privileges	 root	granted to  root
Override security	 root	granted to  root
Read users and roles	 Alethia Alonso	inherited from  Manager
	 Manager	granted to  Manager
	 root	granted to  root
Set own password	 Alethia Alonso	inherited from  all  authenticated
	 Employee	granted to  Employee
	 Klaus Mauer	inherited from  all  authenticated  Employee
	 Natsuo Shinohara	inherited from  all  authenticated  Employee
	 all	granted to  all
	 anonymous	inherited from  all
	 authenticated	granted to  authenticated
	 root	granted to  root and inherited from  all  authenticated
Stop any job	 Alethia Alonso	inherited from  Manager
	 Manager	granted to  Manager
	 root	granted to  root
View unfiltered log	 Alethia Alonso	inherited from  Manager
	 Manager	granted to  Manager
	 root	granted to  root

Privileges by User Report

This report lists the FlowForce Server users. For each user, you can see the currently assigned privileges, and whether they have been granted or inherited.

Privileges by User Report

 Alethia Alonso	Read users and roles	inherited from  Manager
	Set own password	inherited from  all ,  authenticated
	Stop any job	inherited from  Manager
	View unfiltered log	inherited from  Manager
 Employee	Set own password	granted to  Employee
 Klaus Mauer	Set own password	inherited from  all ,  authenticated ,  Employee
 Manager	Read users and roles	granted to  Manager
	Stop any job	granted to  Manager
	View unfiltered log	granted to  Manager
 Natsuo Shinohara	Set own password	inherited from  all ,  authenticated ,  Employee
 all	Set own password	granted to  all
 anonymous	Set own password	inherited from  all
 authenticated	Set own password	granted to  authenticated
 root	Maintain global settings	granted to  root
	Maintain users, roles and privileges	granted to  root
	Override security	granted to  root
	Read users and roles	granted to  root
	Set own password	granted to  root and inherited from  all ,  authenticated
	Stop any job	granted to  root
	View unfiltered log	granted to  root

3.4.4 Password Policies

A password policy defines a set of minimum requirements that a user password must meet in order to be valid (e.g., a password must be at least *N* characters long). FlowForce Servers uses password policies to enable administrators to enforce the complexity of user passwords.

The password complexity rules that you can define within a password policy are as follows:

- The total minimum length of the password (that is, the password must be at least *N* characters long to be valid)
- The minimum number of letters that the password must contain
- The minimum number of digits that the password must contain

You can define as many password policies as required (provided that you have the *Maintain users, roles and privilege* privilege). Once you define password policies, you can assign them to FlowForce users. A user account can have one password policy at a time.

When the user requests a password change, the system checks if the new password meets the complexity requirements defined in the user's password policy. If the password does not meet the complexity requirements defined in the password policy, the password change is denied, and the system displays a relevant message.

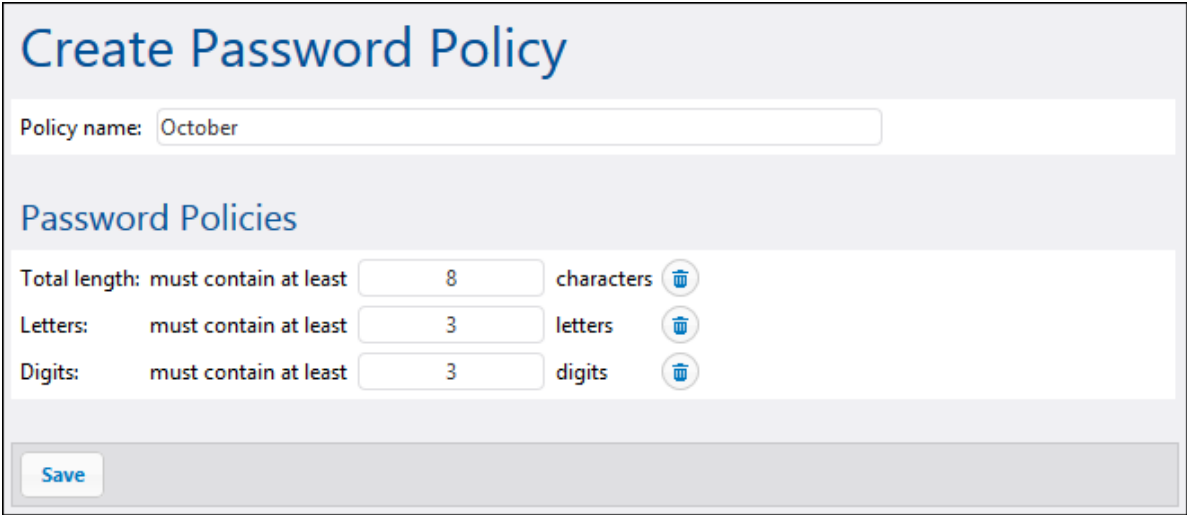
When an administrator changes the password of a user, FlowForce Server does not enforce the password policy. Also, if the password policy changes, any existing passwords remain unaffected. In the latter case, the password policy will be enforced when users attempt to change the existing password.

By default, FlowForce Server includes an empty password policy which does not enforce any password complexity rules. FlowForce Server implicitly assigns the default password policy to any user account that does not have a custom password policy. The default password policy cannot be changed.

Create and assign password policies

To create a new password policy:




1. Click **Administration**, and then click **Password Policies**.
2. Click **Create Policy**.



Create Password Policy

Policy name:

Password Policies

Total length:	must contain at least	<input type="text" value="8"/>	characters	
Letters:	must contain at least	<input type="text" value="3"/>	letters	
Digits:	must contain at least	<input type="text" value="3"/>	digits	

3. Enter the required password policy rules, and then click **Save**. The list of current users becomes available under the defined policy.
4. Click to select the user records that must be assigned the new policy, and then click **Assign**.

Members

Users available

<input checked="" type="checkbox"/>	Name ↕	Current Policy
<input checked="" type="checkbox"/>	Default for new use	➡ null
<input checked="" type="checkbox"/>	Alethia Alonso	➡ null
<input checked="" type="checkbox"/>	Klaus Mauer	➡ null
<input checked="" type="checkbox"/>	Natsuo Shinohara	➡ null
<input checked="" type="checkbox"/>	root	➡ null

Members of policy
'NewPasswordPolicy'

Name

Assign >>

<< Remove

3.4.5 Settings

The Settings tab of the Administration page enables you to configure the following settings: the default time zone, parameters for the </system/mail/send>²⁷⁹ function, directory service and logging settings.

3.4.5.1 Input Format

Whenever you create jobs that use time-based triggers, you must specify the applicable time zone. For convenience, you can configure globally what time zone should be selected by default in the job configuration page.

To set the default time zone:

1. [Log on](#)⁸¹ to the FlowForce Web administration interface.
2. Click **Administration**.
3. Click **Settings**.
4. Under **Input format**, select the default time zone.
5. Click **Save**.

3.4.5.2 Parameters for System Function /system/mail/send

For jobs that send emails, you need to configure the SMTP address and port of the mail server as well as the SMTP credentials.

FlowForce will first attempt to establish a connection encrypted over TLS or SSL. If the encrypted connection fails, FlowForce attempts to start communication without encryption and then might switch the connection to encrypted if the SMTP server explicitly requires it. Otherwise, the SMTP connection remains in plain text.

Parameters for system function /system/mail/send

To configure the mail settings, take the steps below:

1. [Log on](#)⁸¹ to the FlowForce Web administration interface.
2. Click Administration | Settings.
3. Enter the name and port of the SMTP Server. Standard SMTP servers accept connections on port 25. SMTP servers that require connection to be encrypted over the SSL/TLS protocol accept connections on other ports (typically, 465 or 587).
4. If your SMTP server requires authentication, click the plus icon next to the *User authentication* parameter and enter the username and password.
5. Optionally, enter a RFC2822-compliant mailbox address value in the *Default sender* field. The value entered here is used as the default *From* parameter of the [/system/mail/send](#)²⁷⁹ and [/system/mail/send-mime](#)²⁸⁰ functions.

Test SMTP parameters

To send a test email, click the **Test SMTP Parameters** button and fill in the *From* and *To* parameters. You can test SMTP parameters without saving the settings. You must close the *Test SMTP Parameters* section before saving the settings.

3.4.5.3 Directory Service

If your organization uses Microsoft Active Directory or an LDAP-compliant directory service provider such as Apache Active Directory, OpenLDAP Server, Oracle Unified Directory, and others, you can integrate it with FlowForce Server. From the FlowForce Server perspective, integration with a Directory Service provider means the following:

- Users can log on to FlowForce Server with their domain user name and password.
- Administrators can either allow existing domain users to log on to FlowForce Server with their domain credentials (that is, an implicit user import takes place), or they can explicitly import domain users and groups into FlowForce Server. In either case, the imported accounts are visible in the user administration pages of FlowForce Server. This enables administrators to add or revoke privileges and permissions to groups or user accounts, in the same way as for the built-in FlowForce Server accounts (see [About Privileges](#)⁷² and [How Permissions Work](#)⁹²). Administrators can also assign FlowForce Server roles to groups or user accounts (see [Assign Roles to Users](#)¹¹³).
- Administrators cannot rename or change the password of domain users imported into FlowForce Server.
- Administrators cannot rename or change the membership of domain groups imported into FlowForce Server.
- Administrators can delete imported domain accounts from FlowForce Server. This does not remove the accounts from the domain and does not change in any way their associated domain privileges.
- If the imported domain accounts have FlowForce Server privileges and permissions assigned to them, they are displayed in privilege reports (see [Reports](#)¹¹⁴).

To change the **Directory Service** settings:

1. [Log on](#)⁸¹ to the FlowForce Web administration interface.
2. Go to the **Administration** menu and click **Settings**.

Directory Service

Enable

Connect using: Active Directory

Lightweight Directory Access Protocol (LDAP)

Host:
Host name, domain or IP address. A custom port can be specified with ':389'

User:
User name in the form of 'name@domain.com' or 'cn=name,dc=domain,dc=com'

Password:

SSL:
For SSL to work, LDAP server's CA certificate must be installed on the machine where FlowForce server is

Allow any domain users to log in

Default login domain: Set domain login as default

The available settings are described below.

Enable

Select this check box to enable users to log on to FlowForce Server with their domain user name and password. If you select this check box, you must select either the **Active Directory** or the **Lightweight Directory Access Protocol (LDAP)** option, as further described below.

If you select the **Lightweight Directory Access Protocol (LDAP)** option, make sure that connection details (such as username, password, and so on) are correct. When you click **Save**, FlowForce attempts to communicate with the specified LDAP server and shows an error if the connection details are not valid. Note that FlowForce Server must be able to connect to the LDAP server successfully before you can save the LDAP settings.

If you select the **Active Directory** option, the machine where FlowForce Server runs must be part of a domain controlled by Active Directory.

After you have enabled directory service authentication, an additional drop-down list becomes visible in the FlowForce Server login page, called **Login**. The **Login** drop-down list enables users to select the authentication option and contains the following items:

- *Directly*. This is the default FlowForce Server authentication option. To log in, users must supply their FlowForce username and password.
- *[A specific domain]*, depending on the configured LDAP server. To log in, users must supply their domain username and password—these are managed by the LDAP server.

See also [Logging on to FlowForce Server](#) ⁸¹.

Connect using

Select **Active Directory** to enable direct integration with Active Directory. This is applicable if FlowForce Server runs on Windows and the machine is part of a domain controlled by Active Directory.

Select **Lightweight Directory Access Protocol (LDAP)** to enable integration with an LDAP-compliant Directory Service. Fill in the details as follows:

- **Host** — Enter the host name, domain name, or IP address of the LDAP server. To add a port number, append a colon character, followed by the port number. For example, **somehost:10389**
- **User** — Enter a user name which has administrative rights to query the directory service. The user name can either be in the form of a "Distinguished-Name" (for example `cn=name,dc=domain,dc=com`) or a "User-Principal-Name" (for example, `user@some.domain.com`). Note: The "User-Principal-Name" format applies for Active Directory only; for other LDAP servers, use the "Distinguished-Name" format.
- **Password** — The user's password. Note: If you mistype the password several times, the LDAP server may lock the account. In that case, make sure that the account is not locked out before proceeding.
- **Use SSL** — Select this check box only if the LDAP server was configured to accept SSL-encrypted connections from clients. If you select this option, change the port number to the one used by the LDAP server for secure connections (typically, port **636**). If your organization already uses the same trusted root certificate on both machines, there are typically no additional configuration instructions. Otherwise, the root (CA) certificate of the LDAP server must be installed on the machine where FlowForce Server runs, as follows:
 - a. On the machine where LDAP server is, export the root certificate from the trusted certificate store. Use the tools specific to your operating system for that purpose (for example, the Certificates Snap-In on Windows).
 - b. On the machine where FlowForce Server is, import the certificate into the trusted certificate store, as described in [Import Root Certificates](#) ⁵⁵.

Note: On Windows, SSL errors are reported in **Windows Event Viewer | Windows Logs | System**, where *Source* is set to *Schannel*.

In some cases, LDAP servers can have arbitrary schemas that do not fit into a particular standard. If FlowForce Server cannot detect the schema of your LDAP provider, an error similar to "Directory Service detected an invalid LDAP schema" is displayed. In this case, copy the **directoryservice.cfg** file to the same directory as the FlowForce Server executable. When this file is present, FlowForce Server will not attempt to detect the schema of the LDAP provider automatically.

Allow any domain users to log in

Select this check box if a user's domain account should be imported into the FlowForce user database first time when users log on to FlowForce with their domain credentials. If this option is disabled, domain users can log on to FlowForce Server only if their account has already been imported into FlowForce Server by an administrator.

Default login domain

This option is visible after the **Enable** check box is selected and the settings have been saved.

The drop-down list displays all domains that this machine is member of. The same list of domains will be visible to users in the FlowForce login page, if Directory Service authentication is enabled (see the first option above).


Select the **Set domain login as default** check box if the domain should be selected as the default choice in the **Login** drop-down list of the FlowForce Server authentication page.

If you clear the **Set domain login as default** check box, the built-in FlowForce Server authentication ("Directly") is the default choice.

3.4.5.4 Logging Settings

FlowForce Server provides a logging mechanism to register all kinds of events and the time when they occurred (such as job outcome events, configuration change events, errors, and so on). You can view all the log events from a dedicated page, see [Viewing the Job Log](#)¹⁰⁴. Note that the log events can significantly increase the size of the FlowForce Server internal database over time. For this reason, the log must be periodically archived or cleaned up using the `archive-log` or `truncate-log /system/maintenance`²⁸⁴ functions. There are also other settings available that help you keep the size of the log within reasonable limits, as further described on this page.

The logging that takes place in FlowForce Server can be of two types:

1. Default system logging that does not require manual intervention of any kind. This kind of logging is taken care of by the system and does register all events, but keeps the size of each log record up to a certain limit, for better system stability and performance. If the system logging does not provide enough level of detail, or if you find out that certain log entries (such as parameter values in steps) are truncated because they are too long, you can use explicit logging, as described next.
2. Optional (explicit) logging that you can enforce from the job configuration page. The job configuration page provides a **Log**  button that you can optionally enable next to each parameter which you are interested to track in the log. Doing so will log the full value of that parameter when the job runs. In addition, you can embed any FlowForce expression inside the `log`²³² expression function in order to request that that expression be logged explicitly. Again, this will log the expression in full and its value will not be truncated. FlowForce Server does not limit the size of entries logged as a result of explicit logging.

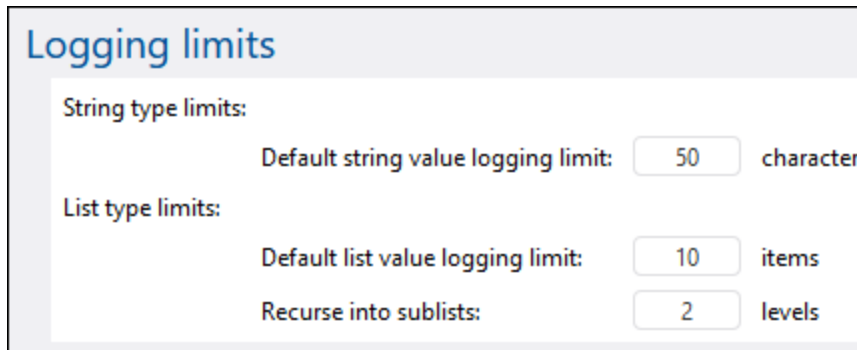
Logging limits

If you do not want to use explicit logging for whatever reason, you can alternatively change the default size of log entries maintained by the system.

Changing the default log size to a higher value may impact system stability and performance, so exercise this option carefully. The recommended approach is to use explicit logging, as mentioned above.

To view or change the default size of log entries:

1. [Log on](#)⁸¹ to the FlowForce Web administration interface.
2. Go to **Administration | Settings** and observe the parameters grouped under "Logging limits".



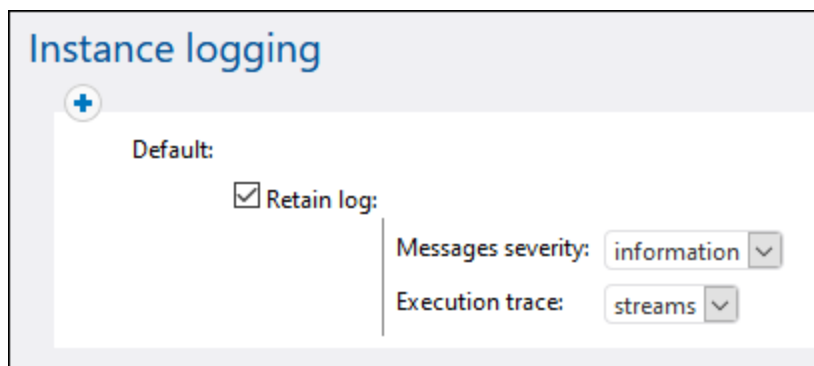
Notice that there are two kinds of logged entries: string types and list types. Consequently, there are two parameters to control the size of each type.

<i>Default string value logging limit</i>	Specifies the default length of log entries that are of type "string". If a log entry exceeds this value then long arbitrary values such as file paths will be truncated.
<i>Default list value logging limit</i>	Same as above, applies to log entries that are of type "list".
<i>Recurse into sublists</i>	This setting affects jobs which operate on lists that contain other lists as children. Set this value to instruct FlowForce to look <i>N</i> levels deep for logging purposes.

Instance logging

The settings in the "Instance logging" section specifically affect the level of information reported in the [Instance log](#)¹⁰⁶ page.

Logged messages can have severity levels, in this order (from lowest to highest): information, warning, error. The "Instance logging" parameters make it possible to skip logging of certain messages according to their severity. You can also configure the amount of tracing information that should be stored by FlowForce Server, or completely disable retention of logs. The image below illustrates the default settings:



Clearing the **Retain log** check box has the effect that no information is reported at all in the [Instance log](#)¹⁰⁶ page.


The **Messages severity** option specifies what messages should be retained:

<i>None</i>	No messages are kept
<i>Error</i>	Keep errors and critical messages
<i>Warning</i>	Keep errors, critical messages, and warnings
<i>Information</i>	Keep errors, critical messages, warnings, and information messages
<i>All</i>	This is the most verbose option. All possible messages are kept, regardless of their severity.

The **Execution trace** parameter specifies the amount of tracing detail that should be stored:

<i>None</i>	No tracing information is kept
<i>Streams</i>	Keep streams but exclude traces
<i>Trace</i>	Keep traces but exclude streams
<i>Full</i>	Keep every possible level of tracing information.



Rules

The "Instance logging" settings described above constitute a "rule". You can create custom rules, in addition to the default one, by clicking the  button. This makes it possible to apply rules conditionally, based on the outcome of the job, which can be one of the following:

<i>Successful execution</i>	The job is considered successful.
<i>Failed execution</i>	The job execution has failed.
<i>Stopped by user</i>	The job was stopped by user action, see Stop Jobs ⁸⁶ .
<i>Interrupted</i>	The service was stopped before the job could finish, or FlowForce Server crashed.

The rules defined on this page are evaluated from top to bottom. If the job outcome matches *any* of the outcomes listed above, the rule is matched. The first matching rule wins.

For example, the configuration illustrated below retains the full message log if the job execution was not successful. In other words, the first rule will be triggered if the outcome is "Failed execution" or "Stopped by user" or "Interrupted". On successful execution, the "Default" rule will be triggered instead, and, even though the log messages will be kept, no tracing information will be available.

Note that you can add all the custom rules only *before* the default rule, not after it. To change the order of rules, use the **Up**  and **Down**  buttons. These buttons are enabled only when there are three or more rules.

If you define custom rules, it is advisable to use the default rule as a "catch all" filter, in case none of the rules before it has matched.

Logging rules at object level

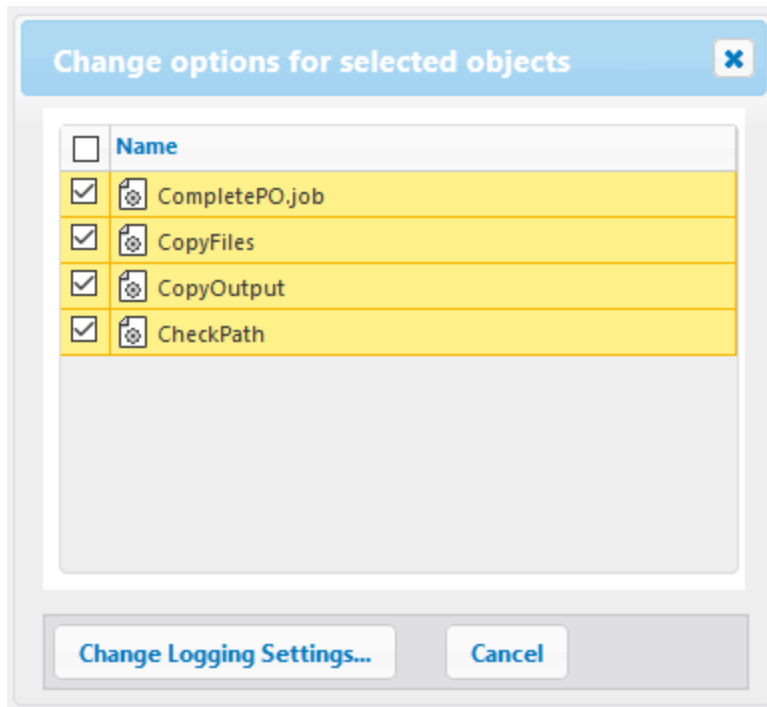
You can create logging rules not only globally at application level, but also for specific FlowForce Server jobs. Note that, if you create a rule on a job that has sub-jobs, then the rule will apply to all the sub-jobs as well.

To set logging rules for a job:

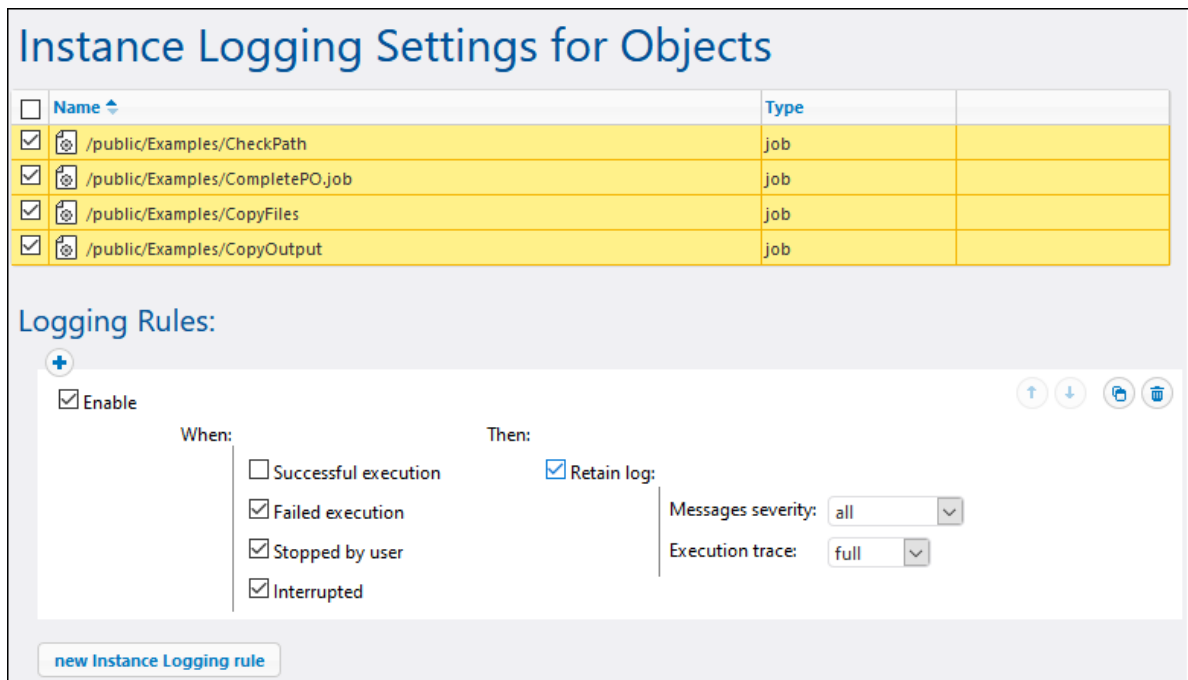
1. Open the job configuration page.
2. Click the **Logging Settings** button in the job configuration page.
3. Click **New Instance Logging rule**.

To set logging rules for multiple jobs:

1. Click **Configuration** and open a container.
2. Select one or more jobs (or the entire container), and then click **Logging settings for selected jobs**. A dialog box appears where you can refine the selection if necessary:



3. Click **Change Logging Settings**.
4. Click **New Instance Logging rule**.



All the logging configuration settings work in the same way as described above in the "Rules" section.

If you have defined logging rules both at object level and at application level, then the priority is established as follows:

- The logging rules defined at object level are checked first.
- If there is a match found at this level, the rule is applied and the rules at application level are no longer checked.
- If there is no match found at this level, the rules at application level are checked.

4 Job Configuration

This section explains how to configure a job in FlowForce Server. Job configuration includes the following procedures, some of which are optional (e.g., caching job results):

- Creating/duplicating a job (*subsection below*)
- [Adding input parameters](#)¹³⁰
- [Adding execution steps](#)¹³²
- [Caching job results](#)¹⁴⁸
- [Setting triggers](#)¹⁵¹
- [Configuring jobs as Web services](#)¹⁵⁸
- [Defining credentials](#)¹⁶²
- [Defining queue settings](#)¹⁶⁶

The job configuration process is closely associated with performing various calculations, computing expressions, and calling functions. For details, see the following sections:

- [Expressions](#)¹⁷¹
- [Expression Functions](#)¹⁸⁰
- [System Functions](#)²³⁵

For information about basic concepts and terms associated with job execution, see [Terminology](#)¹⁵.

Windows network paths

When you create jobs, you will need to refer to file paths on the machine where FlowForce Server runs or to file paths on the network. When you refer to a Windows network path (e.g., a mapped network drive), use the Universal Naming Convention (UNC) syntax. This is necessary because drive letters are not global to the system, and each logon session is assigned its own drive letters.

The UNC has the following syntax: `\\server\sharedfolder\filepath`, where `server` refers to the server name in the network (defined by the DNS); `sharedfolder` refers to a label defined by the administrator (e.g., `admin$` is generally the root directory of the operating system installation); `filepath` refers to the subdirectories below the share.

Create/duplicate a job

This topic provides instructions on how to create jobs in FlowForce Server. The instructions will help you understand the structure of jobs and their settings. Job configuration is a flexible process, which allows you to find more than one way to achieve the same result. To get an idea of various tasks you can perform, see [Job Examples](#)²⁹⁴.

Prerequisites

Make sure you have the following [permissions for the container](#)⁹⁹ in which you want to create a new job:

- Container: *Read, Write*
- Configuration: *Read, Write*

Create a job

Before creating a job, it might be a good idea to store the credentials of the operating system user account with which the job will be executed. For more information, see [Credentials](#)¹⁶³. To create a job, follow the instructions below:

1. Go to the Configuration page and select a container in which you want to create a job.
2. Click **Create** and select **Create Job**. Enter a job name and, optionally, a job description.
3. If you need to pass some values to the job at runtime, create the required job input parameters. For details, see [Input Parameters](#)¹³⁰.
4. In the *Execution Steps* section, [add steps](#)¹³² of the job. Every job must have at least one step.
5. If the last step of the job returns a result, and if you intend to use the result in other jobs, select the return type in the [Execution Result](#)¹⁴⁶ section.
6. If you want FlowForce Server to cache the returned result, specify [caching preferences](#)¹⁴⁸.
7. In the *Triggers* section, add [a trigger](#)¹⁵¹ (or triggers) that will fire the job. If the job runs as [a Web service](#)¹⁵⁸, adding a trigger is not necessary.
8. In the *Credentials* section, select an existing credential record or specify a local credential. For details, see [Credentials](#)¹⁶².
9. If the job returns a result that you want to use in other jobs or configure as a Web service, define the job's [cache settings](#)¹⁴⁸.
10. Optionally, define the job's [queue settings](#)¹⁶⁶.
11. Click **Save**. FlowForce Server validates the entered information. If there are any fields that require your attention, FlowForce Server will highlight them in red.

Duplicate a job

You can create copies of existing jobs when necessary. This can save you time, for example, when you need to quickly create a job using an existing one as a template. To create a copy of an existing job, take the steps below:

1. Open an existing job and click **Save As** at the bottom of the page.
2. Enter the name of the new job and click **Save As**.

If the credentials of the existing job are defined locally within the job, FlowForce will prompt you to enter the password again, for security reasons. If the credentials are defined as standalone credentials, this step is not necessary. For information about standalone and local credentials, see [Credentials](#)¹⁶².

If certain job components cause conflicts when the job is duplicated, FlowForce displays an error and does not duplicate the job. For example, if you attempt to duplicate a job containing a Web service, the service is already in use by the original job and cannot be duplicated. In this case, change the URL of the Web service or remove it completely.

The duplicated job is saved to the same container as the existing job. If you want to move it to a different container, go to the parent container page, select one or more jobs, and click **Move Selected Objects**.

4.1 Input Parameters

In FlowForce Server, job input parameters are similar to function arguments in a programming language. Parameters can be of various types: e.g., file and directory references, text, numbers, Boolean values, and others. Under certain conditions, job input parameters become automatically available on the configuration page of your job. For example, when you add [a file system or HTTP trigger](#)¹⁵² to your job, the *triggerfile* input parameter is added automatically. You can use the *triggerfile* in a step function, for example, to upload this file to an FTP server. For an example, see the [system/ftp/store](#)²⁷⁰ function.

Job input parameters

The fields of a job input parameter are described in the table below.

<i>Name</i>	<p>Mandatory field. Specifies the name of an input parameter. You may need to refer to this parameter later from any of the job's execution steps. Therefore, it is recommended to use a descriptive name.</p> <p>The input parameter name must start with a letter and may contain only the following characters: a-z, A-Z, 0-9, and '_'.</p>
<i>Type</i>	<p>Mandatory field. Specifies the data type of the input parameter, which can be one of the following:</p> <ul style="list-style-type: none"> • string • string as file • string as directory • string as file or directory • stream • number • boolean • credential • certificate • result • AS2 partner (<i>Advanced Edition</i>) • AS2 MDN (<i>Advanced Edition</i>) • SFTP connection (<i>Advanced Edition</i>)
<i>Default</i>	<p>Optional field. Specifies the default value of the parameter. This value will be used if no value is specified by the job caller at runtime.</p>
<i>Description</i>	<p>Optional field. Describes the purpose of the parameter. This description becomes available as a tooltip next to the parameter name when you use the current job as an execution step of another job.</p>

Function parameters

Most [step functions](#)¹⁵ have parameters. When you select a function in an execution step, the function parameters become automatically available. Parameters and their data types vary, depending on the function you select. Parameters can accept different values, including [expressions](#)¹⁷¹ and [expression functions](#)¹⁸⁰.

Under certain conditions, previously set values of function parameters are copied over into parameters of a new function. The parameter value is copied over if the parameter has the same internal name and the same data type in both step functions. Besides this condition, one or more of the following conditions must also be true:

- The parameter is an expression.
- The parameter has logging enabled.
- The parameter is an inline credential with a user name and password.

The parameter value is *not* copied over if:







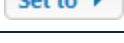
- the parameter is locked (mapping with more than one parameter with the same name);
- the parameter value starts with `altova://packagedfile/` (it will not work with any other function except the one it was deployed with).

To revert to the default value of an optional parameter, use the **Set to** button and choose `<Default value>`. For mandatory parameters, the value has to be removed manually.

As soon as a function is selected and its parameters are retrieved, the parameters are cached for this job configuration page. Every time this function is selected again (for any new step, existing step, or the same step), its parameters are not retrieved; instead, the cached parameters are used.

Buttons

Use the following buttons to manage job input and function parameters.

	Add a parameter.
	Delete a parameter
	Duplicate a parameter.
	Move a parameter up or down.
	Undo the previous delete action.
	Enable/disable logging of the parameter value
	Set a specific value of a parameter

4.2 Job Execution Steps

In FlowForce Server, steps define what a job must do (e.g., delete a file, execute a MapForce mapping, send an email). In its simplest form, a step is an operation with failed or successful outcome. Each step must execute [a function](#)¹⁵. You can create as many steps as required for your job and set the order in which the steps must be executed. You can also use [the result of a step](#)¹⁴⁵ in other steps.

Add an execution step

To add an execution step, [create a job](#)¹²⁸ or open an existing job and select the relevant step type in the *Execution Steps* section of the job configuration page.

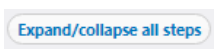





Types of steps

Execution steps can be of the following types:

- [Execution steps](#)¹³²
- [Choose steps](#)¹³³
- [For-Each steps](#)¹³⁵
- [Error/Success-Handling steps](#)¹³⁶
- [Postpone steps](#)¹⁴⁰
- [Resume steps](#)¹³⁸ (used inside Error/Success-Handling steps)

Buttons

The table below gives a list of buttons that allow you to manage steps.

	Allows you to expand or collapse all execution steps. This option could be useful when you want to do a search in the browser or print the page.
	Add a step.
	Delete a step.
	Duplicate a step.
	Move a step up or down.
	Undo the previous delete action.

4.2.1 Execution Steps

An Execution step allows you to execute a specific function. Execution steps can include [system functions](#)²³⁵, [deployed MapForce mappings](#), [StyleVision transformations](#)⁴¹⁷, and [other jobs](#)³¹⁶. For example, you can add the [system/send-mime](#)²⁸⁰ function to send an email (possibly with an attachment) to the specified recipients.

You can configure Execution steps as standalone steps or integrate them into [Choose](#)¹³³, [For-Each](#)¹³⁵, [Error/Success Handling](#)¹³⁶, and [Postpone](#)¹⁴⁰ blocks.

FlowForce Server processes execution steps sequentially, starting with the first (topmost) step down to the last step. This rule also applies to any sub-steps that a step may have. The code listing below illustrates this scenario:

```
Step A
Step B
  Step B1
  Step B2
Step C
Step D
  Step D1
```

Error handling

By default, if FlowForce encounters an error, processing stops, and subsequent steps are not executed. However, in most [system functions](#) ²³⁵, you can set the *Abort on error* parameter to `false`, which will prevent job execution from stopping even after an error has occurred.

4.2.2 Choose Steps

A Choose step allows you to define conditions under which steps within the Choose block will be executed. You can define any number of conditional steps. Within any *When/Otherwise* pair, FlowForce Server executes only the condition that is true; the other condition is ignored.

Depending on your business needs, you can create [Execution steps](#) ¹³², [For-Each](#) ¹³⁵, [Error/Success Handling](#) ¹³⁶, and [Postpone](#) ¹⁴⁰ blocks inside a Choose block. You can also nest other Choose blocks inside your Choose block.

Structure of a Choose block

The Choose block has the following structure:

```
When {expression}
  Execute (step)
Otherwise
  Execute (step)
```

Some of the possible scenarios of using Choose blocks are described below.

Multiple When expressions

The Choose block can contain multiple *When* expressions (*see code listing below*). In such scenarios, FlowForce Server executes the first *When* expression that is true and exists the Choose block. Even if other *When* expressions are also true, they will not be executed.

```
When {expression}
  Execute (step)
When {expression}
  Execute (step)
When {expression}
  Execute (step)
Otherwise
  Execute (step)
```

Nested When expressions

Under each Choose step, you can nest other Choose steps (sub-conditions), for example:

```
When {expression}
  When {expression}
    Execute (step)
  Otherwise
    Execute (step)
Otherwise
  Execute (step)
```

Choose step in a job

The code listing below shows how a Choose block is integrated into the job. The job has steps A, B, C, D, E, and F. Steps C and D are part of the Choose block. After Step B has been executed, Step C or D will be executed, depending on the `When` expression. If the `When` expression returns `true`, Step C will be executed. Otherwise, Step D will be executed. After Step C or D has been executed, the job proceeds to Step E and then Step F.

```
Step A
Step B
When {expression}
  Execute Step C
Otherwise
  Execute Step D
Step E
Step F
```

Example

The job shown below checks the numeric value of the `temperature` input parameter that defines whether the weather is hot or normal. The `When` statement in the Choose block will compute `It is boiling hot!` if the value of the `temperature` parameter is greater than 35. Otherwise, the string `It is alright` will be computed. For simplicity, we have set the default value of the `temperature` parameter to 36. Therefore, the job will return the `It is boiling hot!` string.

If you intend to pass the result of a Choose step to another step or declare the result to be of a particular type, ensure that the `When` and `Otherwise` conditions have the same return type. Otherwise, validation errors might occur.

Job Input Parameters

+
Name:
Type:
Default:
🗑️

Execution Steps

+

▲ **Choose**

When

+

▲ Execute function

Parameters: | Expression:

= Assign this step's result to as T0

+

Otherwise

+

▲ Execute function

Parameters: | Expression:

= Assign this step's result to as T0

+
 = Assign this step's result to

4.2.3 For-Each Steps

A For-Each step allows you to iterate through a sequence (e.g., a list of files within a directory) and repeat an execution step any number of times. The For-Each block has the following structure:

```

For each item in sequence {sequence expression}
  Execute (step)
  
```

Depending on your business needs, you can create [Execution steps](#)¹³², [Choose](#)¹³³, [Error/Success Handling](#)¹³⁶, and [Postpone](#)¹⁴⁰ blocks inside a For-Each block. You can also nest other For-Each blocks inside your For-Each block.

A For-Each block is executed until FlowForce Server finishes looping through all the items of the sequence expression. For an example of a job that uses a For-Each step, see [Copy Files](#)³⁰³.

4.2.4 Error/Success-Handling Steps

When a step of a job fails, the job is considered failed as well. To perform some clean-up actions before the job exits (such as logging or sending email notifications), you can create error/success handling steps. These steps allow you to protect the execution of one or more steps (i.e., protected steps). Error/success handlers and protected steps are part of the so-called *protected block* (see *Structure of a Protected Block* below).

Depending on your business needs, you can create [Execution steps](#)¹³², [Choose](#)¹³³, [For-Each](#)¹³⁵, and [Postpone](#)¹⁴⁰ blocks inside a protected block. You can also nest other protected blocks inside your protected block.

Add an error/success handler

To add an error/success handler, [create a job](#)¹²⁸ or open an existing job and click **new Error/Success handling step**. Then select the relevant error/success handling step.

Structure of a protected block

The example illustrates the structure of a sample protected block. The protected block below consists of the *protected steps* (Steps A and B) and the *error/success handlers* (Steps C, D, E and F). Whenever the protected steps have been executed (successfully or with an error), the handler blocks will be executed next.

Protected block

```

Step A
Step B

On error
  Step C
On retry
  Step D
On success
  Step E
Always
  Step F

```

Protected block

Types of handlers

Each handler block has a special condition, and if this condition is true, the steps of this handler will be executed. Error/success handlers are listed below:

- The *On-Error* block is executed if any of the protected steps has failed.
- The *On-Success* block runs if all protected steps have been successful.

- The *On-Retry* handler runs if any of the protected steps has failed. The protected steps run as many times as specified in the *retry count* option (see *Retry Option below*). By default, this option is set to 0.
- The *Always* handler runs regardless of whether the protected steps have been successful or not.

Order of handler execution

Handler blocks are always executed in the specified order. For example, if there is an Always block followed by an On-Error block, which is in turn followed by another Always block, the two Always blocks will be executed in the order specified whenever the protected steps have finished executing. The On-Error block will run after the first Always block only if the protected steps have encountered an error.


In the sample protected block above, if Step A or B fails, the protected steps are left, and Steps C and F are guaranteed to be executed (because they are On-Error and Always, respectively). Step D will be executed only when retries are left (see *subsection below*).

Retry option

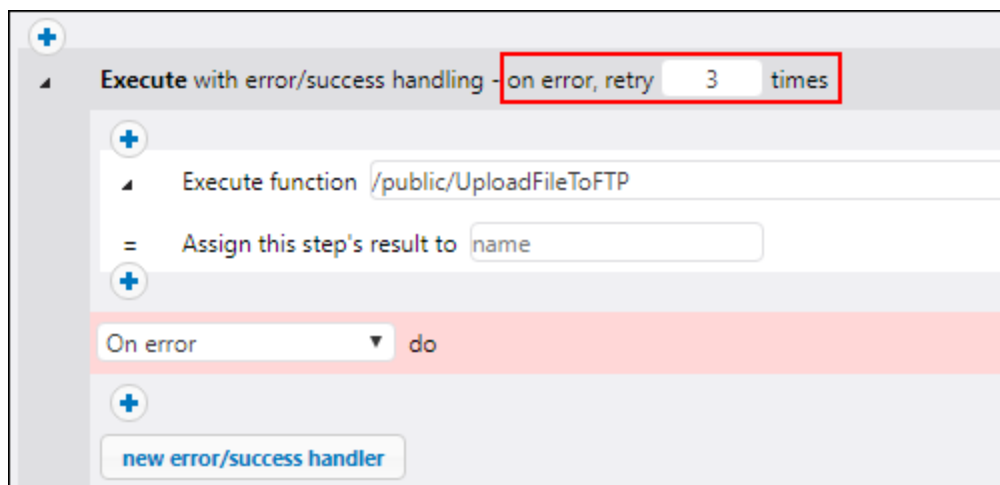
There are cases, in which you may want to run a step again if its execution has failed. For this purpose, FlowForce Server provides the *Retry* option. By default, this option is set to 0, which means that the protected block will be executed once only and no retries will be attempted.

On-Retry blocks are executed only if the protected steps have retries left. The actual retry will start only after all the handler blocks have been successfully executed and only if the protected steps have been left because of an error. When no retries are left, the error is re-raised outside of this protected block.

To add an execution step that will be retried a certain number of times, follow the instructions below:

1. Specify the number of retries you need (e.g., 3 times) in the text field at the top of the Error/Success-Handling step (see *screenshot below*).
2. Click the  button under the *Execute with error/success handling* section and add a step (or steps) that will be retried if an error occurs.

In the example below, the job, which uploads a file to an FTP server, will be retried 3 times on error. The number of retries of any given job instance is reported in the [log](#)¹⁰⁴. In addition, if you need to get and process the number of retries at runtime, call the [retry-count](#)²³¹ expression function.



Set timeout

You can set a timeout on an Error/Success-Handling step, by clicking the **set timeout** button at the top of the Error/Success block. When you click this button, two options will pop up: *timeout* and *force stop after additional N seconds*. The first timeout aborts a job step if it is not successful after the specified time (in seconds). The job step might comply with the request or ignore it. When the first timeout has expired, the second timeout will take place and abort the job step forcefully. Timeouts render the aborted step failed. If you have defined On-Error, On-Retry, or Always handlers, these will be processed next.

Resume steps

Resume steps make it possible to continue execution even after an error has been encountered. Resume steps are relevant to error handling and can occur only in protected-block handler sections. Resume steps can also be used inside [conditionals](#)¹³³ that are used inside protected-block handler sections.

How Resume steps work

The main idea of the Resume step is to replace the result of the failed protected step with a newly constructed result object. This prevents the job from being aborted in case an error is encountered inside the Error/Success-Handling step. When a Resume step has been executed, the following actions take place:

1. The result of the corresponding protected block is set to the computed expression.
2. After that, the Resume step leaves the protected block it is a handler of.
3. Execution resumes after the protected block, which could be the next step after it or job end, as if the protected block were successful.

Once the Resume step has finished executing, no further handler blocks of the current protected block or even steps within the current handler block will be executed. If you need to define some cleanup operations, you can nest another protected block inside the first one and add, for example, On-Always or On-Success handlers.

Add a Resume step

To add a Resume step, follow the steps below:

1. Create a Success/Error-Handling step.
2. Click the **Plus** icon at the desired location.
3. Select *Resume Step* from the list.
4. The Resume step will appear and contain a text field in which you will need to supply an expression function. The result of the current protected block will be set to the expression computed in this field.

The expression passed to the Resume step must be compatible with the protected block return type, which is the type of the last step executed inside the protected sequence. The list below provides the expression functions you can use:

- [make-error-result](#)²⁰⁷
- [make-success-result](#)²⁰⁸
- [merge-results](#)²⁰⁹

Where to place a Resume step

There are different locations where you can place Resume steps in protected blocks. Some of the possible examples are given below:

- *Example 1:* You can place a Resume step inside a handler block.
- *Example 2:* You can also split the protected block into two nested ones, one which contains a Resume step and the other which executes any On-Always handlers that are left.

Example

In practice, it is not necessary to define all handler types for every job. The most common scenario is to define only On-Error and Always handlers (*screenshot below*).

In the sample job below, the first step runs a script from the `c:\scripts` directory by calling the [\system\shell\commandline](#)²⁸⁵ function. The execution of this step is protected by two handlers: On-Error and Always. Only if the execution of the first step fails will the On-Error handler send an email with the ID of the failed job instance in the subject line. The Always handler will be executed regardless of whether the first step has been successful or not. The Always handler will log a message by running a script from the `c:\scripts` directory.

For more examples, see [Adding Error Handling to a Job](#)³²⁵.

Execute with error/success handling

+

Execute function `/system/shell/commandline`

Parameters: Command:

Abort on error: **+**

Working directory:

= Assign this step's result to as result

+

On error

+

Execute function `/system/mail/send`

Parameters: From:

To:

Subject:

Message body: **+**

Attachment: **+**

Abort on error: **+**

= Assign this step's result to as boolean

+

Always

+

Execute function `/system/shell/commandline`

Parameters: Command:

Abort on error: **+**

Working directory:

4.2.5 Postpone Steps


A typical FlowForce job returns a result *only after* the processing of all steps has finished, assuming that no error has occurred. For jobs configured as [Web services](#)¹⁵⁸, this means that the HTTP transaction must be kept open for the entire duration of the job execution, which may take several minutes or even hours in some

cases, depending on the amount of processed data. To handle such cases more efficiently, you can create postponed steps.

Postponed steps take place *only after* all non-postponed steps have been processed and the job has returned a result. Even though a job with postponed steps might return a result early, the job is considered in progress until the execution of all postponed steps has finished.

Add a Postpone step

You can create postponed steps anywhere in the job where a step is allowed. To add a postponed step, take the steps below:

1. [Create a new job](#) ¹²⁸ or open an existing one.
2. Click the **new Postpone step** button in the *Execution steps* section.
3. Click  inside the Postpone block to add a step or several steps that will be postponed.

Depending on your business needs, you can create [Execution steps](#) ¹³², [Choose](#) ¹³³, [For-Each](#) ¹³⁵, and [Error/Success Handling](#) ¹³⁶ blocks inside a Postpone block. You can also nest other Postpone blocks inside your Postpone block.

Error handling

A job can contain multiple Postpone blocks (each containing a step or several steps to be postponed) at various places in the job. Creating postponed steps might be useful for error handling: If an error occurs within a postponed block, other steps in the job will not be affected. A postponed block is like a mini-job and behaves in the same way as a regular job:

- If a step in a postponed block encounters an error, the step will be canceled, along with any subsequent steps in the same postponed block, and an error will be logged.
- Postponed blocks do not influence each other. In a job with multiple postponed blocks, a postponed block will run even if the preceding postponed block results in a failure.
- If a postponed step within a [protected block](#) ¹³⁶ encounters an error, all postponed steps that are part of that block will be canceled.

Possible scenarios

The subsections below describe some of the possible scenarios of using postponed steps.

Job with several postponed steps

The sample job below will run in the following order: A, C, B, D. The non-postponed steps are executed first, followed by the postponed steps. Step C returns a result.

```
A
postpone B
C
postpone D
```

Postpone steps in Choose steps

You can also add postponed steps to Choose steps. In this case, the postponed step will be run only if the respective *When* or *Otherwise* branch is run as well.

```
when expression=true
{
  postpone A
```

```

    B
    C
  }
  otherwise
  {
    postpone D
    E
    F
  }

```

In this job, if the expression is `true`, the steps will run in the following order: B, C*, A. Otherwise, the run order will be: E, F*, D. The asterisks indicate steps that return results.

Postpone steps in For-Each steps

The sample job below shows a For-Each step, in which the postponed steps will be processed after the non-postponed steps, in the same order as in the loop they are part of.

```

for each item in list
{
  A
  postpone B
}

```

For example, if the loop runs three times, the steps above will run in the following order: A1, A2, A3*, B1, B2, B3. The digits indicate loop numbers. The asterisk indicates a step that returns a result.

Postponed steps nested in postponed steps

You can also nest postponed steps within other postponed steps (see code listing below). In this case, outer steps of the same depth are processed first, and the nested postponed steps will be executed only after the processing of their parent sequence has finished. In our sample job, the steps will run in the following order: A, G, N, B, D, F, C, E, H, K, M, J, L. Step N returns a result.

```

A
postpone
[
  B
  postpone C
  D
  postpone E
  F
]
G
postpone
[
  H
  postpone J
  K
  postpone L
  M
]
N

```

If you need to create and test advanced configurations like the one above, you can always track the execution order of steps in the [log](#)¹⁰⁴.

Example

The example below illustrates a possible scenario in which postponed steps might be useful. This job is run as [a Web service](#)¹⁵⁸ and can be invoked at any time by a client, including from the browser.

Step A runs a time-consuming shell command that lists recursively all the directories and files within a large system directory. For this reason, Step A is defined as a postponed step. Step B takes the standard output (`stdout`) produced by A and writes it to a file. Step B depends on the output produced by Step A and, therefore, has to be part of the postponed sequence, too. Step C informs callers of the service that the task has been submitted successfully. Whenever the Web service is called, the steps above will run in the following order: C, A, B. The reason for that is that A and B are postponed steps, so C is executed first.

The advantage of this configuration is that the job returns the result immediately after running Step C, and the HTTP transaction can end, freeing up server resources for other requests. After returning the result of the job, FlowForce will go on to run postponed Steps A and B.

When you invoke the job above in your browser, the message *Task has been submitted successfully* will be displayed in the browser. At the same time, the job continues running until it creates `output.txt`. If neither A nor B fails, the output file will be created at the following path: `C:\FlowForce\Postponed\output.txt`.

The screenshot displays the 'Execution Steps' configuration for a 'Postpone' job. The steps are listed in a vertical sequence:

- Step A:** 'Execute function /system/shell/commandline'. Parameters include Command: 'dir "C:\Program Files" /S', Abort on error: +, and Working directory: +. It is configured to 'Assign this step's result to output as result'.
- Step B:** 'Execute function /system/filesystem/copy'. Parameters include Source: '{as-file(stdout(output))}', Target: 'C:\FlowForce\Postponed\output.txt', Overwrite: , Abort on error: +, and Working directory: +. It is configured to 'Assign this step's result to name as boolean'.
- Step C:** 'Execute function /system/compute'. Parameters include Expression: 'Task has been submitted successfully'. It is configured to 'Assign this step's result to name as T0'.

Note about step order

In our example, Step C has to be the last one in the job, because this step outputs the result to the browser. If you move Step C to the very top, it is still executed first, and Step B is still the last to be executed. However, this would change the job result, and the browser would display some empty output similar to []. The reason is that the result of a job is always the result of the *last executed step*. Postponed steps do not have a return value but produce an empty sequence instead.

4.3 Execution Result

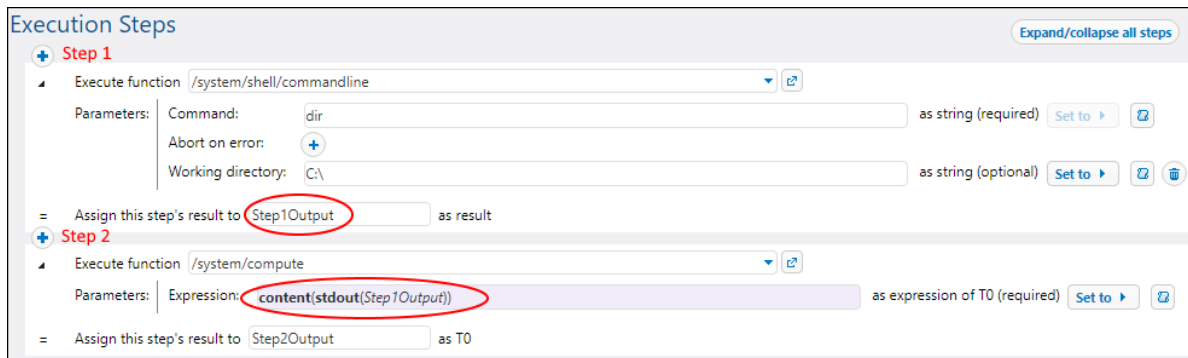
This topic explains how to use the result of a step in another step, how to change the data type of the step result, and how to declare the return type of the whole job.

Step result

There are situations in which you may want to use the step result in an other step. In this case, you will most likely need to change the data type of the step result. Our sample job has an execution step which lists files and directories located on the C drive (*Step 1 in screenshot below*). When this step returns a result, FlowForce Server automatically assigns the generic type `result` to the step result. Our goal is to send the directory listing by email.

Since the *Message body* parameter in the [/system/mail/send](#)²⁷⁹ function is of type `string` (see *Step 4 in Send by Email*), we will need to convert `result` to `string`. To achieve this goal, we will use the [stdout](#)²⁰² and [content](#)¹⁸³ functions in one expression, which will allow converting `result` to `stream` and then to `string` (see *Expression in Step 2 below*). Follow the instructions below.

1. Fill in the function and parameters, as shown in Step 1 (see screenshot below).
2. Assign the result of Step 1 to Step1Output: Type Step1Output in the *Assign this step's result to* field (circled in red below). You will need to refer to this value later to access the result of the step.
3. Add a new execution step which calls the [/system/compute](#)²³⁷ function.
4. Enter the following expression in *Parameters*: `content(stdout(Step1Output))`, where Step1Output is the result of Step 1.
5. Assign the result of Step 2 to Step2Output.



Now the job output data type is a string. The next step is to create a new execution step that will send the result of Step 2 by email.

Send by email

To send the directory listing to an email address, take the following steps:

1. Add a new execution step, as shown in the screenshot below. Before using the [/system/mail/send](#)²⁷⁹ function, you must configure the [mail server settings](#)¹¹⁸.

Step 3

Execute function `/system/mail/send`

Parameters:

- From:
- To: as string (required) [Set to](#)
- Subject: as string (required) [Set to](#)
- Message body: as string (optional) [Set to](#)
- Attachment:

Click "Set to" and select "Step2Output"

= Assign this step's result to as boolean

- The last step is to add a trigger that will fire the job. In our example, we have added a timer trigger that will run the job every 60 minutes (see below) and click **Save**.

Triggers

Run every day(s)

Repeat every minutes the whole day, or from to

Start:

Expires:

Time zone:

enabled

[new Timer](#) [new Filesystem trigger](#) [new HTTP trigger](#)

Job result

If you want to [cache the result of your job](#)¹⁴⁸, you must declare the return type of the job. Declaring the return type of a job might also be meaningful if you want to use your job as an execution step of other jobs. When you declare the return type, keep in mind the following points:

- Declaring the return type is meaningful only for jobs that actually return a result.
- If you want to cache the result of a job, you must declare the return type.
- The return type of a job must be the same as the data type of the last step in the job. Otherwise, FlowForce Server returns an error. When type-matching errors occur, use [expression functions](#)¹⁸⁰ to change the data type of the last step in the job to the data type declared as the job return type. For details, see the example above.

To define the return type of a job, take the following steps:

- Create a new job or open an existing one for editing.
- Select a return type in the *Execution Result* section on the Configuration page.

Return types

The available return types are listed below.

- ignore/discard
- string
- stream
- number
- boolean
- credential
- certificate
- result
- AS2 partner (*Advanced Edition*)
- AS2 MDN (*Advanced Edition*)
- SFTP connection (*Advanced Edition*)

The default option is `ignore/discard`. It instructs FlowForce Server to ignore or discard the result of the job. Select this option if the job does not return a result or if you do not need to process the returned result in any way.

4.4 Cache Result

Caching is a useful feature that reduces the server load and the response time of jobs. Caching the result of a job means that FlowForce Server prepares and stores the job result in some internal repository (i.e., the cache). If the job has parameters, the system creates a cache entry for every parameter combination. When the job with the cached result is called from another job (referred to as a *consumer*), FlowForce Server returns the cached result to the consumer (instead of executing the job again), which reduces the response time.

When you work with cached job results, note the following:

- It is mandatory [to declare the return type](#) ¹⁴⁶ of a job whose result is cached.
- The cached job and the consumer job must use the same credentials. If the credentials differ, the job executes as if no cache were defined.
- When you change the configuration of the cached job, the existing cache data is invalidated.

For examples, see [Caching Job Results](#) ³⁴⁷.

Caching settings

The screenshot below shows the *Caching Result* section of the Configuration page. The available settings are listed below.

Cache the result

Select this check box if you want the job results to be cached. By doing so, you are instructing any consumers of the current job to read the cached result rather than execute the job. If the current job is executed directly (not through a consumer), FlowForce Server refreshes the cache. The job is executed directly when, for example, a defined trigger has fired or the job's Web service has been invoked. If the job parameters are not found in the cache, a new cached entry is created based on the supplied parameter combination.

Initiated by consumer

When this option is enabled, any job that is calling the current job (i.e., the consumer job) will compute and populate the cache if it does not exist. Otherwise, only triggers and Web service calls will populate the cache.

Maximum number of cache entries

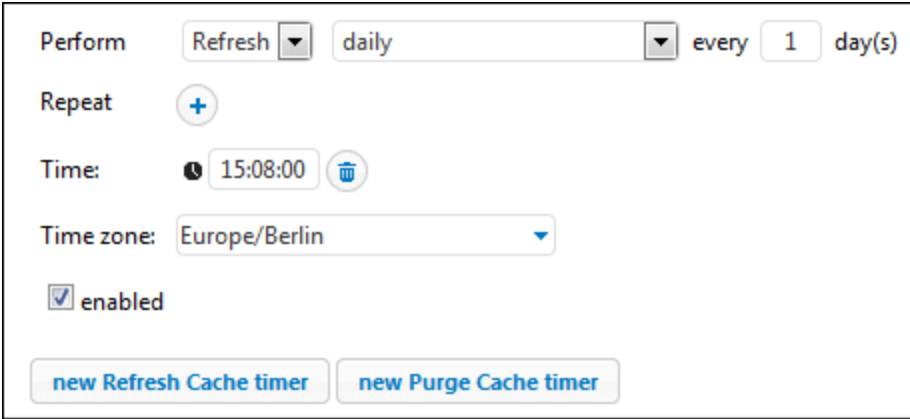
This option restricts the number of cached job results per job. When the job has parameters, you might want to set this option to the number of all possible parameter combinations.

Auto-create a new cache consumer job

A cache consumer job is a Web service at the HTTP address you specify. The consumer Web service acts as a convenient way to retrieve and manage the cache of the job whose result is being cached. When invoked, the consumer job attempts to use the cached result of the main job in the first place. If there is no cached result and the *Initiated by consumer* option is disabled, the consumer retrieves the actual result returned by the main job. If there is no cached result and the *Initiated by consumer* option is enabled, the consumer retrieves the actual result returned by the main job and populates the cache.

Refresh Cache timer

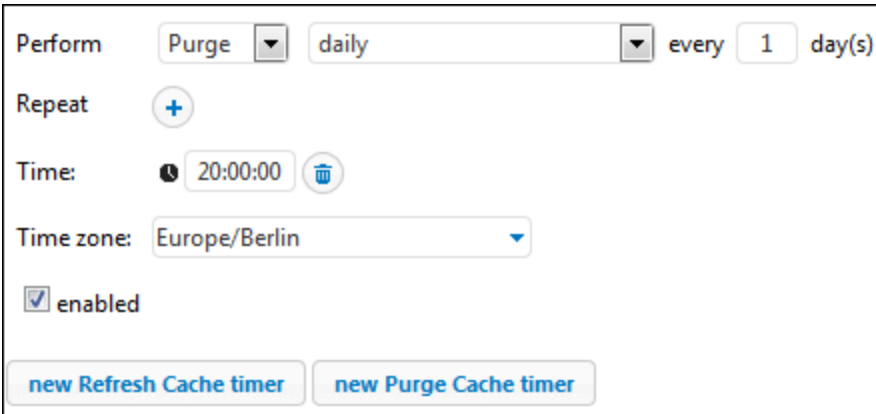
The Refresh Cache timer (*see below*) controls how often the system should refresh the cache of the current job. All currently cached parameter combinations are refreshed.



The screenshot shows a configuration panel for the Refresh Cache timer. It includes a 'Perform' dropdown set to 'Refresh', a frequency dropdown set to 'daily', and a value of '1' for 'every' days. There is a 'Repeat' button with a plus sign. The 'Time' field is set to '15:08:00' with a trash icon. The 'Time zone' dropdown is set to 'Europe/Berlin'. An 'enabled' checkbox is checked. At the bottom, there are two buttons: 'new Refresh Cache timer' and 'new Purge Cache timer'.

Purge Cache timer



The Purge Cache timer (*see below*) controls how often the system should purge the cache of the current job.



The screenshot shows a configuration panel for the Purge Cache timer. It includes a 'Perform' dropdown set to 'Purge', a frequency dropdown set to 'daily', and a value of '1' for 'every' days. There is a 'Repeat' button with a plus sign. The 'Time' field is set to '20:00:00' with a trash icon. The 'Time zone' dropdown is set to 'Europe/Berlin'. An 'enabled' checkbox is checked. At the bottom, there are two buttons: 'new Refresh Cache timer' and 'new Purge Cache timer'.

Save and Refresh the Cache

Click this button to refresh the cache manually. The button is located at the bottom of the Configuration page.

If you want to delete Refresh Cache and Purge Cache timers, click the  button. The  button (**Duplicate**) enables you to create a copy of the current trigger with the same settings.

4.5 Triggers

When you create a job, you must specify conditions that will start the job. These conditions are known as [triggers](#)¹⁵¹. FlowForce Server continuously checks for trigger conditions and executes the job whenever a specific trigger condition is met. A job can have multiple triggers.

You can create multiple triggers for the same job and enable or disable any of the defined triggers. Whenever any of the enabled triggers fires, FlowForce Server executes all steps of the job. If you use triggers in jobs that have [input parameters](#)¹³⁰, all these parameters must have default values; otherwise, the job will not be executed. The following types of triggers are available in FlowForce Server:

- [Timer triggers](#)¹⁵² allow you to schedule jobs to start at a specific time and run for a specific time interval. Time triggers can be set to run daily, weekly, on specific days of the week or month.
- [File system triggers](#)¹⁵⁴ start jobs when there is a change in a file or folder. Note that deleted files cannot be monitored. You can configure the directory polling interval (e.g., every 60 seconds) and optionally set the start and expiry date of the trigger. You can also use wildcards to filter specific files of the directory.
- [HTTP triggers](#)¹⁵⁵ enable you to poll a URI for changes. Specifically, you can poll the `Last-Modified` and `Content-MD5` HTTP header fields for changes. You can configure the polling interval (e.g, every 60 seconds) and optionally set the start and expiry date of the trigger.

Add a trigger

To add a trigger, [create a job](#)¹²⁸ or open an existing one, navigate to the *Triggers* section of the job configuration page, and add the relevant trigger.

Enable/disable triggers

By default, when you create a trigger, the *enabled* check box is selected, which means the trigger is active. To disable a trigger, clear the *enabled* check box.

Potential issues


There are situations in which FlowForce Server can disable triggers at runtime to avoid some issues. For example, if FlowForce Server has trouble using credentials, it may disable a trigger to avoid locking the credential. Note trigger behavior in this respect:




- Timer triggers do not require logon to work. Therefore, logon errors are detected only when the job is started.
- Watch triggers (file system and HTTP triggers) do require logon to work, as they have to access files in the context of the user.

Overall, FlowForce Server never disables triggers completely. FlowForce Server tries to avoid swamping the system with failed logon attempts that would lead to account lock-out and other issues. Normally, no action is required for triggers if the credentials have not changed, but you can explicitly reactivate the trigger by re-saving the credentials.

Manage a trigger

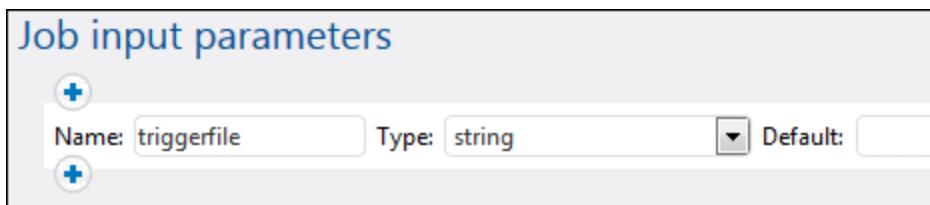
Use the buttons to the right of a trigger to manage the trigger (*see below*).

	Set the value of a trigger's parameter (e.g., <i>Start</i>).
---	---

	Delete a trigger or delete the value of the trigger's parameter (e.g., <i>Repeat</i>).
	Duplicate a trigger.
	Undo the previous delete action.

The triggerfile parameter

Whenever you create a file system or HTTP trigger, FlowForce Server automatically adds a `triggerfile` input parameter to the job (see *screenshot below*). When the job runs, FlowForce Server sets this parameter to the file that triggers the job (file system triggers) and the name of the temporary file that contains the downloaded content of the polled URI (HTTP triggers).



Job input parameters

+
Name: Type: Default:

+
+

You can pass the value of the `triggerfile` parameter as an input value in any subsequent steps of the job. This way, you can use or process the triggering file as required. By default, the `triggerfile` parameter contains the absolute path of the triggering file. To extract portions of the path, use the [file system expression](#)²¹⁴ functions. See an example of a job that uses the `triggerfile` parameter in [Creating a Directory Polling Job](#)³¹⁹.

4.5.1 Timer Triggers

Time triggers allow you to schedule jobs to start at a specific time and run for a specific time interval. Time triggers have flexible recurring options: e.g., they can be set to run daily, weekly, on specific days of the week or month. The screenshot below illustrates a sample timer trigger.

Triggers

Run: on days of week every 1 week(s)

Days of week:	Mon	Tue	Wed	Thu	Fri	Sat	Sun
	<input type="checkbox"/> all	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Repeat: every 60 minutes the whole day, or from 08:00:00 to 20:00:00

Start: 2018-01-01 01:00:00

Expires: 2018-12-31 01:00:00

Time zone: Europe/Berlin

enabled

The subsection below explains how to define timer settings.

Timer trigger overview

Timer triggers have the following parameters: *Run*, *Repeat*, *Start*, *Expires*, *Time Zone*, and *Enabled* (see descriptions below).

Run

Defines whether the trigger should fire once or every *N* number of days. The following options are available: Once, daily, on days of week, on days of months, on days in weeks of months.

Repeat

Defines the *Repeat* options of the trigger. The repeat events occur on days specified in the **Run** drop-down list (see previous parameter). The *every* field defines the repeat frequency in minutes. The *from* and *to* fields define the time range between repeat events.

Start

Defines the trigger's starting date and time. The start date and time entries are mandatory if you have selected **Once** from the **Run** drop-down list. When you click in the date field, a pop-up calendar opens, which allows you to select the start date. You can also type in the date manually.

Expires

Defines the expiry date and time of the trigger.

Time zone

Defines the time zone of the start and expiry date and time. [The default time zone](#)¹¹⁸ is defined in the server administration settings.

Enabled

The *Enabled* check box allows you to enable or disable the trigger. This option is useful when you create and test new jobs.

4.5.2 File System Triggers

File system triggers start jobs when a change is detected in a file or folder (e.g., a new file has been added). Note that deleted files cannot be monitored. You can configure the directory polling interval (e.g., every 60 seconds) and optionally set the start and expiry date of the trigger. You can also use wildcards to filter specific files of the directory. The screenshot below illustrates a sample file system trigger.

The screenshot shows a configuration window titled "Triggers". It contains the following fields and controls:

- Check:** A dropdown menu set to "Modified Date".
- of file or directory:** A text input field containing "C:\data".
- polling interval:** A text input field containing "60", followed by the text "seconds".
- Wait:** A text input field containing "0", followed by the text "seconds for settle".
- Start:** A button with a "+" sign.
- Expires:** A button with a "+" sign.
- Time zone:** A dropdown menu set to "Europe/Berlin".
- enabled:** A checked checkbox.

The subsection below explains how to define the settings of file system triggers.

File system trigger overview

File system triggers have the following parameters: *Check*, *Of file or directory*, *Polling interval*, *Wait N seconds for settle*, *Start*, *Expires*, *Time zone*, and *Enabled* (see descriptions below).

☐ Check

Specifies how the trigger should poll the directory or file. Valid options are listed below:

- *Newly created:* The trigger fires whenever any new files or directories are added to the specified directory. In terms of server load, this option requires the least server resources. When a new trigger is added and the job is saved, any existing files in that directory will be considered as newly created, and the job will be executed for each. If a file is deleted and then added again later, the job will be executed for it again. Note that this will happen only if the polling interval has already elapsed since the deletion. The trigger also fires if a file has been renamed. This trigger does not fire if any files from the polled directory are subsequently modified. If you need such behavior, see *Modified Date* below.
- *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 or renamed, the trigger fires. This option takes slightly more resources from the server than the previous one.
- *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, 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 or renamed, the trigger also fires.

Of file or directory

You can choose any path, in which you would like to check changes. You can also use wildcards to specify directories for a file system trigger. For example, you can specify the following path: `C:\inbound\A*\B*`. FlowForce will scan all the subdirectories of `C:\inbound`: It will first scan its child directories starting with `A` and then scan all the child directories of `A` for directories/files starting with `B`.

Polling interval

Specifies the frequency (in seconds), with which the directory will be polled. The default value is 60 seconds. The minimum value is 1 second.

Wait N seconds for settle

The server will wait *N* seconds before checking the file. If the file is still in the specified location and has not changed during the settle period, the job will start. Otherwise, the server will wait again for the specified period and then check again if the file has changed since the last check. This option allows FlowForce Server to wait until the file has been fully written and ensure that the file is not being edited/changed by anybody.

Start

Defines the trigger's starting date and time. This is an optional field. When you click in the date field, a pop-up calendar opens, which allows you to select the start date. You can also type in the date manually.

Expires

Defines the date and time when the trigger expires.

Time zone

Defines the time zone of the start and expiry date and time. [The default time zone](#)¹¹⁸ is defined in the server administration settings.

Enabled

The *Enabled* check box allows you to enable or disable the trigger. This option can be useful when you create and test new jobs.

4.5.3 HTTP Triggers

HTTP triggers allow you to monitor a URI (Uniform Resource Identifier) for changes. Specifically, you can poll the `Last-Modified` and `Content-MD5` HTTP header fields for changes. You can configure the polling interval (e.g, every 60 seconds) and optionally set the start and expiry date of the trigger. The screenshot below illustrates a sample HTTP trigger.

The screenshot shows a configuration window titled "Triggers". It contains several input fields and controls: a "Check" dropdown menu set to "Content", an "of URI" text field containing "http://example.org", a "polling interval" field with "60" and "seconds", a "Wait" field with "0" and "seconds for settle", "Start:" and "Expires:" labels each followed by a plus sign in a circle, a "Time zone" dropdown menu set to "Europe/Berlin", and an "enabled" checkbox that is checked.

The subsection below explains how to define the settings of HTTP triggers.

HTTP trigger overview

HTTP triggers have the following parameters: *Check*, *Of URI*, *Polling interval*, *Wait N seconds for settle*, *Start*, *Expires*, *Time zone*, *Enabled* (see descriptions below).

[-] Check

Specifies how the trigger should poll the URI. The following options are available:

- *HTTP Header Date* instructs the system to check the `Last-Modified` HTTP header. If the `Last-Modified` HTTP header is missing, the `Content-MD5` header is checked (see *next option*).
- *Content* instructs the system to check the optional `Content-MD5` HTTP header. This is a 128-bit digest used as a message integrity check. If the MD5 header has changed after the polling interval has passed, the trigger fires. If the header is not provided by the server, the content is retrieved and hashed locally.

[-] Of URI

In this field, you need to specify the URI you would like to check for changes.

[-] Polling interval

Specifies the frequency in seconds, with which the URI will be polled.

[-] Wait N seconds for settle

The server will wait *N* seconds before checking the file. If the file is still in the specified location and has not changed during the settle period, the job will start. Otherwise, the server will wait again for the specified period and then check again if the file has changed since the last check. This option allows FlowForce Server to wait until the file has been fully written and ensure that the file is not being edited/changed by anybody.

[-] Start

Defines the trigger's starting date and time. This is an optional field. When you click in the date field, a pop-up calendar opens, which allows you to select the start date. You can also type in the date manually.

[-] Expires

Defines the date and time when the trigger expires.

[-] Time zone

Defines the time zone applicable to the start and expiry date and time. [The default time zone](#)¹¹⁸ is defined in the server administration settings.

Enabled

The *Enabled* check box allows you to enable or disable the trigger. This option is useful when you create and test new jobs.

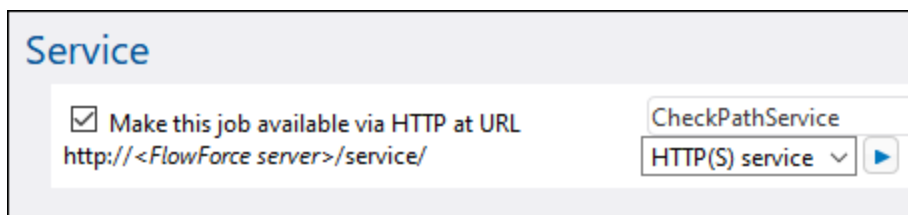
4.6 Jobs as Web Services

FlowForce Server allows you to configure jobs as Web services. FlowForce Server Advanced Edition also allows you to configure AS2 messages as Web services. Jobs configured as Web services are primarily meant to be accessed programmatically. For testing and debugging purposes, you can also invoke such jobs from a browser.

Configure a job as a Web service


To make a job available as a Web service, take the following steps:

1. [Create a new job](#)¹²⁸ or open an existing one for editing.
2. Select the *Make this job available via HTTP at URL* check box (see screenshot below).



3. Type the name of the Web service in the text field.

Jobs configured as services remain active as long as FlowForce server is running.

Note: The  button (**Call Web Service**) is available only if you have set the *Host name* field of the *FlowForce Server* service from the [Setup page](#)⁴⁷. Clicking this button invokes the Web service in a new browser window. If you have not configured a host name for FlowForce Server, the button is not displayed, but you can still call the Web service by typing its URL manually in the browser's address bar.

Possible outcomes

When the Web service is invoked, FlowForce Server runs the job execution steps specified and returns one of the following:

- The first result file of the last step if the job produces a result file.
- The standard output of the last step if no result files are produced (this might be the case when you are working with command line output).

A valid result is returned with an 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 are also returned as `text/plain`. The result is returned as the response body.

Execution errors are reported as an HTTP 5xx status with a generic error message. For further information, check the FlowForce Server [log](#)¹⁰⁴.

For examples of Web services, see [Expose a Job as a Web Service](#)³³¹.

It is possible to configure FlowForce to return a result before all the job steps are executed. This is particularly useful if the job invoked as a service takes a long time. The early result could be treated by the caller as a confirmation that the task has been accepted by FlowForce Server for processing. For details, see [Postponed](#)

Steps ¹⁴⁰

View current Web services

To view all current Web services, do one of the following:

- Go to the **Home** page and click **Show all active triggers and services**. See also [Active timers](#) ⁸⁷.
- Access the following URL from your browser: `http://[FlowForceServer]:[ServerPort]/service/*`.

In the URL above, `[FlowForceServer]` and `[ServerPort]` refer to the network address and port where FlowForce Server is listening. By default, FlowForce Server runs on <http://localhost:4646> (assuming you are accessing it from the same machine). The server name and port are defined on the **Administration** page. For more information, see [Defining the Network Settings](#) ⁴⁷.

Web service parameters

When you expose a job as a Web service, all job parameters automatically become parameters for the service. A job parameter must have a default.

When the service is invoked, FlowForce Server verifies the parameters supplied in the request against those defined in the job. If parameter validation fails, FlowForce Server returns a 5xx HTTP status. In this case, FlowForce Server also displays an HTML parameter form for debugging and testing purposes.

For each parameter of type `stream`, the **Browse** button becomes available on the page, and you can use the button to upload the file required as a parameter.

To display the testing HTML form unconditionally, supply the built-in parameter `showForm` in the request (with any value).

To call a FlowForce Web service with parameters, a client can use one the following options:

1. For parameters of simple type such as strings or numbers, a client can supply them in the URL of the GET or POST request. For an example, see [Expose a Job as a Web Service](#) ³³¹.
2. In the case of POST requests, a client can additionally provide parameters as `multipart/form-data` or as `application/x-www-form-urlencoded`. If the parameter is of type `stream` in FlowForce, then the client must provide them as `multipart/form-data`. For such parameters, the browser test HTML form displays the **Browse** button next to the corresponding parameter.
3. The client call can also include arbitrary content in the body of the POST request (this specifically refers to content such as JSON or XML, posted not as a parameter but as the body of the HTTP request). In order for this to be possible, the FlowForce job must contain a *single* parameter of type `stream`. If you need additional non-stream parameters, these must be supplied in the POST URL. However, only one parameter of type `stream` must be defined in FlowForce; other parameters must be of non-stream type. When these conditions are met, the request body will be treated as data for the `stream` parameter. No other configuration is required. For an example, see [Post JSON to FlowForce Web Service](#) ³³⁹.

Web service authentication

By default, FlowForce Server uses HTTP Basic authentication to authenticate clients calling a Web service. User credentials are checked against the FlowForce Server user database (the same user name and password used to log on to FlowForce Server Web administration interface).


Authentication Required

http://127.0.0.1:4646 is requesting your username and password. The site says: "Altova FlowForce"

User Name: john_doe

Password: ●●●●●●

OK Cancel

To make a Web service available without credentials, grant the *Use Service* permission to the default  **anonymous** user (see also [How Permissions Work](#)⁹²). 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.

If you supply invalid credentials, the request interface returns 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 also returns an HTTP status of 401.

If you supply valid credentials, but the authenticated user is not granted the *Use Service* permission on this service, the request interface will return an HTTP 4xx failure status. If you try accessing a service that does not exist, you will get the HTTP 4xx failure status.

Optionally, domain authentication can also be configured, in addition to HTTP basic authentication. For information about how to configure it, see [Changing the Directory Service Settings](#)¹¹⁹. Once domain authentication has been configured, users will be able to access Web services exposed by FlowForce Server, provided that they supply a valid username and password for the respective domain. Importantly, for Active Directory, the username must contain the prefix `NT/` and must include the domain name, for example: `NT/john.doe@my.domain.com`.

Authentication Required

http://127.0.0.1:4646 is requesting your username and password. The site says: "Altova FlowForce"

User Name: NT/john.doe@my.domain.com

Password: ●●●●●●

OK Cancel

Queue settings

Service URL requests are a particular kind of trigger, and are therefore subject to the same queue constraints once the connection has been established. See [Defining Queue Settings](#)¹⁶⁶.

Configuring the maximum size of the HTTP request body

A default limit exists in FlowForce Server that establishes the maximum size of the HTTP request body, which is around 100 MB. When a caller posts HTTP requests to FlowForce jobs exposed as Web services and the HTTP request body exceeds this limit, FlowForce Server may return an error with the following text:

```
The entity sent with the request exceeds the maximum allowed bytes.
```

To accept requests of larger sizes:

1. Open the [flowforce.ini](#)⁶⁷ file in a text editor.
2. Add the option `max_request_body_size` to the **[Listen]** or **[ListenSSL]** section and set it to the maximum number of bytes that should be allowed.

For example, in order to enable a maximum size of 500 MB, your **flowforce.ini** file could look like this:

```
[Listen]
active=1
host=0.0.0.0
port=4646
hostname=somehost.example.org
max_request_body_size=500000000
```

For more information about the .ini file, see [Configuration File Reference](#)⁶⁷.

Reconfiguring FlowForce Server pool threads

If you expect a large number of parallel HTTP service requests (for example, 20 or more at a time), it is possible to reconfigure the server for a slightly larger number of pool threads.

1. Open the [flowforce.ini](#)⁶⁷ file in a text editor.
2. Add the option `thread_pool` to the **[Listen]** or **[ListenSSL]** section of the .ini file and set it to a value larger than 20.
3. Restart the service.

Note: It is a good idea to have two separate **[Listen]** sections, one for FlowForce Web Server (which doesn't require that many pool threads) and the other for all other requests (on a different port, preferably). Otherwise, FlowForce Web Server will be competing with all the other HTTP requests for pool threads.

4.7 Credentials

A credential object is a piece of data that stores authentication information such as usernames and passwords, certificates, API keys, tokens, etc. that are used to securely manage and transmit authentication details and access different services and resources.

Supported protocols

FlowForce Server supports the following protocols:

- FTP
- FTPS
- HTTP
- SFTP (*Advanced Edition*)

Note: In order to use FTPS, you need to (i) use the [/system/ftp](#)²⁴⁵ functions and (ii) set the `Use SSL/TLS encryption parameter to Explicit with encrypted with command channel OR Explicit with encrypted with command and data channel.`

If you have licensed MapForce and MapForce Server to run mappings as FlowForce Server jobs, you can create credential objects not only in FlowForce Server, but also in MapForce at mapping-design time. You can optionally deploy credentials created in MapForce to FlowForce Server, together with the mapping where they belong or as individual objects. A deployed credential does not necessarily have to store any sensitive data such as a username and password.

For information about creating credentials in MapForce and deploying them to FlowForce Server, refer to the MapForce documentation (<https://www.altova.com/documentation>). For details about setting or overriding credentials in mapping jobs, see [Credentials in Mapping Functions](#)⁴²¹.

Important points

Users can refer to credentials from jobs only if they have the relevant permissions granted. To make credentials from a specific container accessible to a user or role, administrators must grant the *Credentials - Use* permission to that user or role (see [How Permissions Work](#)⁹²).

Because 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's database file, see [FlowForce Server Application Data](#)⁶⁴.

Credential types

FlowForce Server supports the following types of credentials:

- *Password* (the combination of a username and password)
- *OAuth 2.0* (*Advanced Edition*)
- *SSH Key* (*Advanced Edition*)

In FlowForce Server, you can define credentials every time you create a new job (i.e., local credentials) or create standalone (i.e., reusable) credential objects. In the case of standalone credentials, when you create a job, you can refer to the credentials defined previously instead of entering them again. Standalone credentials are also convenient, because you can update them easily in one place when they change, and this change will affect all jobs that use that credential reference.

Password credentials

Password credentials are required by each job; they make it possible to run the job as a particular operating system user. Specifically, when you create a job in FlowForce Server, you must supply the credentials of the user account with which the job must be executed. Note that if the user account does not have sufficient rights on the operating system, the job cannot execute successfully. Password credentials are also required when you call built-in [FTP](#)²⁴⁵ functions, where authorization to an FTP server is required. [File watch triggers](#)¹⁵⁴ also require password credentials.

For details about password credentials, see [Credential Type: Password](#)¹⁶³.

OAuth 2.0 credentials (Advanced Edition)

OAuth 2.0 credentials are necessary in jobs that call Web services where OAuth 2.0 is required. OAuth 2.0 credentials can be defined only as standalone (not local) credentials and subsequently be referenced from any jobs where they are required.

SSH Key credentials (Advanced Edition)

An SSH Key is a credential type that is valid only for SFTP. The main principle of this type is based on the usage of a unique pair of keys: the public key encrypts the message, the server receives it, and the private key helps decrypt this message. The credential can be used to authenticate SFTP connections.

4.7.1 Credential Type: Password

To create a password credential, navigate to the container in which you want to store the credential, click **Create | Create Credential**, and fill in the credential fields (*described below*). Before creating credential objects, make sure that you have the *Container - Read, Write* and *Configuration - Read, Write* permissions granted on the container where you want to store the credentials.

☐ Credential name

Mandatory field. This is the name by which the credential is identified in FlowForce Server.

☐ Credential description

An optional description that provides more information about this credential.

☐ Credential type (Advanced Edition)

Allows you to choose a credential type: *Password*, *OAuth 2.0*, or *SSH Key*.

☐ User name

Mandatory field. The name of the user associated with this credential. For example, if the credential is used to identify a user account on the Windows operating system, enter the Windows user account name. To specify a user name in a Windows domain, use the *username@domain* format.

If you want to use the credential for HTTP or FTP (*see Allow Usage For below*), this may also be an HTTP or FTP user name.

☐ Password

Specifies the credential's password. The password may be an empty string if the context where it will be used requires only a username without a password.

☐ Allow usage for

You can use a credential for HTTP, FTP, SSH/SFTP, and for job execution.

HTTP

Select this check box if the credential is referenced in jobs that call Web services which require basic HTTP authentication.

FTP

Select this check box if the credential is referenced in jobs that connect to FTP servers using [/system/ftp](#)²⁴⁵ functions.

SSH/SFTP (Advanced Edition)

Select this check box if the credential is referenced in jobs that connect to SFTP servers using the `/system/sftp` functions.

Job execution

Select this check box if the credential identifies an operating system user account. In order to run successfully, any job requires a credential with this usage enabled. Ensure that the user account identified by the credentials has sufficient rights on the operating system. For example, if credentials are going to be referred to in a job that writes to a directory, the user account must have rights to write to that directory.

4.7.2 Refer to Credentials from Jobs

Assuming that you have been granted the required [permissions](#)⁹² to use a credential object, you can refer to it from various contexts where credentials are necessary, for example:

- You have created a credential that identifies a user account on the operating system where FlowForce Server runs (that is, the option **Allow usage for job execution** is enabled). You may subsequently refer to this credential from multiple jobs. This example is described below.
- You have created a credential that identifies an FTP username and password (that is, the option **Allow usage for FTP** is enabled). You may refer such a credential from any job that calls an [FTP](#)²⁴⁵ function.

The following example is illustrative of the common case where you need to refer to password credentials that identify a user account on the operating system where FlowForce Server runs:

1. Create a credential where the option **Allow usage for job execution** is enabled, as illustrated in [Defining Credentials](#)¹⁶³.
2. Create a new job or edit an existing one.
3. Under "Credential", click **Select existing credential**, and browse for the credential record defined previously.

Credential

Run job using credential: Select existing credential: /public/my.credentials
 Define local credential:

Queue settings

Minimum time between runs: 0 seconds
Maximum parallel runs: 1 instances

Save

public

Name

- my.credentials

If you have jobs that contain credential records defined locally, you can refer to them as if they were credentials objects themselves, for example:

Credential

Run job using credential: Select existing credential: /public/JobWithLocalCredentials
 Define local credential:

Queue settings

Minimum time between runs: 0 seconds
Maximum parallel runs: 1 instances

Save

public

Name

- JobWithLocalCredentials
- my.credentials

In this case, the credentials of the embedded job (the one that has local credentials) will be used as credentials of the main job. Note that credentials are linked, not copied: if you change the locally defined credentials in the embedded job, they will be propagated to the main job as well.

4.8 Queue Settings

Queue settings enable you to control the usage of server resources more efficiently. For example, through queue configuration, you can limit the number of job instances running in parallel at any given moment.

An execution queue is a processor of jobs. It controls how job instances run. In order to run, every job instance is assigned to a target execution queue. The queue controls how many job instances (of all the jobs assigned to the queue) can be running at any one time and the delay between runs. By default, the queue settings are local to the job, but you can also define queues as standalone objects shared by multiple jobs. When multiple jobs are assigned to the same execution queue, they will share that queue for executing.

Queues benefit from the same security access mechanism as other FlowForce Server configuration objects. Namely, a user must have the *Define execution queues* privilege in order to create queues (see also [Define Users and Roles](#)⁷²). In addition, users can view queues and assign jobs to queues if they have appropriate container permissions (see also [How Permissions Work](#)⁹²). By default, any authenticated user gets the *Queue - Use* permission, which means they can assign jobs to queues. To restrict access to queues, navigate to the container where the queue is defined and change the permission of the container to *Queue - No access* for the role `authenticated`. Next, assign the permission *Queue - Use* to any roles or users that you need. For more information, see [Restricting Access to the /public Container](#)¹⁰².

Global vs local queues

You can create a queue as a standalone object (global) or within the framework of a particular job (local). Local queues do not support distributed processing (clusters). The queue must be created as a standalone object (external to the job) in order to benefit from distributed processing. Distributed processing is supported only in Advanced Edition. For information about creating standalone and local queues, see the subsections below.

Create global queues

To create a queue as a standalone object, take the steps below:

1. Open the Configuration page and navigate to the container where you want to create the queue.
2. Click **Create** and select **Create Queue** (*screenshot below*).

3. Enter a queue name, and, optionally, a description.
4. Configure the relevant settings. For details, see *Queue Settings* below.
5. Click **Save**.

Queue settings

The queue-configuration settings are listed below.

Queue name A name that identifies the queue. This is a mandatory field. It may contain only letters, digits, single spaces, and the underscore (_), dash (-), and full stop (.) characters. It may not start or end with spaces. This field is applicable only if the queue is defined as a standalone (not local) queue.

Queue description Optional description of the queue object. This field is applicable only if the queue is defined as a standalone (not local) queue.

Run on (Advanced Edition) Specifies how all job instances from this queue are to be run:

- *Master or any worker:* Job instances that are part of this queue will run on the master or worker machines, depending on available server cores.
- *Master only:* Job instances will run only on the master machine.
- *Any worker only:* Job instances will run on any available worker but never on the master.

Minimum time between runs An execution queue provides execution slots. Each slot will execute job instances sequentially.

The *Minimum time between runs* setting keeps a slot marked as occupied for a short duration after a job instance has finished, so it will not pick up the next job instance right away. This

reduces maximum throughput for this execution queue, but provides CPU time for other execution queues and other processes on the same machine.

Maximum parallel runs



This option defines the number of execution slots available on the queue. Each slot executes job instances sequentially, so the setting determines how many instances of the same job may be executed in parallel in the current queue. Note, however, that the number of instances you allow to run in parallel will compete over available machine resources. Increasing this value could be acceptable for queues that process lightweight jobs that do not perform intensive I/O operations or need significant CPU time. The default value (1 instance) is suitable for queues that process resource-intensive jobs, which helps ensure that only one such heavyweight job instance is processed at a time.

This option does not affect the number of maximum parallel HTTP requests accepted by FlowForce Server (such as those from clients that invoke jobs exposed as Web services). For details, see [Reconfiguring FlowForce Server pool threads](#)¹⁶¹.

Multiple sets of queue settings (Advanced Edition)

You can define multiple sets of queue settings, each with different processing requirements, by clicking the



button. To change the priority of a specific set of settings, click the **Move up**  or **Move down**  buttons. For example, you can define a rule for the case in which only the master is available and another rule for the case in which both the master and its workers are available. This enables you to create a fallback mechanism for the queue, depending on the state of the cluster at a given time. When processing queues, FlowForce Server constantly monitors the state of the cluster and knows if any worker is unavailable. So, if you defined multiple queue settings rules, FlowForce Server evaluates them in the defined order, from top to bottom, and picks the first rule that has at least one cluster member assigned according to the *Run On* setting.

Example (Advanced Edition)

As an example, let's consider a setup where the cluster includes one master and four worker machines. The queue settings are defined as shown below:

The screenshot shows a 'Queue settings' window with two distinct queue configurations. Each configuration is a card with a plus sign on the left and navigation icons (up, down, copy, delete) on the right. The first configuration is titled 'any worker only' and has 'Minimum time between runs' set to 0 seconds and 'Maximum parallel runs' set to 4 instances. The second configuration is titled 'master only' and has 'Minimum time between runs' set to 5 seconds and 'Maximum parallel runs' set to 1 instance. A 'Save' button is located at the bottom left of the window.

With the configuration illustrated above, FlowForce would process the queue as follows, depending on the current state of the cluster:

- If all workers are available, the top rule will apply. Namely, up to 16 job instances are allowed to run simultaneously (4 instances for each worker). The minimum time between runs is 0 seconds.
- If only three workers are available, the top rule will still apply. Namely, up to 12 job instances are allowed to run simultaneously, and the minimum time between runs is 0 seconds.
- If no workers are available, the second rule will apply. Namely, only 1 instance may run at a given time, and the minimum time between runs is 5 seconds.

This kind of configuration makes execution still possible in the absence of workers. Notice that the *master only* rule is stricter (1 instance only, and 5 seconds delay between runs) so as not to take away too much processing power from the master machine when all the workers fail.

Assign jobs to queues

Once you have configured the queue, you will need to assign a job to it on the job configuration page. In order to do this, take the steps below:

1. Open the configuration of the job that you wish to assign to the queue.
2. Navigate to the queue settings at the bottom of the page.
3. Select the *Select existing queue* option and provide the path to the desired queue object (*screenshot below*).

Queue settings

Run job using queue: Select existing queue: /public/Queue1

Define local queue:

Save

Altova FI

public

Name

Name
Queue1
Queue2

Define local queues

As an alternative to creating standalone queues, you can define the queue settings locally inside the job. To do this, select the *Define local queue* option from the job configuration page and specify your queue preferences. The image below illustrates the default queue settings. With the *Define local queue* option selected, FlowForce Server will assign, at job runtime, the instances of this job to a default queue, with the local settings you specify.

Queue settings

Run job using queue: Select existing queue:

Define local queue:

Minimum time between runs: 0 seconds

Maximum parallel runs: 1 instances

For details about the *Minimum time between runs* and *Maximum parallel runs* properties, see *Queue Settings* above.

4.9 Expressions

FlowForce expressions represent custom code that can be computed and executed by FlowForce Server when a job runs. You can think of FlowForce expressions as a basic scripting language understood by FlowForce that helps you "glue together" multiple steps within a job. FlowForce expressions are typically necessary in the following contexts:

- In parameters of built-in functions (that is, you can write or embed expressions in input fields in the job configuration page). Here are a just a few examples:
 - Change the data type of the result returned by the execution step
 - Pick a specific value from a result that returns a list of values
 - Concatenate multiple values in order to produce a string.
- In "when" steps, to produce conditional statements. This enables you to execute the step if the expression you provide evaluates to Boolean **true**.
- In "for-each" steps. "For-each" steps enable you to loop through a sequence of items, where the sequence is defined by an expression.

This section describes the concepts that will help you build FlowForce expressions for scenarios such as the ones listed above.

4.9.1 Compute an Expression

A simple way to test FlowForce expressions before embedding them in jobs is to create an execution step that calls the `/system/compute` function. For a step-by-step example, see [Creating a "Hello, World!" Job](#).

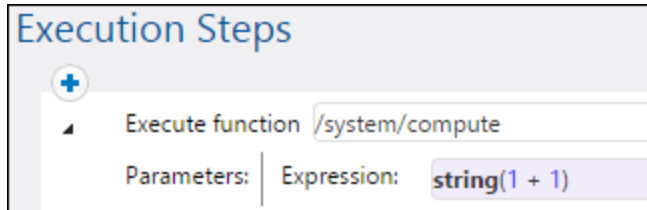
The `/system/compute` function evaluates the value of the **Expression** parameter and returns the computed result. Importantly, this function has no defined return type. The actual type depends on the expression being computed. For example, if you pass to this function the expression `1+1`, the function returns the numeric value `2`. However, if you pass to this function the expression `'1+1'`, it returns the string value `1+1`.

To understand this concept better, create a step that calls the `/system/compute` function and enter "1+1" in the expression field. Make sure to declare the job return type as "string", as shown below.

The screenshot displays the 'Execution Steps' configuration page. A single step is added, titled 'Execute function /system/compute'. The 'Parameters' section shows the 'Expression' field containing '1 + 1'. Below the parameters, there is an option to 'Assign this step's result to' a field labeled 'name', followed by a red 'i' icon and the text 'as T0'. At the bottom of the step configuration, there are four buttons: 'new Execution step', 'new Choose step', 'new For-each step', and 'new error/success handling step'. Below the step configuration, the 'Execution Result' section is visible, with a dropdown menu for 'Declare return type as:' currently set to 'string'.

When you attempt to save the job, FlowForce displays a "Types string and number do not match" error. This error happens because the computed expression is a number, whereas the return type of the job is declared as a string value.

To fix the typing problem, either change the return type of the job to "number" or convert the number to a string. The example below calls the FlowForce expression function `string` which converts a number into a string value.



When you need to compute an expression and return the value as string, you can alternatively use the [/system/compute-string](#) ²³⁹ function. In this case, note that the expression part must be delimited from the string with curly braces (see [Embedding Expressions in String Fields](#) ¹⁷⁵).

4.9.2 Expression Language Rules

To avoid errors in FlowForce expressions, follow these rules:

- Use only allowed or declared values.
- To use a string literally, enclose it within single quotes.
- To embed an expression in a string field, enclose it within curly braces, that is, the { and } characters.
- The expression must produce a data type which is meaningful in the field where the expression was entered.

Let's now have a look at these rules in more detail.

Rule #1: Use only allowed or declared values

The following constructs are allowed in FlowForce expressions:

- FlowForce expression functions (for complete reference, see [Expression Functions](#) ¹⁸⁰)
- FlowForce operators (see [Operators](#) ¹⁷⁸)
- Numeric values
- String values
- Previously declared variables

When you type text inside a field which allows FlowForce expressions, a real-time syntax check takes place. If the syntax is not correct, FlowForce highlights in red the offending characters. Below is an example of a syntax validation error:



The error occurs because neither `source` nor `target` have been declared in the job, so FlowForce cannot interpret the expression. The problem can be fixed by declaring these values (for example, as job input parameters):

The screenshot shows the 'Job Input Parameters' section with two parameters: 'source' and 'target', both of type 'string'. Below this is the 'Execution Steps' section, which includes a step 'Execute function /system/filesystem/copy'. The parameters for this step are 'Source: {source}' and 'Target: {target}'.

Rule #2: Enclose strings in single quotes

If you need to use a string literally, enclose it within single quotes. Otherwise, the expression might produce undesired results or validation will fail. Consider the following examples:

Expression	Will be evaluated as...	Explanation
<code>1+1</code>	2	The data type of the value is numeric.
<code>'1+1'</code>	1+1	The data type of the value is string.
<code>1+1==2</code>	true	The data type of the value is Boolean.

When you need to convert values from one data type to another, use the FlowForce expression functions (see also Rule #4).

Rule #3: Use curly braces in string fields

If you want to embed an expression inside a string field, enclose the expression within curly braces. In the example below, curly braces delimit the expression `instance-id()` (which is a FlowForce expression function) from the rest of the string.

The screenshot shows the 'Execute function /system/mail/send' step. The 'Parameters' section includes 'From:' (with a plus icon), 'To: someone@example.org', and 'Subject: Job {instance-id()} has completed.'.

If the entire field is of type "as expression", do not use curly braces. For example, the **Expression** parameter of the [system/compute](#)²³⁷ built-in function has this type. Below is an example of a correct value for this field (notice no curly braces are used):

Typing curly braces inside the expression field would trigger a syntax error:

See also [Embedding Expressions in String Fields](#)¹⁷⁵.

Rule #4: Use the correct data type

Finally, be aware that FlowForce performs data type checks when you save a job. An error will occur if the expression entered in a field does not match the data type expected by the field. You can see the data type expected by each field displayed on the right side of it, for example:

Therefore, an expression such as `1+1` is not a valid in a string field, because it is implicitly evaluated as numeric. On the other hand, the expression `'1+1'` is valid in a string field. Consider the following examples:

Expression	Will be evaluated as...	Explanation
<code>1/4</code>	0.25 (as Number)	The data type of the value is numeric. Use this expression in a field or context which expects a numeric value; otherwise, job validation would fail.
<code>1+1==2</code>	true (as Boolean)	The data type of the value is Boolean. Use this expression in a field or context which expects a Boolean value; otherwise, job validation would fail.
<code>'apple'</code>	apple (as String)	The data type of the value is string. Use this expression in a field or context which expects a string value; otherwise, job validation would fail.
<code>concat('1','2','3')</code>	123 (as String)	The data type of the value is string.

Expression	Will be evaluated as...	Explanation
		Use this expression in a field or context which expects a string value; otherwise, job validation would fail.
1+'apple'	-	This expression is not valid, and FlowForce will return an error when you attempt to save the job. Evaluation cannot take place because two different data types (string and numeric) are being compared.
{content(stdout(result))}	[...] (as String)	<p>This expression uses two nested expression functions:</p> <ul style="list-style-type: none"> The function <code>stdout</code> gets the standard output of a shell command, as stream. The function <code>content</code> converts the stream value to a string. <p>Although the expression is correct, the job will validate successfully only when the following is true:</p> <ul style="list-style-type: none"> The value "result" has been previously declared. The value "result" actually contains the standard output of a shell command. The expression is embedded into a string field. <p>See also Calling Expression Functions ¹⁷⁶.</p>

4.9.3 Embed Expressions in String Fields

To use a FlowForce expression in a string field, enclose the expression within curly braces, that is, the "{" and "}" characters. The expression part of a string field normally has a light purple background, which helps you distinguish the expression part from the rest of the string, for example:

Execute function `/system/mail/send`

Parameters:	From: +
	To: someone@example.org
	Subject: Job {instance-id()} has completed.

In a string field, only the expression enclosed within curly braces will be treated by FlowForce as an expression. If you want FlowForce to interpret the "{" and "}" characters literally, write double braces instead of a single brace. Consider the following cases:

A string field with the following value...	Will be evaluated as...	Explanation
<code>echo Hello, World!</code>	<code>echo Hello, World!</code>	The string does not use any curly braces (it does not contain an embedded expression), so it is evaluated as is.
<code>echo {Hello, World}</code>	-	The string cannot be evaluated. The embedded expression is not syntactically correct, so FlowForce displays a syntax error.
<code>echo {"Hello, World!"}</code>	<code>echo Hello, World!</code>	The string contains an embedded expression which is syntactically correct. However, the expression is inside a string field, so the evaluation result would be the same if you used no expression at all (see the first example above).
<code>echo {"{Hello, World!"}}</code>	<code>echo {'Hello, World!'}</code>	The string does not contain an expression, since the escape characters <code>{</code> and <code>}</code> were used.

4.9.4 Call Expression Functions

The FlowForce expression language includes a number of functions that can be used to perform basic operations (primarily, handle values returned by execution steps). You can call these functions from any context where FlowForce expressions are valid (for example, by typing them inside text boxes that represent parameters of a function).

FlowForce expression functions should not be confused with the FlowForce built-in functions. Built-in functions are called from FlowForce execution steps (that is, they are executed as steps), while expression functions are called from FlowForce expressions.

As a typical scenario to call expression functions, let's consider the job illustrated below, which consists of two execution steps.

The first step executes a shell command (namely, it outputs the text "Hello, World!"). Notice that the data type returned by this step is "as result". The returned value is declared as **var1**.

The second execution step calls the [/system/compute-string²³⁹](#) built-in function. We called this function in order to convert **var1** to a string. The expression itself is embedded into a string field (which is indicated by the curly braces), and it calls two nested expression functions.

- The function `stdout` returns the standard output of a shell command, as stream.
- The function `content` converts the stream value to a string.

The screenshot shows two execution steps in a configuration window titled "Execution Steps".

- Step 1:** "Execute function /system/shell/commandline". Parameters include "Command: echo Hello, World!", "Abort on error: +", and "Working directory: +". It is assigned to "var1" with the label "as result".
- Step 2:** "Execute function /system/compute-string". Parameters include "Expression: {content(stdout(var1))}". It is assigned to "var2" with the label "as string".

Now that the data type conversion is complete, you can further use the string value **var2** as required by your job processing logic (for example, send it in an email).

For reference to all available expression functions, see [Expression Functions](#)¹⁸⁰.

4.9.5 FlowForce Data Types

FlowForce operates with the data types described below.

string

Represents a string value, for example: 'Hello, World!'.

number

Represents a numeric value, for example: -1, 0, 56, 0.45565.

Boolean

Represents a true or false value.

result

This is an abstract type which represents a result produced by an execution step. An execution step can process MapForce mappings, StyleVision transformation files, shell functions, and others. The result aggregates an exit code, `stdout`, `stderr`, and output files (if applicable).

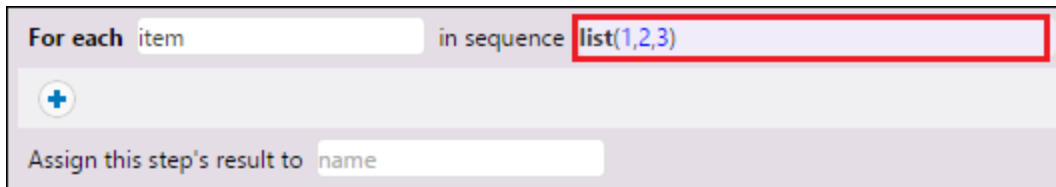
To get access to the result value, give it a name (e.g., `output`) and pass it to the [results](#)²⁰⁵ expression function. This function will convert it to a stream, which you can further process with stream expression functions (see also [Calling Expression Functions](#)¹⁷⁶).

If the execution step runs a shell command, call step-result expression functions to process the output. For example, to return the standard output as a stream, use [stdout\(output\)](#)²⁰². To return the standard error as a stream, use [stderr\(output\)](#)²⁰³. For more information, see [Result Functions](#)²⁰².

item

Sometimes, you need to create expressions that assemble or disassemble lists (see [List Functions](#)²⁰⁹). A list consists of objects of generic type `item`. An item has an abstract data type. You can determine the data type of `item` depending by looking at the type of objects that make up the list (which can be strings, numbers, or even streams). Note that a list can contain only items of the same data type.

The image below illustrates a loop where "item" is of numeric type, since the list itself consists of numeric values.



For a step-by-step example that utilizes lists, see [Copy Files](#)³⁰³.

4.9.6 Operators

To build FlowForce expressions, you can use the operators listed below. Remember that you can test any expression by calling the built-in function [system/compute](#)²³⁷.

Operator	Description	Example
==	Checks if a and b are equal (numerically equal for numbers, code-point equal for strings).	<code>2 + 3 == 5</code> computes true <code>2 + 3 == 4</code> computes false
!=	Checks if a and b are not equal. Note that the following three expressions are equivalent: <ul style="list-style-type: none"> • <code>a != b</code> • <code>not (a == b)</code> • <code>a <> b</code> 	<code>2 + 2 != 5</code> computes true <code>3 + 2 != 5</code> computes false
<	Checks if a is less than b (numerically less for numbers, see below for strings).	<code>4 < 5</code> computes true
<=	Checks if a is less than or equal to b.	<code>5 <= 5</code> computes true
>	Checks if a is greater than b.	<code>5 > 1</code> computes true
>=	Checks if a is greater than or equal to b.	<code>5 >= 5</code> computes true
+	Addition.	<code>1 + 1</code> computes 2
-	Subtraction.	<code>2 - 1</code> computes 1
*	Multiplication.	<code>3 * 2</code> computes 6
/	Division.	<code>6 / 3</code> computes 2

String comparisons are performed as follows:

- The common prefix of the two strings are ignored (evaluated on code points).
- If both remaining strings are non-empty, their first code points are compared numerically.
- Empty strings are less than non-empty strings.

Use parentheses to instruct FlowForce to evaluate the expression inside first. For example:

`2 + 3 * 4` computes 14

`(2 + 3) * 4` computes 20

4.10 Expression Functions

This section provides information about FlowForce expression functions. To understand how to use expressions, see [FlowForce Expressions](#) ¹⁷¹. The list of available expression functions is given below:

- [General Utility Functions](#) ¹⁸⁰
- [Boolean Functions](#) ¹⁸⁵
- [MIME/Stream Expression Functions](#) ¹⁸⁸
- [Result Functions](#) ²⁰²
- [List Functions](#) ²⁰⁹
- [File System Functions](#) ²¹⁴
- [String Functions](#) ²¹⁹
- [Execution State Functions](#) ²²⁹
- [Runtime Information Functions](#) ²³²

4.10.1 General Utility Functions

This section includes general-purpose expression functions that can be used in various contexts.

4.10.1.1 current-message-id

Returns the **Message-ID** header field of an AS2 message. This function must be used in a job that is configured to receive AS2 requests. That is, the check box **Make this job available via HTTP at URL...** must be selected in the job configuration page. Otherwise, this function returns a newly generated **Message-ID** (a new value is generated whenever a new job instance runs and stays constant for that job instance until it ends).

Signature

```
current-message-id() -> string
```

Examples

The following expression produces a filename based on the **Message-ID**. The `substring` function removes the angle brackets (the first and last character) from the **Message-ID**.

```
C:\temp\{substring(current-message-id(), 1, -1)}.msg
```

The following expression does the same as above, and additionally splits the current **Message-ID** apart at character '@' with the help of the `split` function. The `nth` function extracts only the first part—a random hexadecimal value 32 characters long—and uses that as part of a filename.

```
C:\temp\{nth(split(substring(current-message-id(), 1, -1), '@'), 0)}.msg
```

4.10.1.2 new-message-id

Generates and returns a new value for the **Message-ID** header field. You can use this value to populate the header of a MIME message. This function, unlike `current-message-id`, always returns a new **Message-ID**. The **Message-ID** has the following format:

```
'<' UTC timestamp '-' random hex value 32 characters long '@' host name related text '>'
```

For example: <20180306154822808383-5933b654b26c4495bb0b619ab72b3bc6@myservername>.

Signature

```
new-message-id() -> string
```

4.10.1.3 read-lines

Reads the lines from the given file and returns them as a list of strings. The returned strings include the line ends (such as `\n`). You may need to trim each line with the help of the `trim()` function before processing it further, as illustrated in the example below.

Signature

```
read-lines(filename:string, encoding:string="UTF-8") -> list of strings
```

Parameters

Name	Type	Description
filename	string	Specifies the path to a file.
encoding	string	Specifies the encoding to use. The default encoding is 'UTF-8'.

Examples

Let's suppose that you need to process multiple files that reside in multiple directories on the computer where FlowForce Server is installed. All the directory paths are saved as a text file, where each line corresponds to a directory path, for example:

```
C:\FlowForce\Examples\ListDirectories\1
C:\FlowForce\Examples\ListDirectories\2
C:\FlowForce\Examples\ListDirectories\3
```

The job illustrated below consists of two steps. The first step calls the `read-files` function and collects all directory paths from the text file above into a list. The second step iterates through the list of paths and calls

the `list-files` function for each item. Note that the path is also trimmed before processing, to ensure that none of the resulting strings contain spaces or new line characters.

Execution Steps

- Execute function `/system/compute`
 - Parameters: Expression: `read-lines('C:\FlowForce\Examples\ListDirectories\paths.txt')`
 - Assign this step's result to `lines` as T0
- For each `item` in sequence `lines`
 - Execute function `/system/compute`
 - Parameters: Expression: `list-files(trim(item))`
 - Assign this step's result to `name` as T0

If you expose this job as a Web service and access it at the default address and port from a browser, the browser outputs the contents of each directory, as a JSON array, for example:

The screenshot shows a web browser window with the address bar displaying `127.0.0.1:4646/service/ListDirectories`. The page content is a JSON array of file paths, displayed in a tree view. The JSON is as follows:

```

[
  {
    "0": "C:\\FlowForce\\Examples\\ListDirectories\\1\\A.txt",
    "1": "C:\\FlowForce\\Examples\\ListDirectories\\1\\B.txt"
  },
  {
    "0": "C:\\FlowForce\\Examples\\ListDirectories\\2\\C.txt",
    "1": "C:\\FlowForce\\Examples\\ListDirectories\\2\\D.txt"
  },
  {
    "0": "C:\\FlowForce\\Examples\\ListDirectories\\3\\E.txt",
    "1": "C:\\FlowForce\\Examples\\ListDirectories\\3\\F.txt"
  }
]

```

4.10.1.4 is-file

Returns **true** if the function `as-file` would return the name of an existing file, and **false** if `as-file` would create a temporary file.

For example, it returns **true** if the stream was created from a file using the `stream-open` function or returned from a mapping. If the stream is not served from a file or it is a file but a temporary one, this function returns **false**.

Signature

```
is-file(s:stream) -> Boolean
```

Parameters

Name	Type	Description
s	<code>stream</code>	Specifies the input stream.

4.10.1.5 content

Converts the contents of a stream in the specified encoding to a string.

Signature

```
content(stream:stream, encoding:string="UTF-8") -> string
```

Parameters

Name	Type	Description
stream	<code>stream</code>	Specifies the stream source.
encoding	<code>string</code>	Specifies the encoding to use. The default encoding is 'UTF-8'.

Examples

See the following example:

- [Adding Error Handling to a Job](#) ³²⁵

4.10.1.6 get-stream-filename

Returns a stream's file name with extension if the stream supplied as argument was created from a file. Otherwise, it returns the value of the *default* argument.

Signature

```
get-stream-filename(stream:stream, default:string="") -> string
```

Parameters

Name	Type	Description
stream	stream	Specifies the input stream.
default	string	Specifies the default value to return. By default, this is an empty string.

4.10.1.7 sleep-for

The **sleep-for** function waits for the specified number of seconds before returning the second argument. Depending on what you would like the function to return, the second argument can be of any type (a string, number, stream, etc.). You can also use any suitable expression as the second argument. If you do not need a specific result, you can use, for example, 0 as the second argument. The **sleep-for** function can be particularly useful in [On-Retry](#)¹³⁶ blocks (see *example below*). You can also use this function for testing purposes.

Signature

```
sleep-for(number as number, a as any type) -> a
```

Parameters

Name	Type	Description
number	number	The number of seconds for which the step will be delayed.
a	any type	Returns the result.

Example

The example below shows a [Protected block](#)¹³⁶ that calls the [/system/ftp/retrieve](#)²⁶¹ function to download a file from the FTP server to the local directory. If the execution of the **retrieve** function fails, this step will be retried five times. On each retry, the **sleep-for** function will be computed. FlowForce Server will wait for 30 seconds, return the 'Retrying after 30 seconds' string, and then try downloading the file again.

Execution Steps

Execute with error/success handling - on error, retry times

Execute function

Parameters:

FTP Server:	<input type="text" value="10.100.63.200"/>
Port:	<input type="text" value="21"/>
Directory on host:	<input type="text" value="downloads"/>
Login credentials:	<input type="button" value="+"/>
Use passive mode:	<input type="button" value="+"/>
Use SSL/TLS encryption:	<input type="button" value="+"/>
Verify server certificate:	<input type="button" value="+"/>
Server certificate:	<input type="button" value="+"/>
Source file:	<input type="text" value="ExampleFile.txt"/>
Target file:	<input type="text" value="ExampleFile01.txt"/>
Overwrite target:	<input type="button" value="+"/>
Abort on error:	<input type="button" value="+"/>
Working directory:	<input type="text" value="C:\Downloads"/>
Account:	<input type="button" value="+"/>

= Assign this step's result to as boolean

On retry

Execute function

Parameters:

Expression:	<input type="text" value="sleep-for(30, 'Retrying after 30 seconds')"/>
-------------	---

= Assign this step's result to as T0

= Assign this step's result to

4.10.2 Boolean Functions

The Boolean functions are used to evaluate true/false expressions.

4.10.2.1 all

Returns **true** if all Boolean values are **true**; stops evaluation after the first **false** value and returns **false**.

Signature

```
all(boolVal1:Boolean, boolVal2:Boolean, boolValN:Boolean) -> Boolean
```

Parameters

Name	Type	Description
boolVal1	Boolean	Specifies a Boolean value to evaluate. Subsequent values must be separated by a comma.
boolVal2	Boolean	Same as above.
boolValN	Boolean	Same as above.

4.10.2.2 any

Returns **true** if any Boolean value is **true**; stops evaluation after the first **true** value. Returns **false** if all values are **false**.

Signature

```
any(boolVal1:Boolean, boolVal2:Boolean, boolValN:Boolean) -> Boolean
```

Parameters

Name	Type	Description
boolVal1	Boolean	Specifies a Boolean value to evaluate. Subsequent values must be separated by a comma.
boolVal2	Boolean	Same as above.
boolValN	Boolean	Same as above.

4.10.2.3 false

Returns Boolean **false**.

Signature

```
false() -> Boolean
```

4.10.2.4 if

Returns **valueTrue** if the Boolean condition 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.

Signature

```
if(condition:Boolean, valueTrue:any type, valueFalse:any type) -> any type
```

Parameters

Name	Type	Description
condition	Boolean	Specifies the condition to evaluate.
valueTrue	any type	Specifies a subexpression to return when condition evaluates to true .
valueFalse	any type	Specifies a subexpression to return when condition evaluates to false .

Examples

The following expression passes a Boolean as XML Schema conformant value:

```
if(b, "true", "false")
```

An alternative way to do this:

```
if(b, "1", "0")
```

4.10.2.5 not

Returns the negation of the Boolean value supplied as argument.

Signature

```
not(value: Boolean) -> Boolean
```

Parameters

Name	Type	Description
value	Boolean	Specifies the Boolean value to negate.

4.10.2.6 true

Returns Boolean **true**.

Signature

```
true() -> Boolean
```

4.10.3 MIME/Stream Functions

The MIME (Multipurpose Internet Mail Extensions) message format specifies what type of content a message has and how this message is encoded. To find out more about MIME, see [the Microsoft documentation](#). For example, you can use MIME expression functions when you need to manipulate [MIME headers](#).

The MIME/Stream expression functions can be grouped as follows:

- Functions dealing with headers in streams ([get-mime-header](#)¹⁸⁹, [get-mime-headers](#)¹⁸⁹, [set-mime-header](#)¹⁹⁰, [set-mime-headers](#)¹⁹⁰, [add-mime-header](#)¹⁹¹, [add-mime-headers](#)¹⁹¹, [reset-mime-headers](#)¹⁹²)
- Functions dealing with specific tasks ([is-mime-content-type](#)¹⁹², [get-mime-content-type-param](#)¹⁹³, [get-mime-content-id](#)¹⁹⁴, [set-mime-content-id](#)¹⁹⁵, [set-mime-content-disposition](#)¹⁹⁵, [get-mime-content-disposition-param](#)¹⁹⁶)
- Functions converting streams to different streams ([mime-content-encode](#)¹⁹⁶, [mime-flatten](#)¹⁹⁷, [mime-multipart](#)¹⁹⁷, [mime-multipart-from-list](#)¹⁹⁸, [mime-multipart-related](#)¹⁹⁹, [mime-split-multipart](#)²⁰⁰, [mime-parse](#)²⁰⁰)
- Functions producing new streams ([stream-open](#)²⁰¹, [stream-from-string](#)²⁰¹, [empty-stream](#)²⁰²)

4.10.3.1 get-mime-header

Gets a specific MIME header from the current stream if such a header exists; otherwise, it returns the value of the *default* argument.

Signature

```
get-mime-header(s:stream, key:string, default:string=" ") -> string
```

Parameters

Name	Type	Description
s	<code>stream</code>	Specifies the input stream.
key	<code>string</code>	The <i>key</i> from the key-value pair that forms the header.
default	<code>string</code>	Specifies the default value to return. By default, this is an empty string.

Examples

Assuming that stream `msg` contains the header **Content-Disposition: attachment; filename="GETMSG.edi"**, the following expression will return `attachment; filename="GETMSG.edi"`:

```
get-mime-header(msg, "Content-Disposition", "")
```

In this example, if the stream does not have the "Content-Disposition" header, the expression above will return an empty string (the value of the third argument).

4.10.3.2 get-mime-headers

Gets all MIME headers from a stream and returns a list of tuples (key, value). The returned list can be supplied as *headers* parameter to the `add-mime-headers` expression function.

Signature

```
get-mime-headers(s:stream) -> list of (string, string)
```

Parameters

Name	Type	Description
s	<code>stream</code>	Specifies the input stream.

4.10.3.3 set-mime-header

Returns a stream with header *key* set to *value*, and all other headers and content untouched. If you need to change several headers at once, you might want to use the `set-mime-headers` function.

Signature

```
set-mime-header(s:stream, key:string, value:string) -> stream
```

Parameters

Name	Type	Description
s	<code>stream</code>	Specifies the input stream.
key	<code>string</code>	Specifies the key of the header to set.
value	<code>string</code>	Specifies the header value to set.

Examples

To override the "Content-Type" header, use:

```
set-mime-header(s, "Content-Type", "text/plain; charset=iso-8859-1")
```

4.10.3.4 set-mime-headers

Returns a stream with headers augmented by the key-value pairs from *headers*. The new headers will replace any existing headers of the same name.

Signature

```
set-mime-headers(s:stream, headers:list of (string, string)) -> stream
```

Parameters

Name	Type	Description
s	<code>stream</code>	Specifies the input stream.
headers	<code>list of (string, string)</code>	The list of headers to set.

Examples

To override the "Content-Type" header, use:

```
set-mime-headers(s, list(("Content-Type", "text/plain; charset=iso-8859-1")))
```

4.10.3.5 add-mime-header

Returns a stream with added header `key: value`. This function does not remove an existing header with that key.

Signature

```
add-mime-header(s:stream, key:string, value:string) -> stream
```

Parameters

Name	Type	Description
s	<code>stream</code>	The stream to which the header should be added.
key	<code>string</code>	The <i>key</i> from the key-value pair.
value	<code>string</code>	The <i>value</i> from the key-value pair.

4.10.3.6 add-mime-headers

Returns a stream with all headers from **headers** added.

Signature

```
add-mime-headers(s:stream, headers:list of (string, string)) -> stream
```

Parameters

Name	Type	Description
s	<code>stream</code>	Specifies the input stream.
headers	<code>list of (string, string)</code>	The list of headers to be added. Use the <code>list</code> function to create a list.

Examples

The following expression returns a stream with two headers: **Content-Disposition**, and **Content-Transfer-Encoding**.

```
add-mime-headers(empty-stream(), list(('Content-
Disposition','attachment; name=something'), ('Content-Transfer-Encoding','7bit')))
```

Execution Steps

+

Execute function

Parameters: Expression: add-mime-headers(empty-stream(), list(('Content-Disposition','attachment; name=something'), ('Content-Transfer-Encoding','7bit')))

= Assign this step's result to as T0

new Execution step
new Choose step
new For-each step
new error/success handling step

4.10.3.7 reset-mime-headers

Returns a stream with completely fresh headers. Without a header list, it clears all headers.

Signature

```
reset-mime-headers(s:stream, headers:list of (string, string)=empty) -> stream
```

Parameters

Name	Type	Description
s	<code>stream</code>	Specifies the input stream.
headers	<code>list of (string, string)</code>	Specifies the list of headers to create. The default value is empty.

4.10.3.8 is-mime-content-type

Matches the "Content-Type" header of the stream to custom-defined accept rules. Returns **true** if the "Content-Type" header exists and the rules match its value, otherwise returns **false**. A stream without "Content-Type" header will be treated as "application/octet-stream".

The accept rules have the following format, in extended Backus-Naur form (EBNF) notation:

```
Match ::= Single ("," Single)*
Single ::= Spaces? Type-Match ( Spaces? ";" Spaces? Parameter )* Spaces?
Type-Match ::=
  "*" "*" |
  Type "/" "*" |
  Type "/" "*" Suffix |
  Type "/" Subtype
Parameter ::= Name "=" Value
```


Signature

```
is-mime-content-type(s:stream, accept:string) -> Boolean
```

Parameters

Name	Type	Description
s	stream	Specifies the input stream.
accept	string	Specifies the custom-defined accept rules.

Examples

The following expression will return **true** if stream *msg* contains the header **Content-Type: text/html; charset=utf-8** or **Content-Type: text/plain; charset=utf-8**.

```
is-mime-content-type(msg, "text/*; charset=\"utf-8\"")
```

The following expression will return **true** if stream *msg* contains the header **Content-Type: application/rss+xml** or **Content-Type: application/svg+xml**.

```
is-mime-content-type(msg, "application/*+xml")
```

You can also match multiple rules by separating them with a comma. For example, the following expression will return true if stream *msg* contains the header **Content-Type: text/xml** or **Content-Type: application/xml**:

```
is-mime-content-type(msg, "text/xml, application/xml")
```

4.10.3.9 get-mime-content-type-param

Returns the parameter *param* from the "Content-Type" header of a stream if such header and parameter exists; otherwise, it returns the value of the *default* argument. This function can be used to receive messages that follow the optional AS2 profile **Multiple Attachments (MA)**. Namely, it can extract the starting document Content-ID and Content-Type specified as parameters 'start' and 'type' to *multipart/related* content type. It can also be used to extract the character set, as shown in the example below.

Signature

```
get-mime-content-type-param(s:stream, param:string, default:string="") -> string
```

Parameters

Name	Type	Description
s	<code>stream</code>	Specifies the input stream.
param	<code>string</code>	Specifies the name of the parameter to return.
default	<code>string</code>	Specifies the value to return when the requested <i>param</i> does not exist. By default, this is an empty string.

Examples

Assuming that stream `msg` contains the header **Content-Type: text/html; charset=utf-8**, the following expression will return "utf-8":

```
get-mime-content-type-param(msg, "charset", "ascii")
```

4.10.3.10 get-mime-content-id

Returns the value of the **Content-ID** header from the stream supplied as argument, if such header exists; otherwise, it returns the value of the *default* argument.

Signature

```
get-mime-content-id(s:stream, default:string="") -> string
```

Parameters

Name	Type	Description
s	<code>stream</code>	Specifies the input stream.
default	<code>string</code>	Specifies the value to return when Content-ID header does not exist. By default, this is an empty string.

Examples

Let's suppose that stream `msg` has the header **Content-ID: <root.attachment>**. The expression

```
get-mime-content-id(msg, "")
```

returns "<root.attachment>" in this case. If no such header exists, the expression above returns an empty string (the value of the second argument).

4.10.3.11 set-mime-content-id

Returns a stream with the "Content-ID" header set to *value*, and all other headers and content untouched. You can also achieve the same result using the `set-mime-header` function; this function represents a more direct approach.

Signature

```
set-mime-content-id(s:stream, value:string="") -> stream
```

Parameters

Name	Type	Description
s	<code>stream</code>	Specifies the input string.
value	<code>string</code>	Specifies the value to set in the "Content-Disposition".

Examples

Let's assume that you want to set the value of the "Content-ID" header in stream *msg* to <root.attachment>. To do this, use the following expression:

```
set-mime-content-id(msg, "<root.attachment>")
```

4.10.3.12 set-mime-content-disposition

Sets the parameter of [a MIME Content-Disposition header](#) found in stream *s*.

FlowForce Server Advanced Edition: This function is useful when you send AS2 messages with the optional AS2 profile **FileName preservation (FN)**. See also the [get-mime-content-disposition-param](#)¹⁹⁶ function for reading the file name.

Signature

```
set-mime-content-disposition(s:stream, disposition:string, filename:string="") -> string
```

Parameters

- *s* (type: stream) specifies an input stream.
- *disposition* (type: string) specifies the disposition value of the Content-Disposition header.
- *filename* (type: string) specifies the filename value of the Content-Disposition header. By default, this is an empty string.

Example

The following expression sets the Content-Disposition header as follows: `set-mime-content-disposition(msg, "attachment", "GETMSG.edi")`. You can use the `set-mime-content-disposition` function to make a file downloadable.

4.10.3.13 get-mime-content-disposition-param

Returns the parameter *param* from the "Content-Disposition" header of a stream if such header and parameter exists; otherwise, it returns the value of the *default* argument. This function can be used to receive messages that follow the optional AS2 profile **FileName preservation (FN)** to extract the original file name from the MIME header.

Signature

```
get-mime-content-disposition-param(s:stream, param:string, default:string="") -> string
```

Parameters

Name	Type	Description
s	<code>stream</code>	Specifies the input stream.
param	<code>string</code>	Specifies the name of the parameter to return.
default	<code>string</code>	Specifies the value to return when the specified <i>param</i> and <i>header</i> do not exist. By default, this is an empty string.

Examples

Assuming that stream `msg` contains the header **Content-Disposition: attachment; filename="GETMSG.edi"**, the following expression will return "GETMSG.edi":

```
get-mime-content-disposition-param(msg, "filename")
```

4.10.3.14 mime-content-encode

Applies *encoding* as **Content-Transfer-Encoding** to stream *s*.

The supported encodings are:

- Empty string: Equivalent to "binary".
- "base64": Base64 encoding
- "quoted-printable": Quoted printable encoding

- Any other string: No encoding

The function decodes the stream using the current **Content-Transfer-Encoding** and re-encodes it using the specified encoding. The new **Content-Transfer-Encoding** is stored in the headers of the resulting stream.

The function does not guarantee that errors in the source encoding are reported.

Signature

```
mime-content-encode(s:stream, encoding:string=" ") -> stream
```

Parameters

Name	Type	Description
s	<code>stream</code>	Specifies the input stream.
encoding	<code>string</code>	Specifies the encoding to apply. By default, this is an empty string.

4.10.3.15 mime-flatten

Takes a stream with MIME headers and converts it to a stream that includes the original headers in the content. The resulting stream will have a content type of "message/rfc822".

Signature

```
mime-flatten(s:stream) -> stream
```

Parameters

Name	Type	Description
s	<code>stream</code>	Specifies the input stream.

4.10.3.16 mime-multipart

Takes any number of streams and combines them into a multipart/*subtype*.

The boundary is invented automatically. The streams will be flattened before assembly. Multipart messages with additional parameters are not yet supported.

Note for FlowForce Server Advanced Edition users: The subtype should always be *related* for AS2, as AS2 does not define a meaning for other multipart messages. See also the `mime-multipart-related` function.

Signature

```
mime-multipart(subtype:string, s:stream) -> stream
```

Parameters

Name	Type	Description
subtype	string	Specifies the multipart/ <i>subtype</i> to use.
s	stream	Specifies the input stream.

Examples

The following expression returns a stream that includes two files, an EDI file and a PDF.

```
mime-multipart("related", stream-open("c:\example\order.edi", "application/EDIFACT"), stream-open("c:\example\measuredetails.pdf", "application/pdf"))
```

Execution Steps

4.10.3.17 mime-multipart-from-list

Takes a list of streams and combines them into a multipart/*subtype*.

Signature

```
mime-multipart-from-list(subtype:string, s:list of stream) -> stream
```

Parameters

Name	Type	Description
subtype	string	Specifies the multipart/ <i>subtype</i> to use.
s	list of stream	Specifies the input list of streams.

4.10.3.18 mime-multipart-related

Takes any number of streams and combines them into a multipart/*related*. The boundary is invented automatically. The streams will be flattened before assembly.

Note for FlowForce Server Advanced Edition users: This function can be used to assemble a message that follows the optional AS2 profile **Multiple Attachments (MA)**. The first stream will become a main part. All the parts get the "Content-ID" header with invented unique values before assembling multipart, if they don't have it. The invented value is a new **Message-ID** as returned by the `new-message-id` function. Source streams are not affected.

Signature

```
mime-multipart-related(s:list of stream) -> stream
```

Parameters

Name	Type	Description
s	list of stream	Specifies the input list of streams.

Examples

The following expression returns a stream that includes two streams.

```
mime-multipart-related(list(part1, part2))
```

Job Input Parameters

+ Name: Type: Description:

+ Name: Type: Description:

+

Execution Steps

+

Execute function

Parameters: Expression:

= Assign this step's result to as T0

4.10.3.19 mime-split-multipart

If stream *s* is a MIME multipart message, this function splits it and return a list of streams. If stream *s* is not a multipart message (that is, if `is-mime-content-type(s, "multipart/*")` returns **false**), then the function returns a list of one element—stream *s* (unchanged). The function does not guarantee that errors in the source stream are reported.

Signature

```
mime-split-multipart(s:stream) -> list of stream
```

Parameters

Name	Type	Description
s	stream	Specifies the input stream.

4.10.3.20 mime-parse

Parses a MIME message stored in stream *s*, and separates MIME headers and message body. Returns a stream that has message body content, decoded according to the "Content-Transfer-Encoding" header if needed. MIME headers are accessible via expression functions, like `get-mime-header`, `is-mime-content-type` and such. Reverts what was done by `mime-flatten` function. The function does not guarantee that errors in the source stream are reported.

Signature

```
mime-parse(s:stream) -> stream
```

Parameters

Name	Type	Description
s	stream	Specifies the input stream.

4.10.3.21 stream-open

Creates a stream from an existing file.

Signature

```
stream-open(name:string, contenttype:string=contenttype=application/octet-stream) ->
stream
```

Parameters

Name	Type	Description
name	string	The path of the source file for this stream.
contenttype	string	Specifies the contenttype to associate to the stream. The default is contenttype=application/octet-stream

Examples

The following job opens an existing file having the **.txt** extension and writes it back to the same directory with the **.csv** extension:

Execution Steps

+

Execute function
/system/filesystem/copy
▼
🔗

Parameters:	<table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%; vertical-align: top; padding-right: 5px;">Source:</td> <td style="border: 1px solid #ccc; padding: 2px;">{as-file(stream-open('C:\FlowForce\file.txt'))}</td> </tr> <tr> <td style="vertical-align: top; padding-right: 5px;">Target:</td> <td style="border: 1px solid #ccc; padding: 2px;">C:\FlowForce\file.csv</td> </tr> <tr> <td style="vertical-align: top; padding-right: 5px;">Overwrite:</td> <td style="padding: 2px;"><input checked="" type="checkbox"/></td> </tr> <tr> <td style="vertical-align: top; padding-right: 5px;">Abort on error:</td> <td style="padding: 2px;">+</td> </tr> <tr> <td style="vertical-align: top; padding-right: 5px;">Working directory:</td> <td style="padding: 2px;">+</td> </tr> </table>	Source:	{as-file(stream-open('C:\FlowForce\file.txt'))}	Target:	C:\FlowForce\file.csv	Overwrite:	<input checked="" type="checkbox"/>	Abort on error:	+	Working directory:	+
Source:	{as-file(stream-open('C:\FlowForce\file.txt'))}										
Target:	C:\FlowForce\file.csv										
Overwrite:	<input checked="" type="checkbox"/>										
Abort on error:	+										
Working directory:	+										

4.10.3.22 stream-from-string

Creates a stream from a string using the supplied encoding. The content type supplied as argument is associated to the stream. This type of stream is not automatically saved as a file.

Signature

```
stream-from-string(string:string, encoding:string="UTF-8",
contenttype:string=contenttype=text/plain) -> stream
```

Parameters

Name	Type	Description
string	string	The string from which the stream should be created.
encoding	string	Specifies the encoding to use. The default encoding is 'UTF-8'.
contenttype	string	Specifies the <code>contenttype</code> to associate to the stream. The default is <code>contenttype=text/plain</code>

4.10.3.23 empty-stream

Creates an empty stream.

Signature

```
empty-stream() -> stream
```

4.10.4 Result Functions

Result functions allow you to process the result returned by execution steps.

4.10.4.1 stdout

Some execution steps (such as those that run shell commands) return standard output. For example, the shell command `dir` (on Windows) returns a list of directories. When a step returns a result, FlowForce Server automatically assigns to it the generic type `result`. With the `stdout` function, you can get access to the standard output of result, as follows:

```
stdout(result)
```

where `result` is the value returned by some execution step.

This function fails if `result` does not provide standard output.

Signature

```
stdout(result:result) -> stream
```

Parameters

Name	Type	Description
result	result	The result of the step whose standard error you want to get.

Examples

See the following examples:

- [Adding Error Handling to a Job](#)³²⁵
- [Validate an XML Document with Error Logging](#)³⁶¹
- [Check if a Path Exists](#)²⁹⁹

4.10.4.2 stderr

Returns the standard error of the result. Fails if the result does not provide a standard error.

Signature

```
stderr(result:result) -> stream
```

Parameters

Name	Type	Description
result	result	The result of the step whose standard error you want to get.

Examples

See [Adding Error Handling to a Job](#)³²⁵ for an example.

4.10.4.3 exitcode

Returns the numeric exit code of the result.

Signature

```
exitcode(result:result) -> number
```

Parameters

Name	Type	Description
result	result	The result of the step whose exit code you want to get.

Examples

The following job consists of two steps. The first step invokes a Windows command line command which attempts to create a directory called **data** in the current working directory (C:\FlowForce). The result of this step is declared as **outcome**. The second step gets the **outcome** and returns the numeric exit code from it, with the help of the `exitcode` function. The numeric exit is then converted to a string, with the help of the `string` function. This conversion is required because the data type of the expression is string.

Importantly, the **Abort on error** option is not selected; otherwise, the execution would stop in case of error, and so there wouldn't be any exit code for the second step to process.

The screenshot shows the 'Execution Steps' configuration window. It contains two steps:

- Step 1:** 'Execute function /system/shell/commandline'. Parameters: Command: 'mkdir data', Abort on error: , Working directory: 'C:\FlowForce'. Below the parameters, it says '= Assign this step's result to outcome as result'.
- Step 2:** 'Execute function /system/compute-string'. Parameter: Expression: 'The exit code is: {string(exitcode(outcome))}'.

When the job runs for the first time, the **data** directory is supposedly created successfully, and the exit code would be **0**. On subsequent runs, it cannot be created because it already exists, so the exit code would be **1**.

See also [Adding Error Handling to a Job](#) ³²⁵.

4.10.4.4 error-message

This function returns the text of the error message encountered by a step. The typical usage of this function is inside a protected block, and specifically inside the "On Error" handler. The function may return an empty string if no error has been encountered or if it is not technically possible to retrieve the text of the error due to the nature of the job.

Signature

```
error-message(result:result) -> string
```

Parameters

Name	Type	Description
result	result	Supplies the erroneous step from which the error text should be retrieved. To get the erroneous step, call the <code>failed-step()</code> function.

Examples

See [Add error handling to a job](#) ³²⁵.

4.10.4.5 results

Returns a list of streams of the specified result, optionally filtered by name. Use the function `nth` to access a particular value in the list.

Signature

```
results(result:result, name:string) -> list of stream
```

Parameters

Name	Type	Description
result	result	Mandatory parameter. The result of the step from which you want to return a list of streams.
name	string	Optional parameter. When provided, filters by name a particular value in the result.

Examples

Example A. Let's suppose that you have deployed to FlowForce Server a mapping that generates a single XML file as output. An example of such a mapping is **CompletePO.mfd** included with MapForce examples. The name of the target XML component in MapForce is "CompletePO". To process the result of this mapping and save it to a file from FlowForce, configure the job as follows:

Execution Steps

+ Execute function /public/CompletePO.mapping

Parameters:	Customers: (input) +
	Articles: (input) +
	ShortPO: (input) +
	CompletePO: (output) +
	Working-directory: +

= Assign this step's result to mapping_result as CompletePO

+ Execute function /system/filesystem/copy

Parameters:	Source: {as-file(nth(results(mapping_result), 0))}
	Target: output.xml
	Overwrite: <input checked="" type="checkbox"/>
	Abort on error: +
	Working directory: C:\FlowForce\CompletePO

= Assign this step's result to name as boolean

In the job configuration above, the first step runs the mapping and returns the result as **mapping_result**. In the second step, the expression

```
{as-file(nth(results(mapping_result), 0))}
```

processes the **mapping_result** and converts it to a file. Namely, the `results` function picks the list of streams from the MapForce component. The `nth`²⁰⁹ function picks the first item from this list. Finally, the `as-file`²¹⁴ function generates a file from the stream.

The `copy`²⁴¹ function copies the generated file to the working directory. The **Target** text box defines the name of the generated file. Any existing file with the same name will be overwritten.

Example B. Let's suppose that you have deployed to FlowForce Server a mapping that has two target XML components, "MarketingExpenses" and "DailyExpenses". An example of such a mapping is **MarketingAndDailyExpenses.mfd** included with MapForce examples. To generate a file from the "DailyExpenses" component, create a job similar to the one above, but change the expression to:

```
{as-file(nth(results(mapping_result, 'DailyExpenses'), 0))}
```

The only difference here is that the list of streams produced by the mapping is filtered by the name of the desired component (in this case, "DailyExpenses").

Example C. Let's suppose that you have deployed to FlowForce Server a mapping that generates multiple XML files dynamically. The output file names are generated by the mapping itself and are not known before runtime. An example of such a mapping is **DividePersonsByDepartmentIntoGroups.mfd** included with MapForce examples. To generate the third output file of the mapping, create a job similar to the one above, and change the expression to:

```
{as-file(nth(results(mapping_result), 2))}
```

Here we need the third file, so the index supplied as second argument to the `nth` function is **2** (not **3**), because the index is zero-based.

See also the following examples:

- [Creating a Job from a StyleVision Transformation](#)³⁵¹
- [Generate Multiple PDFs from Multiple XMLs](#)³⁷¹

4.10.4.6 make-error-result

This expression function produces a result object with empty `stdout` and `stderr`, the exit code as specified (the default value is 1, which represents an error), and the error message as specified (the default value is an empty string) and no result files. The `make-error-result` function is used in Resume steps. For details, see [Resume Steps](#)¹³⁸.

Signature

```
make-error-result (exit-code:number=1, error-message:string=" ") -> result
```

Example

This example shows how the `make-error-result` function can be used in a job. The sample job illustrated below consists of an [Error/Success-Handling block](#)¹³⁶ that executes a MapForce mapping and an Execution step that computes an expression. In the Error/Success-Handling block, there is also an On-Error handler that contains a Resume step. In case an error occurs during mapping execution, we do not want the job to fail. Therefore, we proceed as follows:

1. We define a Resume step that will replace the result of the failed protected step with a newly constructed result object and set the result of the [protected block](#)¹³⁶ to the computed expression.
2. We also want to use the result of the `make-error-result` function in the next step, by assigning it to `Map`.
3. In the next step, we extract the exit code and the error message from `Map` with the following expression for the `compute-string` function: `{string(exitcode(Map))}, {error-message(Map)}`. This expression converts the exit code to a string and concatenates this string with the error message.

The values of the exit code and the error message are the arguments passed to the `make-error-result` function in the Resume step.

Our sample job is configured as a [Web service](#) ¹⁵⁸. If an error occurs, the browser window will display the following result:

```
1, Mapping has failed!
```

If you do not want to access the expression computed in the Resume step, you do not need to assign the step's result to anything. Instead, you can add any step(s) outside the Error/Success-Handling step. The new step(s) will be processed after the Resume step has been executed.

4.10.4.7 make-success-result

This expression function produces a result object with empty `stdout` and `stderr`, exit code 0, an empty error message, and no result files. The `make-success-result` function is used in Resume steps. For details, see [Resume Steps](#) ¹³⁸.

Signature

```
make-success-result () -> result
```


4.10.4.8 merge-results

This `merge-results` function takes a sequence of result objects and merges it to a single result object. This expression function produces a result object that sets the exit code to the maximum number of all the exit codes in the sequence, the error message to be a concatenation of all the individual error messages in order, `stdout` and `stderr` to be a concatenation of the individual `stdout` and `stderr`, and the result files to be a collection from all the input results, in the order produced.

The `merge-results` function is used in Resume steps. For details, see [Resume Steps](#) ¹³⁸.

Signature

```
merge-results (sequence of result) -> result
```

4.10.5 List Functions

List functions are used to create and disassemble lists. Lists always contain items of a single type (for example, only strings, only number, or only nested lists with the same item type); there are no mixed type lists.

4.10.5.1 nth

Returns the specified item from the list. The index is zero-based. Fails if the index is out of bounds.

Signature

```
nth(list:list, index:number) -> item
```

Parameters

Name	Type	Description
<code>list</code>	<code>list</code>	Specifies the input list.
<code>index</code>	<code>number</code>	Specifies the zero-based index of the item to return.

Examples

The following expression returns "b":

```
nth(list('a', 'b', 'c'), 1)
```

4.10.5.2 length

Returns the number of items in the list.

Signature

```
length(list:list) -> number
```

Parameters

Name	Type	Description
list	list	Specifies the input list object.

4.10.5.3 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.

Signature

```
list(item1:any type, item2:any type, itemN:any type) -> list
```

Parameters

Name	Type	Description
item1	any type	Specifies a single item. Subsequent items must be separated by a comma.
item2	any type	Same as above
itemN	any type	Same as above.

Examples

The following expression returns the list **[1, 2, 3]**. All list items are of numeric type:

```
list(1,2,3)
```

The following expression returns the list **['a', 'b', 'c']**. All list items are of string type:

```
list('a','b','c')
```

4.10.5.4 from-to

Returns the list of integers between "from" and "to" inclusive. If "from" is greater than "to", this list is empty.

Signature

```
from-to(from:number, to:number) -> list of number
```

Parameters

Name	Type	Description
from	number	Specifies the starting index ("from").
to	number	Specifies the ending index ("to").

Examples

The following expression produces **[3, 4, 5, 6, 7]**:

```
from-to(3, 7)
```

4.10.5.5 slice

Returns a partial list from a list.

Signature

```
slice(list:list, start:number, end:number=length(list)) -> list
```

Parameters

Name	Type	Description
list	<code>list</code>	Specifies the input list.
start	<code>number</code>	Specifies the zero-based index of the first list item to include in the slice.
end	<code>number</code>	Specifies the zero-based index of the first item to ignore in the slice.

Examples

The following expression returns `list(2,3)`:

```
slice(list(1,2,3,4),1,3)
```

4.10.5.6 join

Concatenates the lists given by the first argument using the second argument as separator between each pair of lists.

Signature

```
join(lists:list of lists, separator:list=empty list) -> list
```

Parameters

Name	Type	Description
lists	<code>list of lists</code>	Specifies the lists to join. This argument must be a list of two or more lists. All nested lists must be of the same type.

Name	Type	Description
separator	<code>list</code>	<p>Optional argument which specifies the separator by which to delimit the joined lists. If not supplied, no separator will be used.</p> <p>The separator must be of type <code>list</code>. Use the <code>list</code> function to create a separator. For example, the expression <code>list(',')</code> specifies a single comma character as separator.</p>

Examples

The following execution steps illustrate how to join two lists. Step 1 produces the first list. Step 2 produces the second list. Step 3 creates an object of type "list of lists" that contains both lists. Finally, step 4 joins the lists, using the semi-colon character as separator.

Execution Steps

- +**

Execute function `/system/compute`

Parameters: Expression: `list('a', 'b')`

= Assign this step's result to `list1` as T0
- +**

Execute function `/system/compute`

Parameters: Expression: `list('c', 'd')`

= Assign this step's result to `list2` as T0
- +**

Execute function `/system/compute`

Parameters: Expression: `list(list1, list2)`

= Assign this step's result to `list_of_lists` as T0
- +**

Execute function `/system/compute`

Parameters: Expression: `join(list_of_lists, list(';'))`

= Assign this step's result to `name` as T0

4.10.6 File System Functions

File system functions permit access to the file system. To execute these functions, the job must use the credentials of a user account with corresponding access rights on the operating system.

4.10.6.1 as-file

Creates a file if the stream source is a file. Creates a temporary file if the stream source is not a file.

Signature

```
as-file(stream:stream) -> string
```

Parameters

Name	Type	Description
stream	stream	Specifies the stream source.

Examples

The following job creates a file called **file.txt** with one line of text. First, the [stream-from-string](#)²⁰¹ function generates a stream from the text supplied as argument. Next, the `as-file` function takes the stream as argument and generates a temporary file from it. To copy the temporary file to a permanent path, the built-in [copy](#)²⁴¹ function is called from a separate execution step. The file is copied to the working directory of the job (C:\FlowForce) and will be overwritten each time the job runs.

Execution Steps

+
Execute function /system/compute
▼
🔗

Parameters: Expression: `as-file(stream-from-string('This is my file content.))`

= Assign this step's result to as T0

+
Execute function /system/filesystem/copy
▼
🔗

Parameters:

Source:

Target:

Overwrite:

Abort on error: +

Working directory:

See also [Validate an XML Document with Error Logging](#)³⁶¹.

4.10.6.2 list-files

Lists the file/s specified by the path, which may end with a wildcard. It returns the 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.

Signature

```
list-files(path:string) -> list of string
```

Parameters

Name	Type	Description
path	string	Specifies the path to a directory or file.

Examples

See [Copy Files](#) ³⁰³ for an example.

4.10.6.3 list-directories

Lists the subdirectories in the path (which may terminate with a wildcard) and returns the resulting string list.

Signature

```
list-directories(path:string) -> list of string
```

Parameters

Name	Type	Description
path	string	Specifies the path to a directory.

4.10.6.4 join-paths

Combines paths supplied as arguments into one path.

Signature

```
join-paths(string1:string, string2:string, stringN:string) -> string
```

Parameters

Name	Type	Description
string1	<code>string</code>	Specifies a single path step to join. All subsequent arguments must be separated by a comma.
string2	<code>string</code>	Same as above.
stringN	<code>string</code>	Same as above.

Examples

On Windows, the following expressions return "C:\tmp\test.txt":

```
join-paths('C:\tmp', 'test.txt')
join-paths('C:\tmp\', 'test.txt')
join-paths('C:\', 'tmp', 'test.txt')
join-paths('C:\Users', '\tmp', 'test.txt')
join-paths('D:\Data', 'C:\tmp', 'test.txt')
```

On Linux and MacOS, the following expressions return "/home/user/test.txt":

```
join-paths('/home/user', 'test.txt')
join-paths('/var', '/home/user', 'test.txt')
```

4.10.6.5 parent-directory

Extracts the parent directory from a path.

Signature

```
parent-directory(path:string) -> string
```



Parameters

Name	Type	Description
path	<code>string</code>	Specifies the path to a directory.


Examples

Let's assume you have a MapForce mapping which updates a database from an XML file. You've deployed it to FlowForce Server already and created a job from it. Also, you've configured the job run when the content of the directory changes (that is, your job uses a file system trigger, see [File System Triggers](#)¹⁵⁴).

The first step of the job runs the mapping which updates the database:

Execute function /public/Tutorials/Altova_Hierarchical_DB.mapping	
Parameters:	Altova_Hierarchical: (input)  
	Working-directory: C:\FFSERV
Assign this step's result to <input type="text" value="name"/>	

After the mapping step finishes executing, your goal is to move the source XML file into the subdirectory called "processed". This would help you keep a track of which files have been processed. To achieve this goal, add a new step which calls the **/system/filesystem/move** function and enter as *Source* and *Destination* the values shown below:

Execute function /system/filesystem/move	
Parameters:	Source: {triggerfile}
	Destination: {parent-directory(triggerfile)}processed
	Overwrite target: <input checked="" type="checkbox"/>
	Abort on error: 
	Working directory: C:\FFSERV

The parameter value {triggerfile} in the *Source* field instructs FlowForce to move specifically the file which triggered the mapping. The parameter value

```
{parent-directory(triggerfile)}processed
```

in the *Destination* field sets as destination a directory called "processed", inside the current directory. It consists of an expression and of a string. Note that only the expression part is delimited by curly braces (see [Embedding Expressions in String Fields](#)¹⁷⁵). The expression

```
{parent-directory(triggerfile)}
```

calls the `parent-directory` function and supplies to it the value "triggerfile" as argument.

Therefore, when the job runs, the following actions take place:

1. A script or a user copies a file (let's call it **source.xml**) into the current working directory (for example, **C:\FFSERV**).
2. The trigger fires and **source.xml** becomes the "triggerfile".

- FlowForce Server executes the step which runs the mapping.
- FlowForce Server executes the step which moves **source.xml** to the "processed" subdirectory. Note that the path **C:\FFSERV\processed** must exist.

4.10.6.6 filename-with-extension

Extracts the file name and extension from a path.

Signature

```
filename-with-extension(path:string) -> string
```

Parameters

Name	Type	Description
path	string	Specifies the path to a file.

Examples

The following expression returns "file.txt":

```
filename-with-extension("c:\temp\file.txt")
```

4.10.6.7 filename

Extracts the file name (without extension) from a path.

Signature

```
filename(path:string) -> string
```

Parameters

Name	Type	Description
path	string	Specifies the path to a file.

Examples

The following expression returns "file":

```
filename("c:\temp\file.txt")
```

4.10.6.8 extension

Extracts the file extension from a path.

Signature

```
extension(path:string) -> string
```

Parameters

Name	Type	Description
path	string	Specifies the path to a file.

Examples

The following expression returns ".txt":

```
extension("c:\temp\file.txt")
```

4.10.7 String Functions

The string functions enable you to perform the following basic operations:

- Carrying out conversion ([string](#)²²⁰, [number](#)²²⁰, [char](#)²²¹, [code](#)²²²)
- Manipulating the length of strings ([concat](#)²²², [string-join](#)²²³, [split](#)²²⁴, [find-all](#)²²⁴, [trim](#)²²⁵, [trim-start](#)²²⁵, [trim-end](#)²²⁵, [substring](#)²²⁶)
- Working with string properties ([contains](#)²²⁶, [starts-with](#)²²⁷, [ends-with](#)²²⁸, [string-length](#)²²⁸)

4.10.7.1 string

Computes the string representation of the given number, i.e. converts the number supplied as argument into a string.

Signature

```
string(number:number) -> string
```

Parameters

Name	Type	Description
number	number	The number to be convert to string.

Examples

The following expression converts the numeric value **1** into the string "1":

```
string(1)
```

4.10.7.2 number

Computes the number representation of the string, i.e. converts the string supplied as argument into a number.

Signature

```
number(string:string) -> number
```

Parameters

Name	Type	Description
string	string	The input string value to convert.

Examples

The following expression converts the string value "1" into the numeric value **1**:

```
number('1')
```

4.10.7.3 char

Returns a string that contains the Unicode character of the number supplied as argument. For example, `char(10)` returns a Line Feed. To find out the numeric code of a specific Unicode character, use the `code` function.

Signature

```
char(number: number) -> string
```

Parameters

Name	Type	Description
number	<code>number</code>	The numeric code of the character. This code is equivalent to the decimal code used to represent a Unicode character in HTML (for example, 8734 represents the infinity symbol).

Examples

The following execution step returns the infinity symbol:

The screenshot shows a software interface for configuring execution steps. At the top, there is a section titled "Execution Steps" with a plus icon to add new steps. Below this, a step is configured with the function `/system/compute`. The "Parameters" section shows the "Expression" field containing `char(8734)`. Below the parameters, there is an equals sign followed by the text "Assign this step's result to" and a text input field containing the word "name". Below the configuration area, there are three buttons: "new Execution step", "new Choose step", and "new For-each step". Below the buttons is a section titled "Execution Result" with a dropdown menu labeled "Declare return type as:" which is currently set to "string".

4.10.7.4 code

Returns the Unicode value of the first character of the string supplied as argument.

Signature

```
code(string:string) -> number
```

Parameters

Name	Type	Description
string	string	Specifies the input string.

Examples

The following execution step returns the numeric value **32**, which represents the space character:

Execution Steps

+
Execute function /system/compute

Parameters:
Expression: `code(' ')`

=
Assign this step's result to as T0

new Execution step
new Choose step
new For-each step
new error/success handling step

Execution Result

Declare return type as:

4.10.7.5 concat

Concatenates the strings supplied as arguments into one string. To concatenate all items of an object of type "list of string", use the `string-join` function.

Signature

```
concat(string1:string, string2:string, stringN:string) -> string
```

Parameters

Name	Type	Description
string1	<code>string</code>	Specifies a single string item to join. All subsequent arguments must be separated by a comma.
string2	<code>string</code>	Same as above.
stringN	<code>string</code>	Same as above.

Examples

The following expression returns "abc":

```
concat('a', 'b', 'c')
```

4.10.7.6 string-join

Joins the list of strings supplied as argument into a string. Optionally, inserts the separator supplied as argument in between each string.

Signature

```
string-join(list:list of string, separator:string="") -> string
```

Parameters

Name	Type	Description
list	<code>list of string</code>	The input list of string.
separator	<code>string</code>	Optional argument. Specifies the separator by which all joined strings should be delimited.

Examples

The following expression will return the string **a;b;c**:

```
string-join(list('a', 'b', 'c'), ';')
```

4.10.7.7 split

Splits the string supplied as argument at each occurrence of **separator**.

Signature

```
split(string:string, separator:string) -> list of string
```

Parameters

Name	Type	Description
string	string	The input string.
separator	string	The separator string.

Examples

The following expression will return the list ["1", "2", "3"]:

```
split('1;2;3', ';')
```

4.10.7.8 find-all

Extracts all occurrences of **pattern** in the string, where **pattern** is a regular expression.

Signature

```
find-all(string:string, pattern:string) -> list of string
```

Parameters

Name	Type	Description
string	string	The input string.
pattern	string	The pattern as a regular expression.

Examples

The following expression extracts all occurrences of "o" from string "apollo".

```
find-all('apollo', 'o')
```


The result is the following list of string: ["o", "o"]

4.10.7.9 trim

Removes leading and trailing whitespace characters from the string (**Space**, **Tab**, **Line Feed**, **Carriage Return**, **Form Feed**, and **Vertical Tab**).

Signature

```
trim(string:string) -> string
```

Parameters

Name	Type	Description
string	<code>string</code>	The input string.

4.10.7.10 trim-start

Removes leading whitespace from the string supplied as argument (see also the `trim` function).

Signature

```
trim-start(string:string) -> string
```

Parameters

Name	Type	Description
string	<code>string</code>	The input string.

4.10.7.11 trim-end

Removes trailing whitespace from the string supplied as argument (see also the `trim` function).

Signature

```
trim-end(string:string) -> string
```

Parameters

Name	Type	Description
string	<code>string</code>	The input string.

4.10.7.12 substring

Returns a substring from the specified string, beginning with **start** character position, up to the **end** character position. The start and end indexes are zero-based.

If not set, **end** is the length of the supplied string.

The **end** argument can also be a negative integer. A negative value *-n* means "trim the last *n* characters from the string".

Signature

```
substring(string:string, start:number, end:number) -> string
```

Parameters

Name	Type	Description
string	string	The input string.
start	number	The zero-based starting index.
end	number	The zero-based ending index.

Examples

The following expression will return "Force":

```
substring('FlowForce', 4)
```

The following expression will return "t":

```
substring('Altova', 2, 3)
```

The following expression will return "ItoV":

```
substring('Altova', 1, -1)
```

4.10.7.13 contains

Returns **true** if the first string contains at least one occurrence of substring, otherwise **false**.

Signature

```
contains(string:string, substring:string) -> Boolean
```

Parameters

Name	Type	Description
string	<code>string</code>	The input string.
substring	<code>string</code>	The string value to check for.

Examples

The following expression returns **true**:

```
contains('cat', 'a')
```

The following expression returns **false**:

```
contains('cat', 'b')
```

4.10.7.14 starts-with

Returns **true** if the string supplied in the **string** argument starts with the string supplied in the **start** argument.

Signature

```
starts-with(string:string, start:string) -> Boolean
```

Parameters

Name	Type	Description
string	<code>string</code>	The input string.
start	<code>string</code>	The string value to check for.

Examples

The following expression returns **true**:

```
starts-with('cat', 'c')
```

The following expression returns **false**:

```
starts-with('cat', 'b')
```

4.10.7.15 ends-with

Returns **true** if the string supplied in the **string** argument ends with the string supplied in the **end** argument.

Signature

```
ends-with(string:string, end:string) -> Boolean
```

Parameters

Name	Type	Description
string	string	The input string.
end	string	The string value to check for.

Examples

The following expression returns **true**:

```
ends-with('cat', 't')
```

The following expression returns **false**:

```
ends-with('cat', 'a')
```

4.10.7.16 string-length

Returns the number of characters in the string.

Signature

```
string-length(string:string) -> number
```

Parameters

Name	Type	Description
string	string	The input string.

Examples

The following expression will return **3**:

```
string-length('cat')
```

4.10.8 Execution State Functions

This sections includes expression functions that deal with the execution state of a job. For example, if a [protected sequence](#)¹³⁶ encounters an error, you can use the [retry-count](#)²³¹ function to get the number of times a [protected block](#)¹³⁶ has been retried.

4.10.8.1 failed-step

Returns the result of a failed execution step. Using this function is meaningful when you are handling errors with protected blocks, as described in [Handling Step Errors](#)¹³⁶. The **failed-step** function must be part of the "On error" handler; otherwise, the step where you are using it will fail because there is no erroneous step.

This function returns a value of type **result** that represents the result of the erroneous step. To find the **result's** attributes, pass this function as argument to expression functions such as **stdout** or **stderr**, for example:

```
stderr(failed-step())
stdout(failed-step())
```

Whether you should use **stderr** or **stdout** depends on whether the failing step returns the error information in the standard error or standard output streams, respectively.

Signature

```
failed-step() -> result
```

Examples

The job illustrated below uses error handling, so it qualifies for a call to the **failed-step** function. The first execution step attempts to run a shell command which is supplied as a job input parameter. If the command fails with an error, the "On error" handler will be executed. The first and only step of the "On Error" handler calls an error handling sub-job which was created separately and is discussed below.

Job Input Parameters

Type: Default: Description:

Execution Steps

Execute with error/success handling - on error, retry times

Execute function

Parameters: Command:

Abort on error:

Working directory:

= Assign this step's result to as result

On error

Execute function

Parameters: inputResult:

workingDirectory:

Although you can configure your error handling differently, the error-handling sub-job in this example takes two input parameters:

1. **inputResult** - the output of the **failed-step** function, of type **result**.
2. **workingDirectory** - the directory to which the log file containing the error details will be written.

The error handling sub-job looks as follows:

Job Input Parameters

+ Name: Type:

+ Name: Type: Default: 🗑️

Execution Steps

+ ▾ Execute function

Parameters:	Source: <input type="text" value="{as-file(stderr(inputResult))}"/>
	Target: <input type="text" value="error.log"/>
	Overwrite: <input checked="" type="checkbox"/>
	Abort on error: +
	Working directory: <input type="text" value="{workingDirectory}"/>

= Assign this step's result to as boolean

The execution step above invokes the `copy` function in order to create a file called **error.log** in the job's working directory. The expression from the **Source** text box does the following:

1. The `stderr` expression function converts the standard error provided by `inputResult` to a stream. As mentioned above, in some cases, you might need to use `stdout` instead of `stderr`. Both `stdout` and `stderr` take a value of type **result** as argument. That's precisely the return type produced by the `failed-step` function (which in this example was called in the main job).
2. The `as-file` function converts the stream to a file and writes it to the disk. The path of the file is relative to the working directory.

For more examples, see:

- [Add Error Handling to a Job](#) ³²⁵
- [Validate an XML Document with Error Logging](#) ³⁶¹

4.10.8.2 retry-count

Returns a number that indicates how many times FlowForce re-tried the execution of one or more steps that have error/success handling (a so-called "protected block"). Note that the function specifically evaluates the innermost protected block surrounding the function. If no retries took place (that is, if the first run of the protected block was successful), the return value is **0**. See also [On-Retry](#) ¹³⁷.

Signature

```
retry-count() -> number
```

4.10.8.3 instance-id

Returns a unique string for every job execution. This can be used to create a unique directory for each job execution, where the string is used to define the directory name.

Signature

```
instance-id() -> string
```

4.10.8.4 slot-number

Returns the execution slot number of the queue currently running the job. This number should not be used as a file name. The number can be used to access different servers to execute parallel jobs (simple load balancing).

The slot number depends on the queue in which the slot execution was started. If the current job is called by another job, then it inherits the slot number of the calling job.

Signature

```
slot-number() -> number
```

4.10.9 Runtime Information Functions

The runtime information functions can be used to handle the details of the currently running jobs.

4.10.9.1 log

Converts the expression received as argument to string and writes it to the system log. This function is useful in situations where you want to explicitly log the expression produced by a step. Logging values this way has the effect that no truncation of values occurs in the system log when the logged values are too long, see also [Logging Settings](#)¹²².

Signature

```
log(expression:T0) -> string
```


Parameters

Name	Type	Description
expression	T0	The FlowForce expression to be logged, of type T0 (any type).

Examples

Let's assume that you have created a job which gets a list of files from the given path, like the one below.

Execution Steps

- +
 - ▲
 Execute function
 - Parameters: | Expression:
 - = Assign this step's result to as T0

If the number of files in the source directory exceeds the FlowForce default logging limit for lists, then entries in the job log become truncated. As illustrated below, in this example, only the first 10 file names are shown. Also, the last character in each file path has been truncated, because the path has exceeded the default limit of 50 characters.

Date	Message
2020-09-21 15:20:17	Starting instance 67.
2020-09-21 15:20:17	Starting job execution: job /public/Examples/Functions/log-function in queue /public/Examples/Functions/log-function
2020-09-21 15:20:17	Running instance 67 locally.
2020-09-21 15:20:17	▲ Job /public/Examples/Functions/log-function
2020-09-21 15:20:17	▲ System function <input type="text" value="/system/compute"/>
2020-09-21 15:20:17	Computed list("C:\FlowForceExamples\LogFunction\source\File 01.tx[...]", "C:\FlowForceExamples\LogFunction\source\File 02.tx[...]", "C:\FlowForceExamples\LogFunction\source\File 03.tx[...]", "C:\FlowForceExamples\LogFunction\source\File 04.tx[...]", "C:\FlowForceExamples\LogFunction\source\File 05.tx[...]", "C:\FlowForceExamples\LogFunction\source\File 06.tx[...]", "C:\FlowForceExamples\LogFunction\source\File 07.tx[...]", "C:\FlowForceExamples\LogFunction\source\File 08.tx[...]", "C:\FlowForceExamples\LogFunction\source\File 09.tx[...]", "C:\FlowForceExamples\LogFunction\source\File 10.tx[...]", ...)
2020-09-21 15:20:17	Finished job execution: job /public/Examples/Functions/log-function in queue /public/Examples/Functions/log-function

To prevent truncation from happening, enclose the expression inside the `log` function, and save the job configuration.

Execution Steps

+

▲
Execute function `/system/compute`



Parameters:

Expression: `log(list-files('C:\FlowForceExamples\LogFunction\source'))`

=
Assign this step's result to as T0

If you run the job with the new configuration, the log now contains a new entry for the logged expression, in addition to the entry logged by the system. Truncation no longer occurs.

Date	Message
2020-09-21 15:21:41	Starting instance 68.
2020-09-21 15:21:41	Starting job execution: job <code>/public/Examples/Functions/log-function</code> in queue <code>/public/Examples/Functions/log-function</code>
2020-09-21 15:21:41	Running instance 68 locally.
2020-09-21 15:21:41	<div style="border: 1px solid gray; padding: 2px;"> ▲ Job <code>/public/Examples/Functions/log-function</code> </div>
2020-09-21 15:21:41	<div style="border: 1px solid gray; padding: 2px;"> ▲ System function <code>/system/compute</code> </div>
2020-09-21 15:21:41	<div style="border: 1px solid gray; padding: 2px;"> step 1 parameter 'Expression': <code>C:\FlowForceExamples\LogFunction\source\File 01.txt C:\FlowForceExamples\LogFunction\source\File 02.txt C:\FlowForceExamples\LogFunction\source\File 03.txt C:\FlowForceExamples\LogFunction\source\File 04.txt C:\FlowForceExamples\LogFunction\source\File 05.txt C:\FlowForceExamples\LogFunction\source\File 06.txt C:\FlowForceExamples\LogFunction\source\File 07.txt C:\FlowForceExamples\LogFunction\source\File 08.txt C:\FlowForceExamples\LogFunction\source\File 09.txt C:\FlowForceExamples\LogFunction\source\File 10.txt C:\FlowForceExamples\LogFunction\source\File 11.txt C:\FlowForceExamples\LogFunction\source\File 12.txt C:\FlowForceExamples\LogFunction\source\File 13.txt C:\FlowForceExamples\LogFunction\source\File 14.txt C:\FlowForceExamples\LogFunction\source\File 15.txt C:\FlowForceExamples\LogFunction\source\File 16.txt C:\FlowForceExamples\LogFunction\source\File 17.txt C:\FlowForceExamples\LogFunction\source\File 18.txt C:\FlowForceExamples\LogFunction\source\File 19.txt C:\FlowForceExamples\LogFunction\source\File 20.txt C:\FlowForceExamples\LogFunction\source\File 21.txt</code> </div>

In this example, as an alternative to calling the `log` expression function, you can also click the "Enable logging"  button next to the step parameter you wish to log. Doing this is equivalent to using the `log` function, so FlowForce will hide the `log` function next time when you open the job configuration page. The difference between the  button and the `log` function is that the former logs the entire expression displayed in the text box, whereas the `log` function can be used selectively for smaller sub-expressions, for example:

+

▲
Execute function `/system/compute`

Parameters:

Expression: `contains(log("haystack"), "needle")`

=
Assign this step's result to as T0

4.11 System Functions

This section describes system functions in FlowForce Server, which allow you to copy and move files, create directories, send emails, execute shell commands, and perform other actions. These functions are available in the `/system` container. The following groups of system functions are available:

- [/system](#)²³⁵
- [/system/filesystem](#)²⁴¹
- [/system/ftp](#)²⁴⁵
- [/system/mail](#)²⁷⁸
- [/system/maintenance](#)²⁸⁴
- [/system/shell](#)²⁸⁵

Most system functions have parameters. Parameters can accept different values, including [expressions](#)¹⁷¹ and [expression functions](#)¹⁸⁰.

If [RaptorXML/RaptorXML+XBRL Server](#) is integrated into FlowForce Server, an additional container with all RaptorXML/RaptorXML+XBRL Server functions becomes available. For more information, see [Integration with RaptorXML Server](#)⁴³³.

Windows network paths

When you create jobs, you will need to refer to file paths on the machine where FlowForce Server runs or to file paths on the network. When you refer to a Windows network path (e.g., a mapped network drive), use the Universal Naming Convention (UNC) syntax. This is necessary because drive letters are not global to the system, and each logon session is assigned its own drive letters.

The UNC has the following syntax: `\\server\sharedfolder\filepath`, where `server` refers to the server name in the network (defined by the DNS); `sharedfolder` refers to a label defined by the administrator (e.g., `admin$` is generally the root directory of the operating system installation); `filepath` refers to the subdirectories below the share.

4.11.1 /system

The `/system` container includes all the FlowForce built-in functions. Only the [abort](#)²³⁵, [compute](#)²³⁷, [compute-string](#)²³⁹ and [create-file](#)²⁴⁰ functions are found directly in this container. Other functions are located in sub-containers according to their area of applicability (for example, AS2 functions, file system functions, mail functions, and so on).

4.11.1.1 abort

Aborts the execution of a job. This function is typically used inside a condition to deliberately end the job when that condition is true. It is the equivalent of a throw or raise function in a programming language. This function does not return a value.

Note: When the `abort` function is included in an [Error/Success Handling block](#)¹³⁶, this function causes the Error/Success block to restart and does not terminate the job.

Parameters

Name	Type	Description
<i>Message</i>	string	Mandatory string parameter. Specifies the message to output when aborting the job.

Examples

In the job below, the `abort` function is used to finish the job with an error if the value of a checked list exceeds 10 items. If the number of items in the list is less than or equal to 10, the job writes the text *The list has less than 10 items* to a file on the local system.

Execution Steps

- + For each in sequence
 - + Choose
 - When
 - + Execute function
 - Parameters: | Message:
 - = Assign this step's result to
 - When
 - + Execute function
 - Parameters: | Command:
 - Working directory:
 - = Assign this step's result to as result
 - +
 - Otherwise
 - +
 - = Assign this step's result to
 - +
 - = Assign this step's result to

4.11.1.2 compute

Full path: /system/compute

Computes the result of an expression and returns the computed value. The computed value can be used in parameters or expressions of other execution steps. You can also use this function to define the output of a job that is used as a service (see the example).

This function returns the value **T0**, which indicates an arbitrary type. That is, the returned data type will be inferred from the expression used in the **Expression** parameter.

Parameters

Name	Type	Description
Expression	Expression of T0	The FlowForce Server expression to be computed. For more information about expressions, see The FlowForce Expression Language ¹⁷¹ .

Examples

This example illustrates a job with two execution steps. The first step runs a shell command in the **c:\temp** directory, and the result is declared as `hello`.

Next, this result is passed to the second execution step. The second execution step uses expression language (in particular, the `stdout` and `content` functions) to do the following:

- get the standard output of the result of the first step
- convert the output to string

The `compute` function evaluates the expression entered in the **Expression** text box.

See also [Creating a "Hello, World" Job](#)²⁹⁶.

4.11.1.3 compute-string

Full path: `/system/compute-string`

Outputs the result of an expression as a string. This step function does essentially the same as the `compute` function, except that the input format is a string instead of an expression.

Parameters

Name	Type	Description
Expression	string	The FlowForce Server expression (as string) to be computed.

Examples

To understand the difference between the `/system/compute/` and `/system/compute-string` functions, consider the following example:

Job Input Parameters

+ Name: Type: Default: + Description:

Execution Steps

+ Execute function `/system/compute`

Parameters: Expression: `concat(filename(inputname), '.txt')`

= Assign this step's result to as T0

+ Execute function `/system/compute-string`

Parameters: Expression: `{filename(inputname)}.txt`

= Assign this step's result to as string

+ Execute function `/system/compute-string`

Parameters: Expression: `Both expressions are {if(outputname1 == outputname2, ', 'not ')}identical.`

In the job illustrated above, there are three execution steps.

The first step calls the `/system/compute/` function. Notice that no curly braces were used. The entire field stores an expression (as suggested by the background color), so curly braces are implied. The expression

concatenates two values and produces a string depending on the job input parameter. For example, if the input parameter is "c:\temp\invoices.txt", the step will return the string value "invoices.txt" (declared as **outputname1**).

The second step calls the `/system/compute-string` function. This function processes a string which contains an embedded FlowForce expression. Here, curly braces are used to delimit the expression from the rest of the string. Notice that the embedded expression has a background color other than the rest of the string. Although a different technique was used, the step result (**outputname2**) is the same as **outputname1**.

Finally, the third step calls the `/system/compute-string` function again, in order to compare the **outputname1** with **outputname2**. If both values are identical, the result will be the string value "Both expression are identical". Otherwise, the result will be "Both expressions are not identical".

4.11.1.4 create-file

The **create-file** function allows you to store stream content in a file that you may need to use in the future. Files created with the help of the **create-file** function are not temporary. Such files belong to the user and not to FlowForce.

The **create-file** function is similar to the [as-file](#)²¹⁴ function in that it creates the specified target file with the specified stream content, but **create-file** does not create any temporary files. Use `/system/create-file` to store stream content that you intend to keep. Use **as-file** to pass the stream content as a file to some program. This might be a temporary file managed by FlowForce.

The source expression (see *screenshot below*) can be anything that returns a stream. You can use anything you can pass to the **as-file** function. For example, you could use the following options:

- [stdout\(result\)](#)²⁰², [stderr\(result\)](#)²⁰³, [result\(result, name, index\)](#)²⁰⁵ get streams out of step results;
- [mime-flatten\(stream\)](#)¹⁹⁷ produces a message/rfc822 stream from another by prefixing it with its MIME headers;
- [mime-multipart\(string, stream*\)](#)¹⁹⁷ produces a MIME multi-part structure as a stream;
- [stream-open\(filename, contenttype\)](#)²⁰¹ opens a file on disk;
- [empty-stream\(\)](#)²⁰² produces a zero-length stream;
- [stream-from-string\(text, encoding, content-type\)](#)²⁰¹ encodes a string value into a stream.

Example

The screenshot below illustrates the **create-file** function. Our goal is to create a file called `CreateTest.txt` and save it on the desktop. We are going to use the [stream-from-string](#)²⁰¹ function, which encodes a string value into a stream. As a result, we will see our new `CreateTest.txt` file containing the string `MyFileContent`.

Note: To run the job, [set a trigger](#)¹⁵¹ and/or run the job as [a service](#)¹⁵⁸.

Execution Steps

+

Execute function /system/create-file

Parameters:	Target file:	CreateTest.txt
	Source stream:	stream-from-string('MyFileContent')
	Overwrite target:	<input checked="" type="checkbox"/>
	Working directory:	Desktop
	Abort on error:	+

4.11.2 /system/filesystem

The `/system/filesystem` container includes functions used to manage files and directories on the operating system where FlowForce Server runs.

Note: All file paths in job execution steps must be paths on the operating system where FlowForce Server runs, not on your local machine.

4.11.2.1 copy

Full path: `/system/filesystem/copy`

Copies a file from a source to a target directory. Optionally, the file can be copied with a new name to the target directory. When invoked from a simple execution step, this function copies one file at a time. To copy multiple files with FlowForce, enclose the step which calls the copy function inside a **For each** step, as illustrated in the [Copy Files](#)³⁰³ example.

Parameters

Name	Type	Description
Source	string as file	The path and file name of the source file that you want to copy.
Target	string as file	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.

Name	Type	Description
Overwrite	<code>boolean</code>	When true , causes the destination file to be overwritten. The default value is false .
Abort on error	<code>boolean</code>	Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code> , job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code> , FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> .
Working directory	<code>string as directory</code>	Specifies the working directory (for example, c:\somedirectory). If relative paths are used, they will be resolved against the working directory.

Examples

See [Copy Files](#)³⁰³.

4.11.2.2 delete

Full path: `/system/filesystem/delete`

This function deletes a file from the path. When invoked from a simple execution step, this function deletes one file at a time. To delete multiple files with FlowForce, enclose the step which calls the delete function inside a **For each** step, as illustrated in the [Copy Files](#)³⁰³ example.

Note: It is not possible for FlowForce to confirm directly from the `delete` function whether a file has been deleted. All FlowForce can do is get a response from the operating system that it is executing a delete command. If the job has subsequent steps that depend upon the deleted file, you will need to check explicitly whether the file still exists. You can use the [list-files](#)²¹⁵ function to check that.

Parameters

Name	Type	Description
Path	<code>string as directory</code>	The path and file name of the file you want to delete.
Abort on error	<code>boolean</code>	Optional parameter. This parameter determines the

Name	Type	Description
		outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code> , job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code> , FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> .
Working directory	<code>string as directory</code>	Specifies the working directory (for example, <code>c:\somedirectory</code>). If relative paths are used, they will be resolved against the working directory.

4.11.2.3 mkdir

Full path: `/system/filesystem/mkdir`

Creates a directory at the specified path.

Parameters

Name	Type	Description
Path	<code>string as directory</code>	The path of the new directory.
Make parents	<code>boolean</code>	Select this check box to create a hierarchical path like <code>c:\dir1\dir2\dir3</code> in one step.
Abort on error	<code>boolean</code>	Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code> , job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code> , FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> .
Working directory	<code>string as directory</code>	Specifies the working directory (for example, <code>c:\somedirectory</code>). If relative paths are used, they will be resolved against the working directory.

Examples

If **Working-Directory** is `c:\temp`, and **Path** is `temp2\temp3`, the function creates the new directory `c:\temp\temp2\temp3`.

4.11.2.4 move

Full path: `/system/filesystem/move`

Moves or renames a file.

When invoked from a simple execution step, this function moves or renames one file at a time. To move or rename multiple files with FlowForce, enclose the step which calls the move function inside a "for-each" step, similar to how this is done in the [Copy Files](#) ³⁰³ example.

Parameters

Name	Type	Description
Source	<code>string as file</code>	The path and file name of the source file that you want to move.
Destination	<code>string as file</code>	The name of the destination directory. If you supply only the directory name in this field, the original file name will be retained.
Overwrite target	<code>boolean</code>	Optional parameter. Set this parameter to <code>true</code> if you want to overwrite destination files with the same names. The default value is <code>false</code> .
Abort on error	<code>boolean</code>	Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code> , job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code> , FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> .
Working directory	<code>string as directory</code>	Specifies the working directory (for example, <code>c:\somedirectory</code>). If relative paths are used, they will be resolved against the working directory.

4.11.2.5 rmdir

Full path: `/system/filesystem/rmdir`

Removes a directory.

Parameters

Name	Type	Description
Path	<code>string as directory</code>	The name of the directory you want to delete.
Abort on error	<code>boolean</code>	Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code> , job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code> , FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> .
Working directory	<code>string as directory</code>	Specifies the working directory (for example, <code>c:\somedirectory</code>). If relative paths are used, they will be resolved against the working directory.

4.11.3 /system/ftp

The `/system/ftp` container includes functions that are used to connect to an FTP or FTPS server and perform operations such as uploading, retrieving, and deleting files, creating and deleting remote directories, and others.

Triggerfile parameter

Jobs with [file system triggers](#)¹⁵⁴ and [HTTP triggers](#)¹⁵⁵ have a job input parameter called *triggerfile* that receives the absolute path of the file (file system triggers) or URI (HTTP triggers) that triggers the job. You can use the *triggerfile* in FTP functions (e.g., to upload the *triggerfile* to an FTP server). If you need only the name of the *triggerfile* with its extension, use the [file-with-extension](#)²¹⁸ expression:

```
{filename-with-extension(triggerfile)}
```

For an example, see the FTP [store](#)²⁷⁰ function.

Wildcards in FTP functions

The following FTP functions accept wildcards as parameters:

- [/system/ftp/delete-wildcard](#)²⁴⁹
- [/system/ftp/retrieve-wildcard](#)²⁶⁴

- [/system/ftp/store-wildcard](#)²⁷⁵
- [/system/ftp/list](#)²⁵²

When you use functions with wildcards, you can enter the following wildcards:

Wildcard	Usage	Example
*	Matches zero or more characters.	*.htm will match <code>home.htm</code> and <code>index.htm</code>
?	Matches any single character.	*.xm? will match <code>index.xml</code> and <code>project.xmi</code>

The `+` (one or more) wildcard is not supported. Instead, you can use `?*`. For example, `*.c?*` will match `.cs`, `.cp` and `.csp` files but will not match `.c` files.

4.11.3.1 delete

Deletes a file from an FTP server.

Parameters

Name	Type	Description
<i>FTP Server</i>	string	Mandatory parameter. Address of a remote FTP server, as a URL or IP address.
<i>Port</i>	number	Optional parameter. The port number used to connect to the FTP server. The default value is 21.
<i>Directory on host</i>	string	Optional parameter. The name of a directory on the remote FTP server, from which you want to delete a file.
<i>Login credentials</i>	credential	Optional parameter. The username and password of the FTP account. For details, see Credentials ¹⁶² . Skip this parameter if the FTP server does not require credentials.
<i>Use passive mode</i>	boolean	Optional parameter. Use the passive mode if connection problems occur (e.g., if routers or firewalls are set up to prevent active connections). The default

Name	Type	Description
<i>Use SSL/TLS encryption</i>	string	<p>value is <code>true</code>.</p> <p>Optional parameter. The default value is <code>No</code>. To transfer information, FTP uses a command channel and a data channel. If you would like to transfer FTP data without encryption, leave the default value. Otherwise, set this value to one of the following:</p> <ul style="list-style-type: none"> • <code>Explicit with encrypted command channel</code> • <code>Explicit with encrypted command and data channel</code> <p>If you set any of the two options above, the validation of the server certificate will depend on the value of the <i>Verify server certificate</i> parameter. Implicit encryption is deprecated and not supported in FlowForce.</p>
<i>Verify server certificate</i>	string	<p>Optional parameter. Specifies how FlowForce should verify the FTP server's certificate. The following options are available:</p> <ul style="list-style-type: none"> • <code>No verification:</code> Accepts any FTP server certificate. • <code>Verify against system certificate store (default value):</code> On Windows, FlowForce Server uses the <i>certificate</i> store of the user account running the job and the <i>system</i> store to verify the certificate signature. On Linux, FlowForce Server uses the system certificate store, usually located in <code>/usr/lib/ssl/cert.pem</code> and <code>/usr/lib/ssl/certs</code>, or the path to which the

Name	Type	Description
		<p>SSL_CERT_FILE and SSL_CERT_DIR environment variables point.</p> <ul style="list-style-type: none"> Verify against selected server certificate: FlowForce compares the FTP server's certificate with the one specified in the <i>Server Certificate</i> parameter. <p>This parameter requires a server certificate and a secure connection. If a secure connection cannot be established, the FTP function will fail.</p>
<i>Server certificate</i>	certificate	<p>Optional parameter. Specifies the path to a FlowForce certificate. If you select <code>Verify against selected server certificate</code> in the <i>Verify server certificate</i> parameter, the FlowForce certificate will be verified against the FTP server certificate. If you select <code>No verification</code> or <code>Verify against system certificate store</code> in the <i>Verify server certificate</i> parameter, the <i>Server certificate</i> parameter value will be ignored.</p>
<i>Target file</i>	string	<p>Mandatory parameter. The name of a file you want to delete from the FTP server.</p> <p>If you use a relative path, it will be resolved against the path specified in the <i>Directory on host</i> parameter. If you use the absolute path, the path in <i>Directory on host</i> will be ignored.</p>
<i>Abort on error</i>	boolean	<p>Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code>, job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code>,</p>

Name	Type	Description
		FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> .
<i>Account</i>	<code>string</code>	Optional parameter. The FTP account name of the user that has access to the files on the remote server.

4.11.3.2 delete-wildcard

Deletes files matching a wildcard (e.g., `*.xml`) from an FTP server. If execution is successful, the function returns a list of deleted files or an empty list if no match has been found. If execution fails, the outcome depends on the *Abort on error* parameter (see *below*).

Parameters

Name	Type	Description
<i>FTP Server</i>	<code>string</code>	Mandatory parameter. Address of a remote FTP server, as a URL or IP address.
<i>Port</i>	<code>number</code>	Optional parameter. The port number used to connect to the FTP server. The default value is 21.
<i>Directory on host</i>	<code>string</code>	Optional parameter. The name of a directory on the remote FTP server, from which you want to delete files that match a specific wildcard.
<i>Login credentials</i>	<code>credential</code>	Optional parameter. The username and password of the FTP account. For details, see Credentials ¹⁶² . Skip this parameter if the FTP server does not require credentials.
<i>Use passive mode</i>	<code>boolean</code>	Optional parameter. Use the passive mode if connection problems occur (e.g., if routers or firewalls are set up to prevent active connections). The default value is <code>true</code> .

Name	Type	Description
<i>Use SSL/TLS encryption</i>	string	<p>Optional parameter. The default value is <code>No</code>. To transfer information, FTP uses a command channel and a data channel. If you would like to transfer FTP data without encryption, leave the default value. Otherwise, set this value to one of the following:</p> <ul style="list-style-type: none"> • <code>Explicit with encrypted command channel</code> • <code>Explicit with encrypted command and data channel</code> <p>If you set any of the two options above, the validation of the server certificate will depend on the value of the <i>Verify server certificate</i> parameter. Implicit encryption is deprecated and not supported in FlowForce.</p>
<i>Verify server certificate</i>	string	<p>Optional parameter. Specifies how FlowForce should verify the FTP server's certificate. The following options are available:</p> <ul style="list-style-type: none"> • <code>No verification:</code> Accepts any FTP server certificate. • <code>Verify against system certificate store (default value):</code> On Windows, FlowForce Server uses the <i>certificate</i> store of the user account running the job and the <i>system</i> store to verify the certificate signature. On Linux, FlowForce Server uses the system certificate store, usually located in <code>/usr/lib/ssl/cert.pem</code> and <code>/usr/lib/ssl/certs</code>, or the path to which the <code>SSL_CERT_FILE</code> and

Name	Type	Description
		<p>SSL_CERT_DIR environment variables point.</p> <ul style="list-style-type: none"> Verify against selected server certificate: FlowForce compares the FTP server's certificate with the one specified in the <i>Server Certificate</i> parameter. <p>This parameter requires a server certificate and a secure connection. If a secure connection cannot be established, the FTP function will fail.</p>
<i>Server certificate</i>	certificate	<p>Optional parameter. Specifies the path to a FlowForce certificate. If you select <code>Verify against selected server certificate</code> in the <i>Verify server certificate</i> parameter, the FlowForce certificate will be verified against the FTP server certificate. If you select <code>No verification</code> or <code>Verify against system certificate store</code> in the <i>Verify server certificate</i> parameter, the <i>Server certificate</i> parameter value will be ignored.</p>
<i>Wildcard</i>	string	<p>Optional parameter. Specifies a wildcard, for example, <code>*.xml</code>. Any files matching the wildcard will be deleted from the FTP server. See also Wildcards in FTP functions²⁴⁵.</p> <p>If you use a relative path, it will be resolved against the path specified in the <i>Directory on host</i> parameter. If you use the absolute path, the path in <i>Directory on host</i> will be ignored.</p>
<i>Abort on error</i>	boolean	<p>Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If this parameter is <code>false</code>, the function will return the</p>

Name	Type	Description
		list of directory names that have been deleted successfully and omit those file names that cannot be deleted for some reason. If this parameter is <code>true</code> , the job execution will be aborted in the first file that cannot be deleted. The default value is <code>true</code> .
<i>Account</i>	string	Optional parameter. The FTP account name of the user that has access to the files on the remote server.

4.11.3.3 list

Lists the contents of a directory on an FTP server.

Parameters

Name	Type	Description
<i>FTP Server</i>	string	Mandatory parameter. Address of a remote FTP server, as a URL or IP address.
<i>Port</i>	number	Optional parameter. The port number used to connect to the FTP server. The default value is 21.
<i>Directory on host</i>	string	Optional parameter. The name of a directory (on the remote FTP server), whose contents you want to list.
<i>Login credentials</i>	credential	Optional parameter. The username and password of the FTP account. For details, see Credentials ¹⁶² . Skip this parameter if the FTP server does not require credentials.
<i>Use passive mode</i>	boolean	Optional parameter. Use the passive mode if connection problems occur (e.g., if routers or firewalls are set up to prevent active connections). The default

Name	Type	Description
<i>Use SSL/TLS encryption</i>	string	<p>value is <code>true</code>.</p> <p>Optional parameter. The default value is <code>No</code>. To transfer information, FTP uses a command channel and a data channel. If you would like to transfer FTP data without encryption, leave the default value. Otherwise, set this value to one of the following:</p> <ul style="list-style-type: none"> • <code>Explicit with encrypted command channel</code> • <code>Explicit with encrypted command and data channel</code> <p>If you set any of the two options above, the validation of the server certificate will depend on the value of the <i>Verify server certificate</i> parameter. Implicit encryption is deprecated and not supported in FlowForce.</p>
<i>Verify server certificate</i>	string	<p>Optional parameter. Specifies how FlowForce should verify the FTP server's certificate. The following options are available:</p> <ul style="list-style-type: none"> • <code>No verification:</code> Accepts any FTP server certificate. • <code>Verify against system certificate store (default value):</code> On Windows, FlowForce Server uses the <i>certificate</i> store of the user account running the job and the <i>system</i> store to verify the certificate signature. On Linux, FlowForce Server uses the system certificate store, usually located in <code>/usr/lib/ssl/cert.pem</code> and <code>/usr/lib/ssl/certs</code>, or the path to which the

Name	Type	Description
		<p>SSL_CERT_FILE and SSL_CERT_DIR environment variables point.</p> <ul style="list-style-type: none"> Verify against selected server certificate: FlowForce compares the FTP server's certificate with the one specified in the <i>Server Certificate</i> parameter. <p>This parameter requires a server certificate and a secure connection. If a secure connection cannot be established, the FTP function will fail.</p>
<i>Server certificate</i>	certificate	<p>Optional parameter. Specifies the path to a FlowForce certificate. If you select <code>Verify against selected server certificate</code> in the <i>Verify server certificate</i> parameter, the FlowForce certificate will be verified against the FTP server certificate. If you select <code>No verification</code> or <code>Verify against system certificate store</code> in the <i>Verify server certificate</i> parameter, the <i>Server certificate</i> parameter value will be ignored.</p>
<i>Wildcard</i>	string	<p>Optional parameter. If you want to list only files with a specific pattern, you can use a wildcard (e.g., *.js). See also Wildcards in FTP functions²⁴⁵.</p> <p>If you use a relative path, it will be resolved against the path specified in the <i>Directory on host</i> parameter. If you use the absolute path, the path in <i>Directory on host</i> will be ignored.</p>
<i>Abort on error</i>	boolean	<p>Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code>, job execution</p>

Name	Type	Description
		will be terminated. If the <i>Abort on error</i> parameter is <code>false</code> , FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> .
<i>Account</i>	string	Optional parameter. The FTP account name of the user that has access to the files on the remote server.

4.11.3.4 mkdir

Creates a directory on an FTP server.

Parameters

Name	Type	Description
<i>FTP Server</i>	string	Mandatory parameter. Address of a remote FTP server, as a URL or IP address.
<i>Port</i>	number	Optional parameter. The port number used to connect to the FTP server. The default value is 21.
<i>Directory on host</i>	string	Optional parameter. The name of a directory on the remote FTP server, where you want to create a directory.
<i>Login credentials</i>	credential	Optional parameter. The username and password of the FTP account. For details, see Credentials ¹⁶² . Skip this parameter if the FTP server does not require credentials.
<i>Use passive mode</i>	boolean	Optional parameter. Use the passive mode if connection problems occur (e.g., if routers or firewalls are set up to prevent active connections). The default value is <code>true</code> .
<i>Use SSL/TLS encryption</i>	string	Optional parameter. The default

Name	Type	Description
		<p>value is <code>No</code>. To transfer information, FTP uses a command channel and a data channel. If you would like to transfer FTP data without encryption, leave the default value. Otherwise, set this value to one of the following:</p> <ul style="list-style-type: none"> • <code>Explicit with encrypted command channel</code> • <code>Explicit with encrypted command and data channel</code> <p>If you set any of the two options above, the validation of the server certificate will depend on the value of the <i>Verify server certificate</i> parameter. Implicit encryption is deprecated and not supported in FlowForce.</p>
<i>Verify server certificate</i>	<code>string</code>	<p>Optional parameter. Specifies how FlowForce should verify the FTP server's certificate. The following options are available:</p> <ul style="list-style-type: none"> • <code>No verification:</code> Accepts any FTP server certificate. • <code>Verify against system certificate store (default value):</code> On Windows, FlowForce Server uses the <i>certificate</i> store of the user account running the job and the <i>system</i> store to verify the certificate signature. On Linux, FlowForce Server uses the system certificate store, usually located in <code>/usr/lib/ssl/cert.pem</code> and <code>/usr/lib/ssl/certs</code>, or the path to which the <code>SSL_CERT_FILE</code> and <code>SSL_CERT_DIR</code>

Name	Type	Description
		<p>environment variables point.</p> <ul style="list-style-type: none"> Verify against selected server certificate: FlowForce compares the FTP server's certificate with the one specified in the <i>Server Certificate</i> parameter. <p>This parameter requires a server certificate and a secure connection. If a secure connection cannot be established, the FTP function will fail.</p>
<i>Server certificate</i>	certificate	<p>Optional parameter. Specifies the path to a FlowForce certificate. If you select <code>Verify against selected server certificate</code> in the <i>Verify server certificate</i> parameter, the FlowForce certificate will be verified against the FTP server certificate. If you select <code>No verification</code> or <code>Verify against system certificate store</code> in the <i>Verify server certificate</i> parameter, the <i>Server certificate</i> parameter value will be ignored.</p>
<i>Target directory</i>	string	<p>Mandatory parameter. The name of a directory you want to create on the FTP server.</p> <p>If you use a relative path, it will be resolved against the path specified in the <i>Directory on host</i> parameter. If you use the absolute path, the path in <i>Directory on host</i> will be ignored.</p>
<i>Abort on error</i>	boolean	<p>Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code>, job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code>, FlowForce Server will ignore errors and continue job execution. The</p>

Name	Type	Description
		default value is <code>true</code> .
<i>Account</i>	string	Optional parameter. The FTP account name of the user that has access to the files on the remote server.

4.11.3.5 move

Moves a file (*Source file*) on an FTP server to another location (*Target file*) on the same FTP server.

Parameters

Name	Type	Description
<i>FTP Server</i>	string	Mandatory parameter. Address of a remote FTP server, as a URL or IP address.
<i>Port</i>	number	Optional parameter. The port number used to connect to the FTP server. The default value is 21.
<i>Directory on host</i>	string	Optional parameter. The name of a directory on the remote FTP server, where you want to move a file.
<i>Login credentials</i>	credential	Optional parameter. The username and password of the FTP account. For details, see Credentials ¹⁶² . Skip this parameter if the FTP server does not require credentials.
<i>Use passive mode</i>	boolean	Optional parameter. Use the passive mode if connection problems occur (e.g., if routers or firewalls are set up to prevent active connections). The default value is <code>true</code> .
<i>Use SSL/TLS encryption</i>	string	Optional parameter. The default value is <code>No</code> . To transfer information, FTP uses a command channel and a data channel. If you would like to transfer FTP data without

Name	Type	Description
		<p>encryption, leave the default value. Otherwise, set this value to one of the following:</p> <ul style="list-style-type: none"> • Explicit with encrypted command channel • Explicit with encrypted command and data channel <p>If you set any of the two options above, the validation of the server certificate will depend on the value of the <i>Verify server certificate</i> parameter. Implicit encryption is deprecated and not supported in FlowForce.</p>
<i>Verify server certificate</i>	string	<p>Optional parameter. Specifies how FlowForce should verify the FTP server's certificate. The following options are available:</p> <ul style="list-style-type: none"> • No verification: Accepts any FTP server certificate. • Verify against system certificate store (<i>default value</i>): On Windows, FlowForce Server uses the <i>certificate</i> store of the user account running the job and the <i>system</i> store to verify the certificate signature. On Linux, FlowForce Server uses the system certificate store, usually located in <code>/usr/lib/ssl/cert.pem</code> and <code>/usr/lib/ssl/certs</code>, or the path to which the <code>SSL_CERT_FILE</code> and <code>SSL_CERT_DIR</code> environment variables point. • Verify against selected server certificate: FlowForce

Name	Type	Description
		<p>compares the FTP server's certificate with the one specified in the <i>Server Certificate</i> parameter.</p> <p>This parameter requires a server certificate and a secure connection. If a secure connection cannot be established, the FTP function will fail.</p>
<i>Server certificate</i>	certificate	<p>Optional parameter. Specifies the path to a FlowForce certificate. If you select <i>Verify against selected server certificate</i> in the <i>Verify server certificate</i> parameter, the FlowForce certificate will be verified against the FTP server certificate. If you select <i>No verification or Verify against system certificate store</i> in the <i>Verify server certificate</i> parameter, the <i>Server certificate</i> parameter value will be ignored.</p>
<i>Source file</i>	string	<p>Mandatory parameter. The name of a file that you want to move.</p> <p>If you use a relative path, it will be resolved against the path specified in the <i>Directory on host</i> parameter. If you use the absolute path, the path in <i>Directory on host</i> will be ignored.</p>
<i>Target file</i>	string	<p>Mandatory parameter. The name of the moved file at the target location.</p> <p>If you use a relative path, it will be resolved against the path specified in the <i>Directory on host</i> parameter. If you use the absolute path, the path in <i>Directory on host</i> will be ignored.</p>
<i>Abort on error</i>	boolean	<p>Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i></p>

Name	Type	Description
		parameter is <code>true</code> , job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code> , FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> .
<i>Account</i>	<code>string</code>	Optional parameter. The FTP account name of the user that has access to the files on the remote server.

4.11.3.6 retrieve

Downloads a file from an FTP server to a local directory.

Parameters

Name	Type	Description
<i>FTP Server</i>	<code>string</code>	Mandatory parameter. Address of a remote FTP server, as a URL or IP address.
<i>Port</i>	<code>number</code>	Optional parameter. The port number used to connect to the FTP server. The default value is 21.
<i>Directory on host</i>	<code>string</code>	Optional parameter. The name of a directory on the remote FTP server from which you want to retrieve (download) a file.
<i>Login credentials</i>	<code>credential</code>	Optional parameter. The username and password of the FTP account. For details, see Credentials ¹⁶² . Skip this parameter if the FTP server does not require credentials.
<i>Use passive mode</i>	<code>boolean</code>	Optional parameter. Use the passive mode if connection problems occur (e.g., if routers or firewalls are set up to prevent active connections). The default value is <code>true</code> .

Name	Type	Description
<i>Use SSL/TLS encryption</i>	string	<p>Optional parameter. The default value is <code>No</code>. To transfer information, FTP uses a command channel and a data channel. If you would like to transfer FTP data without encryption, leave the default value. Otherwise, set this value to one of the following:</p> <ul style="list-style-type: none"> • <code>Explicit with encrypted command channel</code> • <code>Explicit with encrypted command and data channel</code> <p>If you set any of the two options above, the validation of the server certificate will depend on the value of the <i>Verify server certificate</i> parameter. Implicit encryption is deprecated and not supported in FlowForce.</p>
<i>Verify server certificate</i>	string	<p>Optional parameter. Specifies how FlowForce should verify the FTP server's certificate. The following options are available:</p> <ul style="list-style-type: none"> • <code>No verification:</code> Accepts any FTP server certificate. • <code>Verify against system certificate store (default value):</code> On Windows, FlowForce Server uses the <i>certificate</i> store of the user account running the job and the <i>system</i> store to verify the certificate signature. On Linux, FlowForce Server uses the system certificate store, usually located in <code>/usr/lib/ssl/cert.pem</code> and <code>/usr/lib/ssl/certs</code>, or the path to which the <code>SSL_CERT_FILE</code> and

Name	Type	Description
		<p>SSL_CERT_DIR environment variables point.</p> <ul style="list-style-type: none"> Verify against selected server certificate: FlowForce compares the FTP server's certificate with the one specified in the <i>Server Certificate</i> parameter. <p>This parameter requires a server certificate and a secure connection. If a secure connection cannot be established, the FTP function will fail.</p>
<i>Server certificate</i>	certificate	<p>Optional parameter. Specifies the path to a FlowForce certificate. If you select <code>Verify against selected server certificate</code> in the <i>Verify server certificate</i> parameter, the FlowForce certificate will be verified against the FTP server certificate. If you select <code>No verification</code> or <code>Verify against system certificate store</code> in the <i>Verify server certificate</i> parameter, the <i>Server certificate</i> parameter value will be ignored.</p>
<i>Source file</i>	string	<p>Mandatory parameter. The name of a file you want to download from the FTP server.</p> <p>If you use a relative path, it will be resolved against the path specified in the <i>Directory on host</i> parameter. If you use the absolute path, the path in <i>Directory on host</i> will be ignored.</p>
<i>Target file</i>	string	<p>Mandatory parameter. The name of the downloaded file in a local directory.</p> <p>If you use a relative path, it will be resolved against the path specified in the <i>Working directory</i> parameter. If you use the absolute</p>

Name	Type	Description
		path, the path in <i>Working directory</i> will be ignored.
<i>Overwrite target</i>	boolean	Optional parameter. Set this parameter to <code>true</code> if you want to overwrite destination files with the same names. The default value is <code>false</code> .
<i>Abort on error</i>	boolean	Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code> , job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code> , FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> .
<i>Working directory</i>	string	Optional parameter. Specifies the local working directory where the file downloaded from the FTP server will be stored.
<i>Account</i>	string	Optional parameter. The FTP account name of the user that has access to the files on the remote server.

4.11.3.7 retrieve-wildcard

Downloads files from an FTP server to a local directory if the files match a wildcard (e.g., `*.xml`). If execution is successful, the function returns a list of files (absolute local paths) or an empty list if no match has been found. If execution fails, the outcome depends on the *Abort on error* parameter (see *below*).

Parameters

Name	Type	Description
<i>FTP Server</i>	string	Mandatory parameter. Address of a remote FTP server, as a URL or IP address.
<i>Port</i>	number	Optional parameter. The port number used to connect to the FTP server. The default value is 21.

Name	Type	Description
<i>Directory on host</i>	string	Optional parameter. The name of a directory on the remote FTP server, from which you want to download files that match a particular wildcard.
<i>Login credentials</i>	credential	Optional parameter. The username and password of the FTP account. For details, see Credentials ¹⁶² . Skip this parameter if the FTP server does not require credentials.
<i>Use passive mode</i>	boolean	Optional parameter. Use the passive mode if connection problems occur (e.g., if routers or firewalls are set up to prevent active connections). The default value is <code>true</code> .
<i>Use SSL/TLS encryption</i>	string	<p>Optional parameter. The default value is <code>NO</code>. To transfer information, FTP uses a command channel and a data channel. If you would like to transfer FTP data without encryption, leave the default value. Otherwise, set this value to one of the following:</p> <ul style="list-style-type: none"> • <code>Explicit with encrypted command channel</code> • <code>Explicit with encrypted command and data channel</code> <p>If you set any of the two options above, the validation of the server certificate will depend on the value of the <i>Verify server certificate</i> parameter. Implicit encryption is deprecated and not supported in FlowForce.</p>
<i>Verify server certificate</i>	string	<p>Optional parameter. Specifies how FlowForce should verify the FTP server's certificate. The following options are available:</p> <ul style="list-style-type: none"> • <code>No verification:</code> Accepts any FTP server

Name	Type	Description
		<p>certificate.</p> <ul style="list-style-type: none"> • Verify against system certificate store (<i>default value</i>): On Windows, FlowForce Server uses the <i>certificate</i> store of the user account running the job and the <i>system</i> store to verify the certificate signature. On Linux, FlowForce Server uses the system certificate store, usually located in <code>/usr/lib/ssl/cert.pem</code> and <code>/usr/lib/ssl/certs</code>, or the path to which the <code>SSL_CERT_FILE</code> and <code>SSL_CERT_DIR</code> environment variables point. • Verify against selected server certificate: FlowForce compares the FTP server's certificate with the one specified in the <i>Server Certificate</i> parameter. <p>This parameter requires a server certificate and a secure connection. If a secure connection cannot be established, the FTP function will fail.</p>
<i>Server certificate</i>	certificate	<p>Optional parameter. Specifies the path to a FlowForce certificate. If you select <i>Verify against selected server certificate</i> in the <i>Verify server certificate</i> parameter, the FlowForce certificate will be verified against the FTP server certificate. If you select <i>No verification</i> or <i>Verify against system certificate store</i> in the <i>Verify server certificate</i> parameter, the <i>Server certificate</i> parameter value will be ignored.</p>

Name	Type	Description
<i>Wildcard</i>	string	Optional parameter. Specifies a wildcard, for example, *.xml. Any files matching the wildcard will be downloaded from the FTP server to a local directory. See also Wildcards in FTP functions ²⁴⁵ . If you use a relative path, it will be resolved against the path specified in the <i>Directory on host</i> parameter. If you use the absolute path, the path in <i>Directory on host</i> will be ignored.
<i>Abort on error</i>	boolean	Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code> , job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code> , FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> . Note that some files may still be retrieved even if execution fails.
<i>Working directory</i>	string	Optional parameter. Specifies the local working directory where the files downloaded from the FTP server will be stored.
<i>Account</i>	string	Optional parameter. The FTP account name of the user that has access to the files on the remote server.

4.11.3.8 rmdir

Deletes a directory from an FTP server.

Parameters

Name	Type	Description
<i>FTP Server</i>	string	Mandatory parameter. Address of

Name	Type	Description
		a remote FTP server, as a URL or IP address.
<i>Port</i>	number	Optional parameter. The port number used to connect to the FTP server. The default value is 21.
<i>Directory on host</i>	string	Optional parameter. The path on the remote FTP server, from which you want to delete a directory.
<i>Login credentials</i>	credential	Optional parameter. The username and password of the FTP account. For details, see Credentials ¹⁶² . Skip this parameter if the FTP server does not require credentials.
<i>Use passive mode</i>	boolean	Optional parameter. Use the passive mode if connection problems occur (e.g., if routers or firewalls are set up to prevent active connections). The default value is <code>true</code> .
<i>Use SSL/TLS encryption</i>	string	<p>Optional parameter. The default value is <code>No</code>. To transfer information, FTP uses a command channel and a data channel. If you would like to transfer FTP data without encryption, leave the default value. Otherwise, set this value to one of the following:</p> <ul style="list-style-type: none"> • <code>Explicit with encrypted command channel</code> • <code>Explicit with encrypted command and data channel</code> <p>If you set any of the two options above, the validation of the server certificate will depend on the value of the <i>Verify server certificate</i> parameter. Implicit encryption is deprecated and not supported in FlowForce.</p>
<i>Verify server certificate</i>	string	Optional parameter. Specifies how

Name	Type	Description
		<p>FlowForce should verify the FTP server's certificate. The following options are available:</p> <ul style="list-style-type: none"> • No verification: Accepts any FTP server certificate. • Verify against system certificate store (<i>default value</i>): On Windows, FlowForce Server uses the <i>certificate</i> store of the user account running the job and the <i>system</i> store to verify the certificate signature. On Linux, FlowForce Server uses the system certificate store, usually located in <code>/usr/lib/ssl/cert.pem</code> and <code>/usr/lib/ssl/certs</code>, or the path to which the <code>SSL_CERT_FILE</code> and <code>SSL_CERT_DIR</code> environment variables point. • Verify against selected server certificate: FlowForce compares the FTP server's certificate with the one specified in the <i>Server Certificate</i> parameter. <p>This parameter requires a server certificate and a secure connection. If a secure connection cannot be established, the FTP function will fail.</p>
<i>Server certificate</i>	certificate	Optional parameter. Specifies the path to a FlowForce certificate. If you select <i>Verify against selected server certificate</i> in the <i>Verify server certificate</i> parameter, the FlowForce certificate will be verified against the FTP server certificate. If you

Name	Type	Description
		select No verification or Verify against system certificate store in the <i>Verify server certificate</i> parameter, the <i>Server certificate</i> parameter value will be ignored.
<i>Target directory</i>	string	Mandatory parameter. The name of a directory you want to delete from the FTP server. If you use a relative path, it will be resolved against the path specified in the <i>Directory on host</i> parameter. If you use the absolute path, the path in <i>Directory on host</i> will be ignored.
<i>Abort on error</i>	boolean	Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code> , job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code> , FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> .
<i>Account</i>	string	Optional parameter. The FTP account name of the user that has access to the files on the remote server.

4.11.3.9 store

Uploads a file from a local directory to an FTP server.

Parameters

Name	Type	Description
<i>FTP Server</i>	string	Mandatory parameter. Address of a remote FTP server, as a URL or IP address.
<i>Port</i>	number	Optional parameter. The port number used to connect to the

Name	Type	Description
		FTP server. The default value is 21.
<i>Directory on host</i>	string	Optional parameter. The name of a directory on the remote FTP server, to which you want to upload a file.
<i>Login credentials</i>	credential	Optional parameter. The username and password of the FTP account. For details, see Credentials ¹⁶² . Skip this parameter if the FTP server does not require credentials.
<i>Use passive mode</i>	boolean	Optional parameter. Use the passive mode if connection problems occur (e.g., if routers or firewalls are set up to prevent active connections). The default value is <code>true</code> .
<i>Use SSL/TLS encryption</i>	string	<p>Optional parameter. The default value is <code>No</code>. To transfer information, FTP uses a command channel and a data channel. If you would like to transfer FTP data without encryption, leave the default value. Otherwise, set this value to one of the following:</p> <ul style="list-style-type: none"> • <code>Explicit with encrypted command channel</code> • <code>Explicit with encrypted command and data channel</code> <p>If you set any of the two options above, the validation of the server certificate will depend on the value of the <i>Verify server certificate</i> parameter. Implicit encryption is deprecated and not supported in FlowForce.</p>
<i>Verify server certificate</i>	string	Optional parameter. Specifies how FlowForce should verify the FTP server's certificate. The following options are available:

Name	Type	Description
		<ul style="list-style-type: none"> • No verification: Accepts any FTP server certificate. • Verify against system certificate store (<i>default value</i>): On Windows, FlowForce Server uses the <i>certificate</i> store of the user account running the job and the <i>system</i> store to verify the certificate signature. On Linux, FlowForce Server uses the system certificate store, usually located in <code>/usr/lib/ssl/cert.pem</code> and <code>/usr/lib/ssl/certs</code>, or the path to which the <code>SSL_CERT_FILE</code> and <code>SSL_CERT_DIR</code> environment variables point. • Verify against selected server certificate: FlowForce compares the FTP server's certificate with the one specified in the <i>Server Certificate</i> parameter. <p>This parameter requires a server certificate and a secure connection. If a secure connection cannot be established, the FTP function will fail.</p>
<i>Server certificate</i>	certificate	Optional parameter. Specifies the path to a FlowForce certificate. If you select <i>Verify against selected server certificate</i> in the <i>Verify server certificate</i> parameter, the FlowForce certificate will be verified against the FTP server certificate. If you select <i>No verification</i> or <i>Verify against system certificate store</i> in the <i>Verify server certificate</i> parameter, the

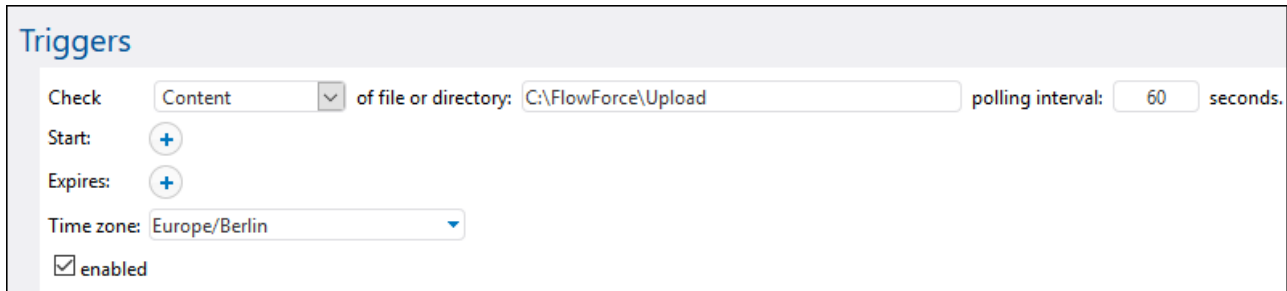
Name	Type	Description
		<i>Server certificate</i> parameter value will be ignored.
<i>Source file</i>	string	Mandatory parameter. The name of a local file you want to upload to the FTP server. If you use a relative path, it will be resolved against the path specified in the <i>Working directory</i> parameter. If you use the absolute path, the path in <i>Working directory</i> will be ignored.
<i>Target file</i>	string	Mandatory parameter. The name of the uploaded file that will be stored on the FTP server. If you use a relative path, it will be resolved against the path specified in the <i>Directory on host</i> parameter. If you use the absolute path, the path in <i>Directory on host</i> will be ignored.
<i>Abort on error</i>	boolean	Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code> , job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code> , FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> .
<i>Working directory</i>	string	Optional parameter. Specifies the local working directory, from which a file will be uploaded to the FTP server.
<i>Account</i>	string	Optional parameter. The FTP account name of the user that has access to the files on the remote server.

Example

This example shows you how to upload a file from a local directory to a remote FTP server, without knowing the file name and extension at job configuration time.

Set a trigger

To upload a file without knowing its name and extension, we need to set [a file system trigger](#)¹⁵⁴. The trigger shown below monitors the `C:\FlowForce\Upload` directory for changes. Whenever there is a change in this directory, the job fires, and the absolute path of the file that triggered the job becomes available in the *triggerfile* input parameter (*subsection below*).



The screenshot shows a configuration window titled "Triggers". It contains the following fields and controls:

- Check:** A dropdown menu set to "Content", followed by the text "of file or directory:" and a text input field containing "C:\FlowForce\Upload".
- polling interval:** A text input field containing "60" followed by the text "seconds".
- Start:** A plus sign icon (+).
- Expires:** A plus sign icon (+).
- Time zone:** A dropdown menu set to "Europe/Berlin".
- enabled:** A checked checkbox.

Add an execution step

The execution step below calls the `store` function to upload a file from the local directory (`C:\FlowForce\Upload`) to the `uploads` directory on the FTP server. Every time a change is detected in the local working directory, the job will fire, and the `store` function will upload the file in which the change has occurred to the FTP server. Since we have already indicated the local working directory, it is sufficient to use the relative path of the triggerfile in the *Source file* parameter. For the target file, we must use the relative path of the triggerfile so that it is resolved correctly against the directory on the FTP server.

In the *Source file* and *Target file* parameters, we use the [file-with-extension](#)²¹⁸ function that takes the absolute path of the triggerfile and extracts the file name and its extension. For example, if the triggerfile were `C:\FlowForce\Upload\Example.txt`, the expression would return `Example.txt`.

Job Input Parameters

+ Name: Type: Default:

+

Execution Steps

+

Execute function

Parameters:	FTP Server: <input type="text" value="10.100.63.200"/>
	Port: <input type="text" value="21"/>
	Directory on host: <input type="text" value="uploads"/>
	Login credentials: <input checked="" type="radio"/> Select existing credential: <input type="text" value="/public/my.ftp.credentials"/>
	<input type="radio"/> Define local credential:
	Use passive mode: <input style="border: 1px solid #ccc; border-radius: 50%; width: 20px; height: 20px; display: inline-block; vertical-align: middle;" type="text" value="+"/>
	Source file: <input type="text" value="{filename-with-extension(triggerfile)}"/>
	Target file: <input type="text" value="{filename-with-extension(triggerfile)}"/>
	Abort on error: <input style="border: 1px solid #ccc; border-radius: 50%; width: 20px; height: 20px; display: inline-block; vertical-align: middle;" type="text" value="+"/>
	Working directory: <input type="text" value="C:\FlowForce\Upload"/>
	Account: <input style="border: 1px solid #ccc; border-radius: 50%; width: 20px; height: 20px; display: inline-block; vertical-align: middle;" type="text" value="+"/>

4.11.3.10 store-wildcard

Uploads files from a local directory to an FTP server if the files match a wildcard (e.g., *.xml). If execution is successful, the function returns a list of uploaded files (absolute local paths) or an empty list if no match has been found. If execution fails, the outcome depends on the *Abort on error* parameter (see below).

Parameters

Name	Type	Description
<i>FTP Server</i>	string	Mandatory parameter. Address of a remote FTP server, as a URL or IP address.
<i>Port</i>	number	Optional parameter. The port number used to connect to the FTP server. The default value is

Name	Type	Description
		21.
<i>Directory on host</i>	string	Optional parameter. The name of a directory on the remote FTP server, to which you want to upload files that match a particular wildcard.
<i>Login credentials</i>	credential	Optional parameter. The username and password of the FTP account. For details, see Credentials ¹⁶² . Skip this parameter if the FTP server does not require credentials.
<i>Use passive mode</i>	boolean	Optional parameter. Use the passive mode if connection problems occur (e.g., if routers or firewalls are set up to prevent active connections). The default value is <code>true</code> .
<i>Use SSL/TLS encryption</i>	string	Optional parameter. The default value is <code>No</code> . To transfer information, FTP uses a command channel and a data channel. If you would like to transfer FTP data without encryption, leave the default value. Otherwise, set this value to one of the following: <ul style="list-style-type: none"> • <code>Explicit with encrypted command channel</code> • <code>Explicit with encrypted command and data channel</code> <p>If you set any of the two options above, the validation of the server certificate will depend on the value of the <i>Verify server certificate</i> parameter. Implicit encryption is deprecated and not supported in FlowForce.</p>
<i>Verify server certificate</i>	string	Optional parameter. Specifies how FlowForce should verify the FTP server's certificate. The following options are available:

Name	Type	Description
		<ul style="list-style-type: none"> • No verification: Accepts any FTP server certificate. • Verify against system certificate store (<i>default value</i>): On Windows, FlowForce Server uses the <i>certificate</i> store of the user account running the job and the <i>system</i> store to verify the certificate signature. On Linux, FlowForce Server uses the system certificate store, usually located in <code>/usr/lib/ssl/cert.pem</code> and <code>/usr/lib/ssl/certs</code>, or the path to which the <code>SSL_CERT_FILE</code> and <code>SSL_CERT_DIR</code> environment variables point. • Verify against selected server certificate: FlowForce compares the FTP server's certificate with the one specified in the <i>Server Certificate</i> parameter. <p>This parameter requires a server certificate and a secure connection. If a secure connection cannot be established, the FTP function will fail.</p>
<i>Server certificate</i>	certificate	Optional parameter. Specifies the path to a FlowForce certificate. If you select <i>Verify against selected server certificate</i> in the <i>Verify server certificate</i> parameter, the FlowForce certificate will be verified against the FTP server certificate. If you select <i>No verification</i> or <i>Verify against system certificate store</i> in the <i>Verify server certificate</i> parameter, the

Name	Type	Description
		<i>Server certificate</i> parameter value will be ignored.
<i>Wildcard</i>	string	Optional parameter. Specifies a wildcard, for example, *.xml. Any local files matching the wildcard will be uploaded to the FTP server. See also Wildcards in FTP functions ²⁴⁵ . If you use a relative path, it will be resolved against the path specified in the <i>Working directory</i> parameter. If you use the absolute path, the path in <i>Working directory</i> will be ignored.
<i>Abort on error</i>	boolean	Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code> , job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code> , FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> . Note that some files may still be uploaded even if execution fails.
<i>Working directory</i>	string	Mandatory parameter. The local working directory from which files matching the wildcard will be uploaded to the FTP server.
<i>Account</i>	string	Optional parameter. The FTP account name of the user that has access to the files on the remote server.

4.11.4 /system/mail

The `/system/mail` container includes the [send](#)²⁷⁹ and [send-mime](#)²⁸⁰ functions that are used to send email.

4.11.4.1 send

Sends an email to the specified recipients, generally the administrator. Before using this function, you must [configure the mail server settings](#)¹¹⁸. For an example that uses the `send` function, see [Adding Error Handling to a Job](#)³²⁵.

Parameters

Name	Type	Description
<i>From</i>	string	Mandatory parameter. The sender's email address (e.g., <code>flowforce@<hostname></code>).
<i>To</i>	string	Mandatory parameter. The recipient's email address. This field may contain a comma-separated list of email addresses.
<i>Subject</i>	string	Mandatory parameter. The subject line of a message.
<i>Message body</i>	string	Optional parameter. The body text of a message. The message body supports ASCII and Unicode characters. The text box for the message body allows entering multiple lines and expressions in curly braces <code>{ }</code> .
<i>Attachment</i>	string as file	Optional parameter. The file name of an attachment.
<i>Abort on error</i>	boolean	Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code> , job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code> , FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> .

4.11.4.2 send-mime

The `send-mime` function sends an email to the specified recipients, generally to the administrator. Before using this function, you must [configure the mail server settings](#)¹¹⁸. Unlike the `send`²⁷⁹ function, the *Message body* parameter of this function enables you to get the message body (e.g., as HTML) from a stream.

To create a stream for the message body directly in FlowForce, you can call expression functions such as `stream-open`²⁰¹ or `stream-from-string`²⁰¹. You can also use [MIME expression functions](#)¹⁸⁸ to customize email or attachment message headers. To prevent an email from landing into the Junk folder of the recipient, you should create MIME headers in accordance with the requirements of the receiving server or program.

To get HTML content for the message body, it is recommended to call a [StyleVision Server](#) transformation that produces HTML output as MIME. For an example that shows how to deploy a StyleVision transformation to FlowForce Server, see [Create a Job from a StyleVision Transformation](#)³⁵¹. For more information about StyleVision Server integration, see [Integration with Other Altova Servers](#)⁴⁰⁵.

FlowForce Server does not collect any images, stylesheets, or similar resources referenced by HTML files into a MIME stream. To create the HTML message body with StyleVision Server, take the steps below:

1. Design the HTML body of an email in [Altova StyleVision](#). The design may contain local images and stylesheets.
2. Deploy the StyleVision transformation to FlowForce Server. In FlowForce, the transformation becomes a built-in FlowForce function that can be executed by StyleVision Server.
3. Create a job that calls the StyleVision Server transformation above, making sure to select the *GenerateHtmlOutputAsMime* option on the job configuration page.
4. On the job configuration page, call FlowForce Server expression functions to pick up the generated MIME stream and pass it to the *Message body* parameter of the `send-mime` function (see [Example 1 below](#)).

If any external resources referenced by the HTML file cannot be embedded into the MIME stream, they will be added as attachments to the email.

Parameters

Name	Type	Description
<i>From</i>	string	Mandatory parameter. The sender's email address (e.g., <code>flowforce@<hostname></code>).
<i>To</i>	string	Mandatory parameter. The recipient's email address. This field can contain a comma-separated list of email addresses.
<i>Subject</i>	string	Mandatory parameter. Subject line of a message.
<i>Message body</i>	stream	Mandatory parameter. Body text of a message.

Name	Type	Description
<i>Attachment</i>	sequence of stream	Optional parameter. Attachment(s) sent with an email. Each attachment must be a FlowForce expression that produces a stream. Call stream functions ¹⁸⁸ to create streams from strings or files. Call MIME expression functions ¹⁸⁸ to add, modify, or delete MIME headers.
<i>Abort on error</i>	boolean	Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code> , job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code> , FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> .

Example 1

The job illustrated below calls the `send-mime` function to send an email in HTML format.

Step 1

The first execution step generates HTML output by calling a StyleVision Server transformation. This transformation was designed with StyleVision and then deployed to FlowForce Server. The function uses `BiggestCities.xml` as an input file. The output file is `BiggestCitiesPerCity.html` that will be generated as a MIME type (the *GenerateHtmlOutputAsMime* check box is selected). The result of this execution step is called `output` and will be used in the next step.

Step 2

The second execution step calls the `compute` function to compute the `nth`²⁰⁹ (`results`²⁰⁵(`output`), 0) expression that picks up the MIME stream from the result generated by the previous step. The result of the second step is called `message` and will be used in the *Message body* parameter in the last execution step.

Step 3

The third execution step sends an email that has the result of the previous step (`message`) as the message body.

Execution Steps

- Execute function `/public/BiggestCitiesPerCity.transformation`

Parameters:

 - InputXml: `altova://packagedfile/BiggestCities.xml`
 - OutHtml: `BiggestCitiesPerCity.html`
 - GenerateHtmlOutputAsMime:
 - OutRtf:
 - OutFo:
 - OutPdf:
 - OutDocx:
 - Working-directory: `C:\FlowForce`

= Assign this step's result to `output` as `ReturnTypeRtf, ReturnTypeMime, ReturnTypeDocx, ReturnType`
- Execute function `/system/compute`

Parameters:

 - Expression: `nth(results(output), 0)`

= Assign this step's result to `message` as `T0`
- Execute function `/system/mail/send-mime`

Parameters:

 - From: `flowforce@yourhostname`
 - To: `to@example.org`
 - Subject: `Example mail`
 - Message body: `message`
 - Attachment:
 - Abort on error:

= Assign this step's result to `name` as `boolean`

Example 2

The job illustrated below calls the `send-mime` function to send an email in HTML format. The email contains an image attachment in `.png` format.

The first execution step prepares the HTML code for the message body. For simplicity, the HTML code in this example is typed directly in the text box. The recommended way to get HTML output is to call a StyleVision Server transformation, as illustrated in the previous example. The result of this step is a string called `body_html` that will be used in the next step.

The second execution step creates the body of the email. The [stream-from-string](#)²⁰¹ function converts the result of the first execution step to a stream. The encoding (UTF-8) and MIME type (text/html) are supplied as arguments to the function. The result of this step is `message` of type T0 (any type) and will be used in the message body in the last step.

The third execution step creates an attachment of the email, also as a stream. The step computes the following expression:

```
add-mime-header(stream-open('C:\sample.png', 'image/png'), 'Content-
Disposition', 'attachment; filename=sample.png')
```

The [stream-open](#)²⁰¹ function opens the image as a stream. The [add-mime-header](#)¹⁹¹ function adds the `Content-Disposition` header to the stream. The value of the `Content-Disposition` header is `'attachment; filename=sample.png'`, which indicates that `sample.png` can be downloaded and saved locally. The result of this step is `attachment` that will be used in the last step.

The last step sends an email. The `send-mime` function uses the result of the second step (`message`) as the message body. In the `Attachment` parameter, the [mime-content-encode](#)¹⁹⁶ function is required, because `sample.png` is a binary file. Such files have to be encoded as Base64 in order to be preserved during transmission.

Execution Steps

1. Execute function `/system/compute-string`
 - Parameters: Expression: `Hello from FlowForce Server` as string (required) Set to ▶
 - = Assign this step's result to `body_html` as string
2. Execute function `/system/compute`
 - Parameters: Expression: `stream-from-string(body_html, 'UTF-8', 'text/html')` as expression of T0 (required)
 - = Assign this step's result to `message` as T0
3. Execute function `/system/compute`
 - Parameters: Expression: `add-mime-header(stream-open('C:\sample.png', 'image/png'), 'Content-Disposition', 'attachment; filename=sample.png')` as
 - = Assign this step's result to `attachment` as T0
4. Execute function `/system/mail/send-mime`
 - Parameters: From: `flowforce@example.org` as string
 - To: `someone@example.org` as string
 - Subject: `Binary file as attachment` as string
 - Message body: `message` as expression of stream
 - Attachment:
 - `mime-content-encode(attachment, 'base64')` Set to ▶

Another approach

Another approach to the job described above is to use the [set-mime-content-disposition](#)¹⁹⁵ function instead of the [add-mime-header](#)¹⁹¹ function (*third execution step above*). The expression will look as follows:

```
set-mime-content-disposition(stream-open('C:
\sample.png', 'image/png'), 'attachment', 'sample.png')
```

4.11.5 /system/maintenance

The `/system/maintenance` container includes functions used to perform maintenance operations on the server.

4.11.5.1 archive-log

Full path: `/system/maintenance/archive-log`

Moves the older log records to an archive file on the server. Returns the name of the archive file that was created, as string value.

Parameters

Name	Type	Description
Older than, days	<code>number</code>	Archives files older than the number of days entered here. The default value is 30 .
Archive directory	<code>string</code>	Archive directory name, (for example, c:\temp). Mandatory.
Archive file prefix	<code>string</code>	Specifies the prefix of the archive file. The default value is flowforcelog .
Delete archived records	<code>boolean</code>	Select this check box to delete archived records from the FlowForce Server database.
Working directory	<code>string</code>	Specifies the working directory of the job (for example, c:\somedirectory). If relative paths are used, they will be resolved against the working directory.

4.11.5.2 cleanup-files

Full path: `/system/maintenance/cleanup-files`

Deletes those files that are not in use or referenced by any deployed objects (such as MapForce mappings and StyleVision transformations). Returns the number of files that were deleted, as numeric value.

When you delete deployed objects, or when you re-deploy existing objects with modified files, any files associated with previously deployed objects become unused. By default, FlowForce Server does not delete the

unused files. Therefore, in order to clean up the disk space, it is strongly recommended to create a job which periodically calls this function, especially in enterprise environments where multiple users deploy objects to FlowForce Server.

To see the current disk space used by deployed objects, check the size of the *files* folder located in the FlowForce Server [instance-data folder](#)¹⁸.

This function does not have any parameters.

4.11.5.3 truncate-log

Full path: `/system/maintenance/truncate-log`

Deletes log records older than the date supplied. Returns the number of records that were deleted, as numeric value.

Parameters

Name	Type	Description
Older than, days	number	Truncates (deletes) records older than the number of days entered here. The default value is 30.

4.11.6 /system/shell

The `/system/shell` container includes the [commandline](#)²⁸⁵ function, which is used to execute shell commands or scripts.

4.11.6.1 commandline

Full path: `/system/shell/commandline`

Executes a shell command or a batch file.

To have FlowForce Server jobs read environment variables, they must be defined in scripts, and those scripts must be executed with the `/system/shell/commandline` function. Be aware that FlowForce Server is running a non-interactive shell, which means all behavior specific to interactive shells is not applicable (such as executing `.profile` or `.bashrc` on Linux).

If the exit code from the last shell command is other than "0", the outcome is as follows:

- If the parameter **Abort on error** is **true** (default), this function aborts execution. In this case, you can handle the error by means of protected blocks (see [Handling Step Errors](#)¹³⁶).

- If the parameter **Abort on error** is **false**, the function returns the result of the shell command, including the standard output, the standard error, and the exit code.

If the exit code from the last command is "0" (success), the function returns the result of the last shell command, as generic type. To handle the value returned by this function in another step or job, do the following:

1. Name the returned result by entering a value in the **Assign this step's result to** text box (for example, "myresult").
2. Create a new step which executes either the function `compute` or `compute-string`, depending on what return type you need.
3. Enter as argument to the above function an expression which gets the desired part from the generic result. For example, enter the expression `stdout(myresult)` to get the standard output of the result as stream, and `stderr(myresult)` to get the standard error output stream. To get the same values as string, use `content(stdout(myresult))` and `content(stderr(myresult))`, respectively.

Note that the `stdout` function (and the job) will fail if the shell command does not return a standard output. Likewise, the `stderr` function will fail if there is no standard error.

See also [Handling Data Types in Steps](#)¹⁴⁵ and [Step Result Functions](#)²⁰².

Parameters

Name	Type	Description
Command	<code>string</code>	Enter the shell command to execute.
Abort on error	<code>boolean</code>	Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code> , job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code> , FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> .
Working directory	<code>string as directory</code>	Specifies the working directory of the job (for example, <code>c:\somedirectory</code>). If relative paths are used, they will be resolved against the working directory.

Examples

The following job executes a Windows batch file called **DoTransform.bat**. Assuming that the **DoTransform.bat** requires some XML file as input, the input XML file must be copied to the working directory. In this example, the working directory is **C:\codegen\xslt2**.

Execution Steps

+

- Execute function /system/shell/commandline
 - Parameters:
 - Command: DoTransform.bat
 - Abort on error: **+**
 - Working directory: C:\codegen\xslt2
 - = Assign this step's result to as result

The following job calls RaptorXML Server to run an XSLT transformation with parameters. It is assumed that the PATH environment variable contains the path to the RaptorXML Server executable, for example **C:\Program Files (x86)\Altova\RaptorXMLServer2024\bin**. For more information about RaptorXML Server, see <https://www.altova.com/raptorxml>.

on /system/shell/commandline

Command: raptorxml xslt --input=c:/Test.xml --output=c:/Output.xml c:/Test.xslt

For a step-by-step example which handles the output returned by the command line, see [Check if a path exists](#)²⁹⁹.

4.12 Import/Export Configuration Data

You can [export jobs and other configuration objects](#)²⁸⁸ (including deployed MapForce mappings and StyleVision transformations) from FlowForce Server as follows:

- To another running FlowForce Server instance (online export)
- To a file (offline export)

When you export objects to another running FlowForce Server instance, the exported objects become immediately available in the Web administration interface of that server.

When you export objects to a file, FlowForce Server creates a `.zip` archive which contains the selected objects and their dependencies. The `.zip` archive has the following naming convention: `export_YYMMDDhhmmss`. For example, a file exported on the 6th of August 2018-2024 at 10:51:33 server time would be named `export_2018-20240806105133.zip`.

You can subsequently [import the .zip archive](#)²⁹² into the same FlowForce Server instance (provided the imported objects no longer exist at the destination, or you want to overwrite them), or into another instance.

4.12.1 Export Configuration Data

You can export specific records within a container or entire containers. Before exporting objects, FlowForce informs you on a separate page about all objects that are dependent on or referenced by the objects that you wish to export. This enables you to see if there are missing dependencies. If you export objects to a running FlowForce Server, you can also see whether each object already exists on the destination server.

By default, FlowForce Server does not export the following categories of sensitive data:

- Passwords defined locally in jobs
- Passwords available as standalone credential records
- Passwords stored with system functions (such as [/system/ftp](#)²⁴⁵ functions)
- OAuth 2.0 client secret, authorization token and refresh token (*Advanced Edition*)
- Private keys in a certificate and private key pair (*Advanced Edition*)

To export sensitive data, select the *Export sensitive data* check box during export. Be aware that, if you select the check box, the exported archive will include the sensitive data in plain text format.

If you do not select the check box, the sensitive data will not be exported. In this case, upon importing data back into FlowForce Server, you have the option to overwrite each individual record or skip it. If you choose to overwrite, the existing sensitive data will be replaced with empty values.

How to export configuration data

To export configuration data, take the steps below:

1. Open the Configuration page of the Web administration interface and select the records you want to export. You can select specific records in a container or the entire container.
2. Click **Export Selected Objects**.

3. In the **Export Selected Objects** dialog, you can choose to export your configuration data to a running FlowForce Server instance (*Export to server*) or to a file with a `.zip` extension. When you switch to the *Export to server* option (red rectangle below), you will need to enter the host name of the destination FlowForce Server, the port where it runs, your user name and password on the destination FlowForce Server instance, and then click **Export**.

Export Selected Objects

Name

my.credentials

ConvertProducts.mapping

ConvertProducts.job

Export to server

Server: Port:

User:

Password: Use SSL:

Export to file

Export **Cancel**

4. After you have selected the relevant export option, the Export page will open and display all records to be exported. This page enables you to include/exclude records and to view the records' dependencies (screenshot below). The screenshot below illustrates the Export page after the *Export to server* option has been selected.

Export

<input type="checkbox"/> Name	Type ↕	Dependencies	Remote Server
<input type="checkbox"/> /public/my.credentials	credential	Referenced by: /public/ConvertProducts.job	Exists on remote server
<input checked="" type="checkbox"/> /public/ConvertProducts.mapping	MapForce mapping	Referenced by: /public/ConvertProducts.job	Will be created on remote server
<input checked="" type="checkbox"/> /public/ConvertProducts.job	job	Depends on: /public/ConvertProducts.map /public/my.credentials /system/filesystem/copy	Will be created on remote server
<input type="checkbox"/> /system/filesystem/copy	function	Referenced by: /public/ConvertProducts.job	Exists on remote server

Export options

Export sensitive data:

Downgrade credentials for FlowForce 2019r3 or earlier:





The records with a yellow background are those that are about to be exported. The records that are grayed out represent dependencies on built-in system functions, so you cannot take actions on them.

The *Remote Server* column indicates if the file exists at the destination. If the dependencies already exist at the destination, you can safely exclude such records from the export. Otherwise, if you export without dependencies and the dependencies do not exist at the destination, such jobs will likely fail.

The *Export sensitive data* option enables you to optionally include passwords in the exported package. For security reasons, it is not recommended to select this check box unless you really need to transfer such sensitive data in plain text out of FlowForce Server. For details about exporting sensitive information, see the subsection *How to Export Sensitive Data* below.

The *Downgrade credentials for FlowForce 2019r3 or earlier* check box must be selected if the exported list includes records of type `credential` and if the target FlowForce Server is of version 2019r3 or earlier. After that release, credentials got new "Allow usage" options, and so the check box makes it possible to make newer credential records compatible with older versions of FlowForce. For more information about *Allow usage* options, see [Credentials](#)¹⁶².

The screenshot below shows the Export page after the *Export to file* option has been selected. In this case, it is not possible to determine whether the exported objects exist at the destination. For this reason, the *Current State* column shows the following information: "Might or might not exist on remote server". If the dependencies exist at the destination when you import the `.zip` archive back into FlowForce, you can safely exclude the dependencies from the export. If you are not sure, choose to export all dependencies. Otherwise, when you later attempt to import data, the import might fail.

Export			
<input type="checkbox"/> Name	Type ↕	Dependencies	Current State
<input type="checkbox"/>  /public/my.credentials	credential	Referenced by:  /public/ConvertProducts.job	Might or might not exist on remote server
<input checked="" type="checkbox"/>  /public/ConvertProducts.mapping	MapForce mapping	Referenced by:  /public/ConvertProducts.job	Might or might not exist on remote server

- After you have selected the relevant export option and the records you wish to export, click the **Export** button.

How to export sensitive data

When you export data from FlowForce Server, you can choose whether to include sensitive data in the exported archive. The examples below describe two possible approaches.

Example 1: Exclude sensitive data






This screenshot below shows a job (`AddNumbers`) which refers to a credential record (`my.credentials`). The credential is located in the same container as the job.

<input type="checkbox"/> Name	Type ↕
<input type="checkbox"/>  my.credentials	credential
<input type="checkbox"/>  AddNumbers	job

If you choose to export both objects to a file (`.zip` archive) without selecting the *Export sensitive data* option, the following happens:

- The job will be exported.
- The credential record will be exported without the password.

If you later import the `.zip` archive into a FlowForce Server environment where the two objects do not exist, both objects will be created successfully. Note that the password associated with the credential record will be empty. If the objects already exist in the target environment, you can overwrite them or clear the corresponding check box and skip them. The screenshot below shows that the objects in the target environment will be overwritten.

Import			
<input type="checkbox"/> Name	Type ↕	Dependencies	Current State
<input checked="" type="checkbox"/>  /public/Jobs/my.credentials	credential	Referenced by:  /public/Jobs/AddNumbers	Will be overwritten
<input checked="" type="checkbox"/>  /public/Jobs/AddNumbers	job	Depends on:  /public/Jobs/my.credentials  /system/compute-string	Will be overwritten

If you choose to overwrite both records, the following happens:

1. The job existing in FlowForce Server will be overwritten by the job from the `.zip` archive.
2. The credential record existing in FlowForce Server will be overwritten by the one from `.zip` archive, and the destination password will become empty.

If you do not overwrite the credential, the existing credential will remain untouched.

Example 2: Include sensitive data

In this example, the same two records as above have been selected for export, and the *Export sensitive data* option has been enabled. In this case, the following happens:

1. The job will be exported.
2. The credential record will be exported and will include the password.

If you later import the `.zip` archive into a FlowForce Server environment where the two objects do not exist, both objects will be created successfully. The password associated with the credential record will be the one from the `.zip` file.

If the objects already exist in the target environment, you can overwrite them or clear the corresponding check box and skip them. If you choose to overwrite the records, the following happens:








1. The job existing in FlowForce Server will be overwritten by the job from the `.zip` archive.
2. The credential record existing in FlowForce Server will be overwritten by the one from `.zip` archive. The destination password will also be overwritten by the one from the `.zip` archive.

If you do not overwrite the credential, the existing credential will remain untouched.

4.12.2 Import Configuration Data

To import an archive exported previously, follow the instructions below:

1. Open the Configuration page and click **Import Objects**.
2. Click **Choose File** in the **Import Objects** dialog and select a `.zip` archive that you previously exported from FlowForce Server.
3. Click **Import**. FlowForce Server displays the records that are about to be imported, along with their dependencies (*screenshot below*). The *Current State* column informs about what will happen to each record after you click the **Import** button.

Import			
<input type="checkbox"/> Name	Type ↕	Dependencies	Current State
<input checked="" type="checkbox"/>  /public/my.credentials	credential	Referenced by:  /public/Cleanup	Will be overwritten
<input checked="" type="checkbox"/>  /public/Cleanup	job	Depends on:  /public/my.credentials  /system/maintenance/cleanup-files	Will be created
<input type="checkbox"/>  /system/maintenance/cleanup-files	function	Referenced by:  /public/Cleanup	Exists

4. Click **Import**.

Missing dependencies

When you export data from FlowForce Server, be aware that some objects may have dependencies on other objects. If you do not export dependencies together with the object that depends on them, this may lead to errors when you later import that data back into FlowForce Server.

If the archive you are importing has external dependencies that cannot be found in the target instance of FlowForce Server, the *Current State* column will display the status `Does not exist`.

5 Job Examples

This chapter includes step-by-step FlowForce job configuration examples. The table below lists all the examples, along with the specific function kinds and triggers illustrated in each example.

Example	Concepts illustrated		
	Built-in functions	Expression functions	Trigger
Create a "Hello, World!" Job ²⁹⁶	<ul style="list-style-type: none"> /system/compute 		Web service
Check if a Path Exists ²⁹⁹	<ul style="list-style-type: none"> /system/shell/commandline /system/compute-string 	content() stdout() trim()	Web service
Copy Files ³⁰³	<ul style="list-style-type: none"> /system/filesystem/copy 	list-files()	Web service
Create a Job from a MapForce Mapping ³⁰⁸	<ul style="list-style-type: none"> MapForce mapping 		Timer
Use a Job as Step of Another Job ³¹⁶	<ul style="list-style-type: none"> /system/filesystem/copy 		
Create a Directory Polling Job ³¹⁹	<ul style="list-style-type: none"> MapForce mapping /system/filesystem/move 		File system
Add Error Handling to a Job ³²⁵	<ul style="list-style-type: none"> /system/shell/commandline /system/mail/send 	failed-step() error-message() exitcode() stdout() stderr() content() instance-id()	Web service
Expose a Job as a Web Service ³³¹	<ul style="list-style-type: none"> MapForce mapping 		Web service
Post JSON to FlowForce Web Service ³³⁹	<ul style="list-style-type: none"> /system/filesystem/copy 	as-file() instance-id()	Web service
Cache Job Results ³⁴⁷	<ul style="list-style-type: none"> /system/shell/commandline /system/compute 	stdout()	Web service
Create a Job from a StyleVision Transformation ³⁵¹	<ul style="list-style-type: none"> StyleVision transformation /system/compute /system/filesystem/copy 	results() nth() as-file()	Timer
Validate a Document with RaptorXML ³⁵⁹	<ul style="list-style-type: none"> /RaptorXML/valany 		Timer
Validate XML with Error Logging ³⁶¹	<ul style="list-style-type: none"> /RaptorXML/valxml-w ithxsd /system/compute /system/filesystem/copy 	failed-step() stdout() as-file()	Web service
Run XSLT with RaptorXML ³⁶⁶	<ul style="list-style-type: none"> /RaptorXML/xslt 	list()	

Example	Concepts illustrated		
	Built-in functions	Expression functions	Trigger
Generate PDFs from XML Files ³⁷¹	<ul style="list-style-type: none">• MapForce mapping• StyleVision transformation• /system/compute	as-file() results() filename()	Web service

5.1 Create a "Hello, World!" Job

This example shows you how to create a simple job that outputs the text "Hello, World!" in the browser. The text will be created by means of a FlowForce expression. You will be able to trigger the job on demand by clicking a link in the browser (that is, the job will be exposed as a Web service).

Prerequisites

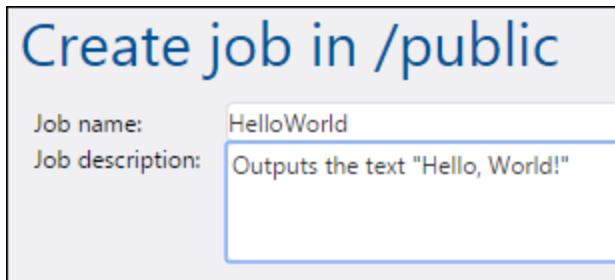
- Required licenses: FlowForce Server
- The *FlowForce Web Server* and *FlowForce Server* services must be listening at the configured [network address and port](#) ⁴⁷
- You have a FlowForce Server user account with permissions to one of the containers (by default, the **/public** container used in this example is accessible to any authenticated user).

Creating the job

1. Log on to FlowForce Server and navigate to the **/public** container.
2. Click **Create | Create Container** and create a new container called "Examples".

The **/public/Examples** container is used by convention in most of the jobs illustrated in this documentation. You can create your jobs in any other containers as well, but if you want to follow all the subsequent tutorials from this documentation literally, it is recommended to create the **/public/Examples** container.

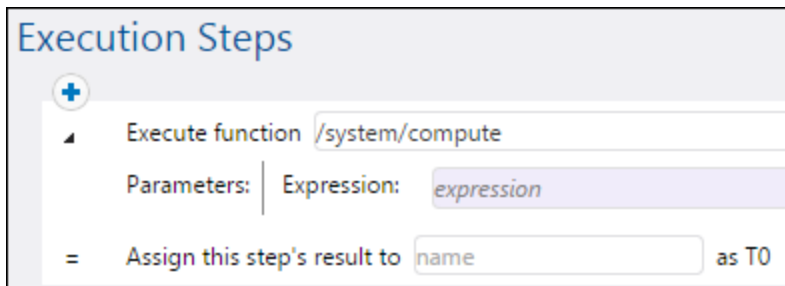
3. In the **/public/Examples** container, click **Create | Create job**, and enter the job title and description.



Job name: HelloWorld

Job description: Outputs the text "Hello, World!"

4. Add a new execution step which calls the built-in function [/system/compute](#) ²³⁷.



Execute function [/system/compute](#)

Parameters: Expression: `expression`

= Assign this step's result to `name` as T0

5. In the **Expression** field, enter the text 'Hello, World', enclosed within single quotes. The content of this field represents a FlowForce Server expression.


6. Declare the execution result as **string**.

7. Select the **Make this job available via HTTP...** check box and type "HelloWorldService" as service name. For more information, see [Exposing Jobs as Web Services](#)¹⁵⁸.

8. Under "Credentials", select an existing credential record or specify a local credential. For more information, see [Credentials](#)¹⁶².
9. Click **Save**.

Running the job

You have now finished creating a job that computes the string value "Hello, World!" and returns it as the job result. To run the job, do one of the following:

- Go to **Home**, and then click **Show all active triggers and services**. Next, click the job's URL displayed in the "Info" column.
- Enter <http://127.0.0.1:4646/service/HelloWorldService> in the browser's address bar. Note that this URL works only if the *FlowForce Server* service listens at the default host address and port name. If you have defined other host and port settings in the [Configuration page](#)⁴⁷, change the address accordingly.
- If you set the optional **Host name** field of FlowForce Server from the [Setup Page](#)⁴⁷, you can execute the web service call directly from the job configuration page, by clicking the  button adjacent to the **Make this job available via HTTP** check box. The button is not displayed otherwise.

If prompted for credentials when accessing the Web service, supply the same credentials you use to log on to FlowForce Server.

Supplying your FlowForce Server user credentials for HTTP authentication is only for testing purposes. For production, it is recommended that you create a new FlowForce user, grant the **Service - Use** permission to this user on the container where the job is, and then access the Web service with the corresponding

user account. To disable HTTP authentication and make the Web service public, grant the **Service - Use** permission to the user **Anonymous**, see [How Permissions Work](#)⁹².

If the job executes successfully, the browser displays the output of the job:

Hello, World!

If the job fails, the browser displays a "Service execution failed" message. In this case, check the FlowForce Server [job log](#)¹⁰⁴ to identify the error.

5.2 Check if a Path Exists

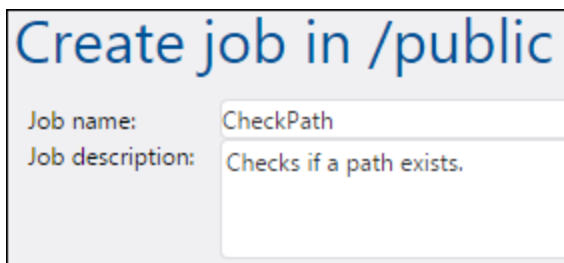
This example shows you how to create a job which informs you if a path (to a file or directory) exists on the operating system. To achieve this goal, you will use a combination of built-in functions and expression functions. The job will be defined as a Web service, so that you can trigger it on demand, by accessing a URL from the browser. The job will take the path as an argument, and will return a string which informs whether the path supplied as argument exists on the operating system where FlowForce Server runs.

Prerequisites

- Required licenses: FlowForce Server
- The *FlowForce Web Server* and *FlowForce Server* services must be listening at the configured [network address and port](#) ⁴⁷
- You have a FlowForce Server user account with permissions to one of the containers (by default, the **/public** container used in this example is accessible to any authenticated user).

Creating the job


1. Log on to FlowForce Server and navigate to a container where you have permission to create new jobs. For consistency with other examples, this tutorial uses the **/public/Examples** container—if you don't have this container yet, create it using the **Create | Create Container** command.
2. In the **/public/Examples** container, click **Create**, and then select **Create job**.
3. Add a job name ("CheckPath", in this example) and, optionally, a job description.

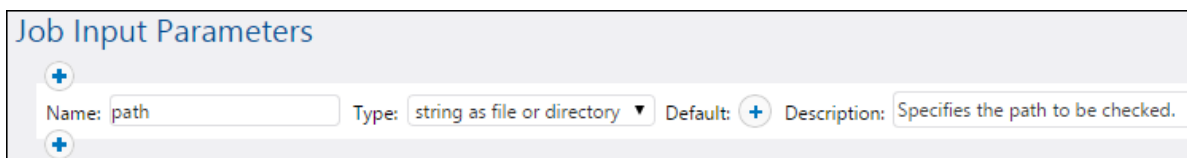


Create job in /public

Job name: CheckPath

Job description: Checks if a path exists.


4. Under Job Input Parameters, click , and add the parameter **path**, as shown below.



Job Input Parameters



+ Name: path Type: string as file or directory Default: + Description: Specifies the path to be checked.

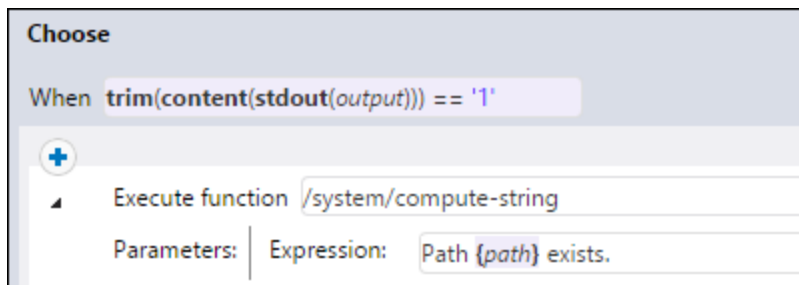
+


5. Add a new execution step which calls the  [/system/shell/commandline](#) ²⁸⁵ function, and enter the shell command which checks for the existence of the file. Make sure to declare the result of this step, as shown below (in this example, we called it **output**).

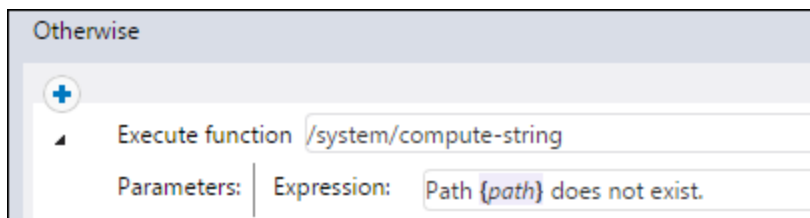


On Windows, the shell command outputs "1" when the path exists and "0" when it does not exist. If FlowForce Server runs on a Unix system, adjust the command accordingly. Notice that the command embeds the FlowForce expression `{path}`. This expression references the input parameter defined in the previous step.

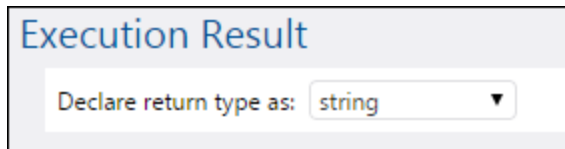
6. Under "Execution Steps", click the  button, and then select **new Choose step**. Then enter `trim(content(stdout(output))) == '1'` as condition expression. This expression consists of three nested functions: `stdout`, `content`, and `trim`. First, the `stdout` function gets the standard output of the result returned by the previous step. Then the `content` function converts the standard output to string. Finally, the `trim` function removes any leading or trailing spaces, carriage returns, or line feeds from the standard output. The result is then compared to "1" using the equality operator. If both values are equal, the path exists. Otherwise, the path does not exist.
7. Under the **When** clause, add an execution step as shown below. This execution step calls the  [/system/compute-string](#)²³⁹ function to build the string value that should be returned when the path exists. Notice that the value embeds the FlowForce expression `{path}`. This expression references the input parameter defined in a previous step.



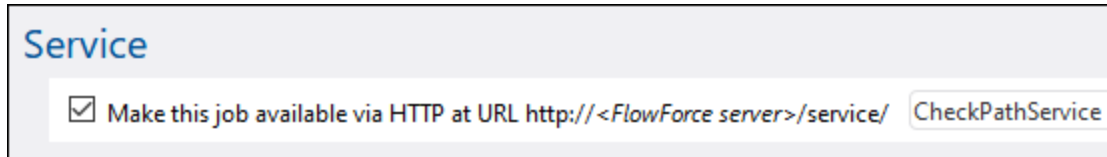
8. Under the **Otherwise** clause, add an execution step as shown below. This execution step calls the  [/system/compute-string](#)²³⁹ function to build the string value that should be returned when the path does not exist. Notice that the value embeds the FlowForce expression `{path}`. This expression references the input parameter defined in a previous step.



9. Under Execution Result, declare the return type as **string**.




- Under Service, click to select the **Make this job available via HTTP** check box, and enter **CheckPathService** as name of the service. For more information, see [Exposing Jobs as Web Services](#)¹⁵⁸.



- Under "Credentials", select an existing credential record or specify a local credential. For more information, see [Credentials](#)¹⁶².
- Click **Save**.

Running the job

To run the job, do one of the following:

- Go to **Home**, and then click **Show all active triggers and services**. Next, click the job's URL displayed in the "Info" column.
- Enter <http://127.0.0.1:4646/service/CheckPathService> in the browser's address bar. Note that this URL works only if the *FlowForce Server* service listens at the default host address and port name. If you have defined other host and port settings in the [Configuration page](#)⁴⁷, change the address accordingly.
- If you set the optional **Host name** field of FlowForce Server from the [Setup Page](#)⁴⁷, you can execute the web service call directly from the job configuration page, by clicking the  button adjacent to the **Make this job available via HTTP** check box. The button is not displayed otherwise.

If prompted for credentials when accessing the Web service, supply the same credentials you use to log on to FlowForce Server.

Supplying your FlowForce Server user credentials for HTTP authentication is only for testing purposes. For production, it is recommended that you create a new FlowForce user, grant the **Service - Use** permission to this user on the container where the job is, and then access the Web service with the corresponding user account. To disable HTTP authentication and make the Web service public, grant the **Service - Use** permission to the user **Anonymous**, see [How Permissions Work](#)⁹².

Since this job has arguments, you will be prompted to supply them when you access the Web service in the browser.

Parameters

path *:

If the job executes successfully, the browser displays the output of the job, for example:

```
Path C:\ exists.
```

If the job fails, the browser displays a "Service execution failed" message. In this case, check the log of the job in FlowForce Server to identify the error, see [Viewing the Job Log](#)¹⁰⁴.

5.3 Copy Files

This example shows you how to copy multiple files on the local file system with the help of a FlowForce Server job.

Let's assume that you would like to copy all the files from directory **C:\FlowForceExamples\CopyFiles\Source** to a new directory **C:\FlowForceExamples\CopyFiles\Target**. (On a UNIX system, please adjust the paths accordingly.) To achieve the goal, we will use a "for-each" step that iterates through all the files in a directory, and then invoke the [/system/filesystem/copy](#)²⁴¹ function for each item in the loop.

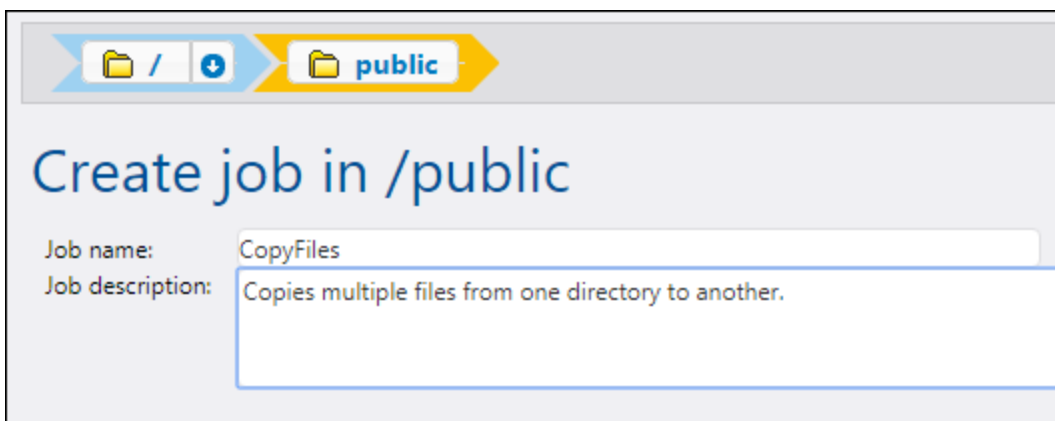
Prerequisites

- Required licenses: FlowForce Server
- The *FlowForce Web Server* and *FlowForce Server* services must be listening at the configured [network address and port](#)⁴⁷
- You have a FlowForce Server user account with permissions to one of the containers (by default, the **/public** container used in this example is accessible to any authenticated user).
- This job copies files from directory **C:\FlowForceExamples\CopyFiles\Source** to directory **C:\FlowForceExamples\CopyFiles\Target**. Make sure to create these directories on the local file system before creating the job. Also, make sure that the source directory contains a few files to test the job.

Creating the job

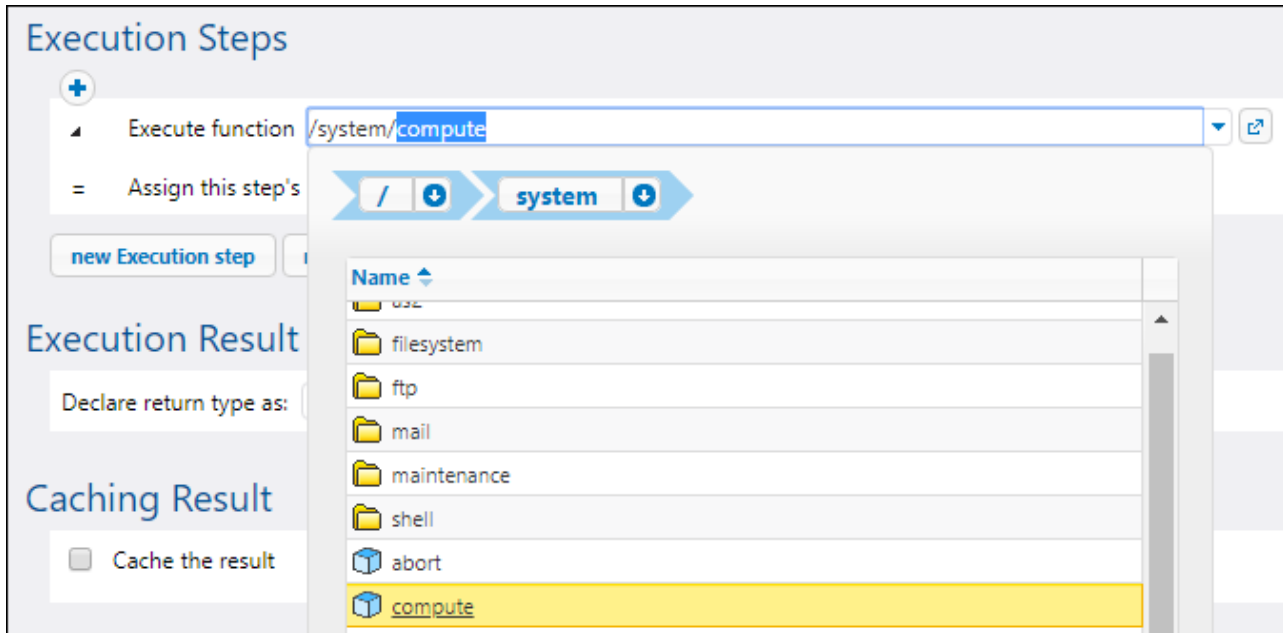
Log on to FlowForce Server and navigate to a container where you have permission to create new jobs. For consistency with other examples, this tutorial uses the **/public/Examples** container—if you don't have this container yet, create it using the **Create | Create Container** command.

In the **/public/Examples** container, create a new job. Enter a job name (for example, "CopyFiles"), and, optionally, a job description.



The screenshot shows a web interface for creating a job. At the top, there is a breadcrumb navigation bar with a folder icon, a slash, a plus icon, another folder icon, and the text 'public'. Below this, the main heading is 'Create job in /public'. There are two input fields: 'Job name:' with the text 'CopyFiles' and 'Job description:' with the text 'Copies multiple files from one directory to another.'

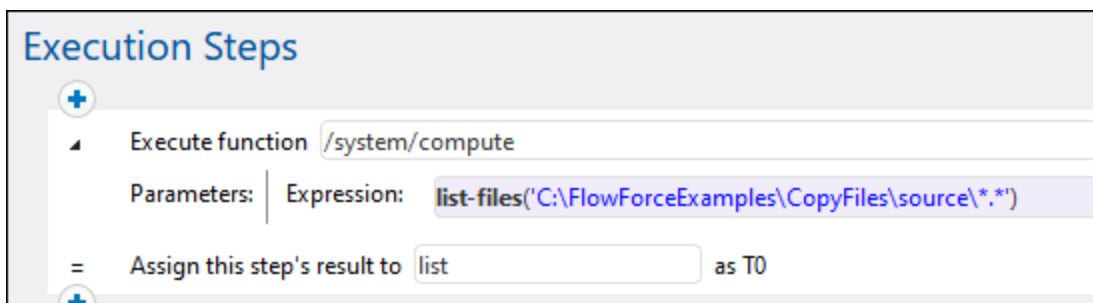
In order to iterate over items in a list, FlowForce Server provides a "for-each" execution step. Such a step iterates over a sequence (list) of items up to and including the last item in the sequence. In this example, our sequence of items will be the list of files in the source directory. To create the required list, click **New Execution Step** and type **/system/compute** next to "Execute function". You can also select this path from the drop-down list, as illustrated below.



Next, enter the following expression in the Expression field:

```
list-files("C:\FlowForceExamples\CopyFiles\Source\*.**")
```

Next, enter a name for the list in the **Assign this step's result to** field (in this case, the name is `list`). This makes it possible to easily refer to the newly created list of files in a subsequent step. Your first execution step should now look as follows:




The expression above invokes the [list-files](#)²¹⁵ expression function. The function takes a path as argument (in this case, `C:\Source*.**`) and returns the list of files (or directories) at the given path. Notice that the path contains the wildcard `*.**` to select all the files in the directory. If necessary, you can adjust the wildcard to select only specific file extensions, for example `*.txt`. For more information about expressions in FlowForce, see [FlowForce Expressions](#)¹⁷¹.

You can now proceed to creating the actual "for-each" iteration step. Click **New For-Each step** and type `list` in the "in sequence" box. (This refers to the `list` created in the previous execution step.)



Execution Steps

- Execute function `/system/compute`
 - Parameters: Expression: `list-files('C:\FlowForceExamples\CopyFiles\source*.*)`
 - = Assign this step's result to `list` as T0
- For each `item` in sequence `list`


Tip: You could also copy the expression to the "in sequence" box of the "for-each" step and thus get rid of the first execution step altogether.


Next, click the  button and add a new execution step inside the "for-each" step. This step will invoke the [/system/filesystem/copy](#)²⁴¹ function for each item in the loop, as illustrated below.

Execution Steps

- Execute function `/system/compute`
 - Parameters: Expression: `list-files('C:\FlowForceExamples\CopyFiles\source*.*)`
 - = Assign this step's result to `list` as T0
- For each `item` in sequence `list`
 - Execute function `/system/filesystem/copy`
 - Parameters:
 - Source: `{item}`
 - Target: `C:\FlowForceExamples\CopyFiles\target`
 - Overwrite:
 - Abort on error: 
 - Working directory: 

As shown above, the `copy` function is called with the following arguments:

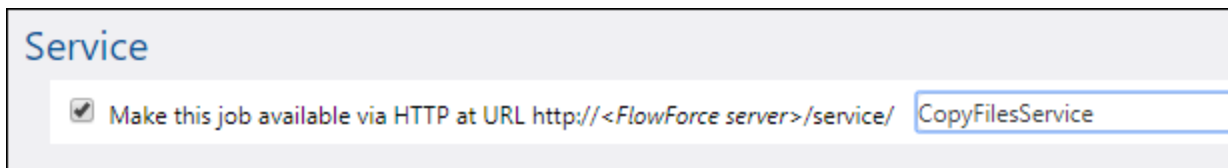
- The **Source** is the current item (file) in the loop. You can either type `{item}` in the Source box or click the  button and select `item`.
- The **Target** is the target path. In this example, the path is entered as is; however, you could also supply it as an argument to the job.

- The **Overwrite** option is enabled, meaning that if a file with the same name already exists in the source directory, it will be overwritten. To prevent this from happening, click the  button.

For the sake of simplicity, we will not set the other two arguments in this example. For further information, see the description of the [/system/filesystem/copy](#)²⁴¹ function.

The job created so far now includes all the required processing steps, but it has no trigger yet. To trigger the job at recurring time intervals, you could use a timer trigger, see [Timer Triggers](#)¹⁵². Or you can monitor the source directory for changes and trigger the job by means of a file system trigger, see [File System Triggers](#)¹⁵⁴. Finally, you can trigger the job on demand, as a Web service call.

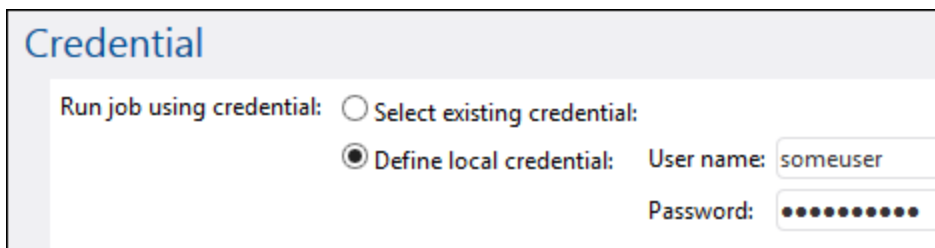
In this example, we will trigger the job on demand, by clicking a URL in the browser (in fact, this invokes the job as a Web service). To turn the job into a Web service, select the **Make this job available via HTTP...** check box and enter the name of the Web service.



Service

Make this job available via HTTP at URL `http://<FlowForce server>/service/`

Finally, the job needs your credentials to run. Therefore, enter your operating system username and password (not your FlowForce Server username and password) in the "Credential" section, as shown below. Alternatively, if you created standalone credentials previously, as described in [Defining Credentials](#)¹⁶³, you can select them using the **Select existing credential** option.




Credential

Run job using credential: Select existing credential:
 Define local credential: User name:
 Password:

Running the job

To test the job, do one of the following:

- Go to **Home**, and then click **Show all active triggers and services**. Next, click the job's URL displayed in the "Info" column.
- Enter <http://127.0.0.1:4646/service/CopyFilesService> in the browser's address bar. Note that this URL works only if the *FlowForce Server* service listens at the default host address and port name. If you have defined other host and port settings in the [Configuration page](#)⁴⁷, change the address accordingly.
- If you set the optional **Host name** field of FlowForce Server from the [Setup Page](#)⁴⁷, you can execute the web service call directly from the job configuration page, by clicking the  button adjacent to the **Make this job available via HTTP** check box. The button is not displayed otherwise.

If prompted for credentials when accessing the Web service, supply the same credentials you use to log on to

FlowForce Server.

Supplying your FlowForce Server user credentials for HTTP authentication is only for testing purposes. For production, it is recommended that you create a new FlowForce user, grant the **Service - Use** permission to this user on the container where the job is, and then access the Web service with the corresponding user account. To disable HTTP authentication and make the Web service public, grant the **Service - Use** permission to the user **Anonymous**, see [How Permissions Work](#)⁹².

Upon successful execution, the job will copy all the files from the source to the target directory. Otherwise, a "Service execution failed" error is displayed in the browser. If you see this error, check the log of the job for further information, see [Viewing the Job Log](#)¹⁰⁴. Possible causes may include incorrect credentials, incorrect file paths, insufficient permissions on the file system, and others. For example, the job fails if the **Overwrite** check box is not selected and the target directory already contains a file with the same name, as illustrated below:

Date	Message
2020-09-17 12:26:53	Starting instance 8.
2020-09-17 12:26:53	Starting job execution: job /public/Examples/CopyFiles in queue /public/Examples/CopyFiles
2020-09-17 12:26:53	Running instance 8 locally.
2020-09-17 12:26:53	Job /public/Examples/CopyFiles
2020-09-17 12:26:53	System function /system/compute
2020-09-17 12:26:53	For each item in list
2020-09-17 12:26:53	Iteration 1
2020-09-17 12:26:53	System function /system/filesystem/copy
2020-09-17 12:26:53	Executing FlowForce.copy with parameters: Source: "C:\FlowForceExamples\CopyFiles\source\invoices.txt",
2020-09-17 12:26:53	Step FlowForce.copy failed: Failed copying the file: The file exists.
2020-09-17 12:26:53	Job execution failed: job /public/Examples/CopyFiles in queue /public/Examples/CopyFiles

5.4 Create a Job from a MapForce Mapping

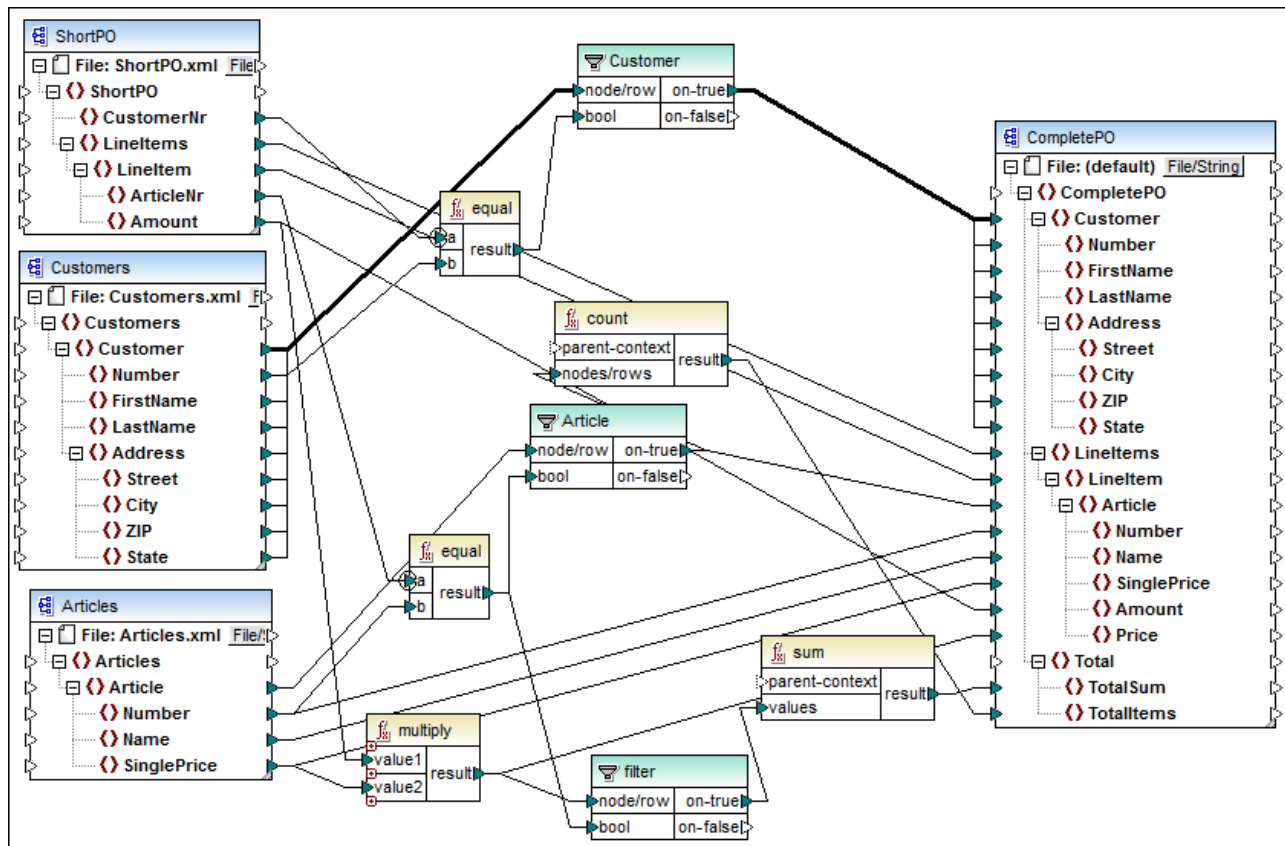
This example shows you how to create a FlowForce Server job from a MapForce mapping. First, you will deploy a demo mapping file from MapForce to FlowForce Server. Once the mapping is deployed to FlowForce Server, you will create a server job from it. The job will be configured to run daily at a specific time.

Prerequisites

- Required licenses: MapForce Enterprise or Professional edition, MapForce Server or MapForce Server Advanced Edition, FlowForce Server
- The *FlowForce Web Server* and *FlowForce Server* services must be listening at the configured [network address and port](#)⁴⁷
- You have a FlowForce Server user account with permissions to one of the containers (by default, the **/public** container used in this example is accessible to any authenticated user).
- The mapping job created in this example generates an XML file every time when it runs. Therefore, on the operating system where FlowForce Server runs, you must have rights to create files in some directory (this example uses the **C:\FlowForceExamples\Mapping** directory).

Demo files used

The mapping file used in this example is called **CompletePO.mfd**, and it is available at the following path on the computer where MapForce is installed: **<Documents>\Altova\MapForce2024\MapForceExamples**. Note that the "MapForceExamples" directory is created when you run MapForce for the first time (but not earlier).



CompletePO.mfd

The demo mapping illustrated above takes three XML files as input and produces a single XML file as output. In this example, the input XML files will be included automatically in the package deployed to FlowForce Server. Other mappings may require extra preparation steps before you can deploy them, as described in [Deploying Mappings to FlowForce Server](#)⁴¹².

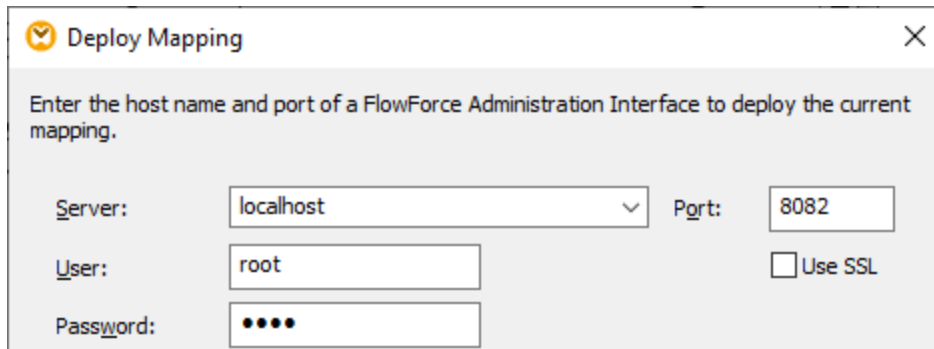
Creating the job

Deploying a mapping means that MapForce organizes the resources used by the mapping into a single package and sends it through HTTP (or HTTPS, if configured) to FlowForce Server.

To deploy the mapping to FlowForce Server:

1. Open the **CompletePO.mfd** file in MapForce.
2. If you haven't done so already, set the transformation language of the mapping to "Built-in".
3. On the **File** menu, click **Deploy to FlowForce Server**.
4. In the **Server** and **Port** text boxes, enter the server name and port of the Web administration interface (for example, 127.0.0.1 and 8082, if the *FlowForce Web Server* service is listening on the same machine at the default port). Change these values if you have configured a different address and port, see [Defining the Network Settings](#)⁴⁷.
5. In the **User** and **Password** text boxes, enter your FlowForce Server user name and password.
6. Select either **Directly** from the **Login** drop-down list, or leave the **<Default>** option as is.

If Directory Service integration is enabled, enter your domain user name and password, and then select your domain name from the **Login** drop-down list. For more information, see [Changing the Directory Service Settings](#) ¹¹⁹



Deploy Mapping

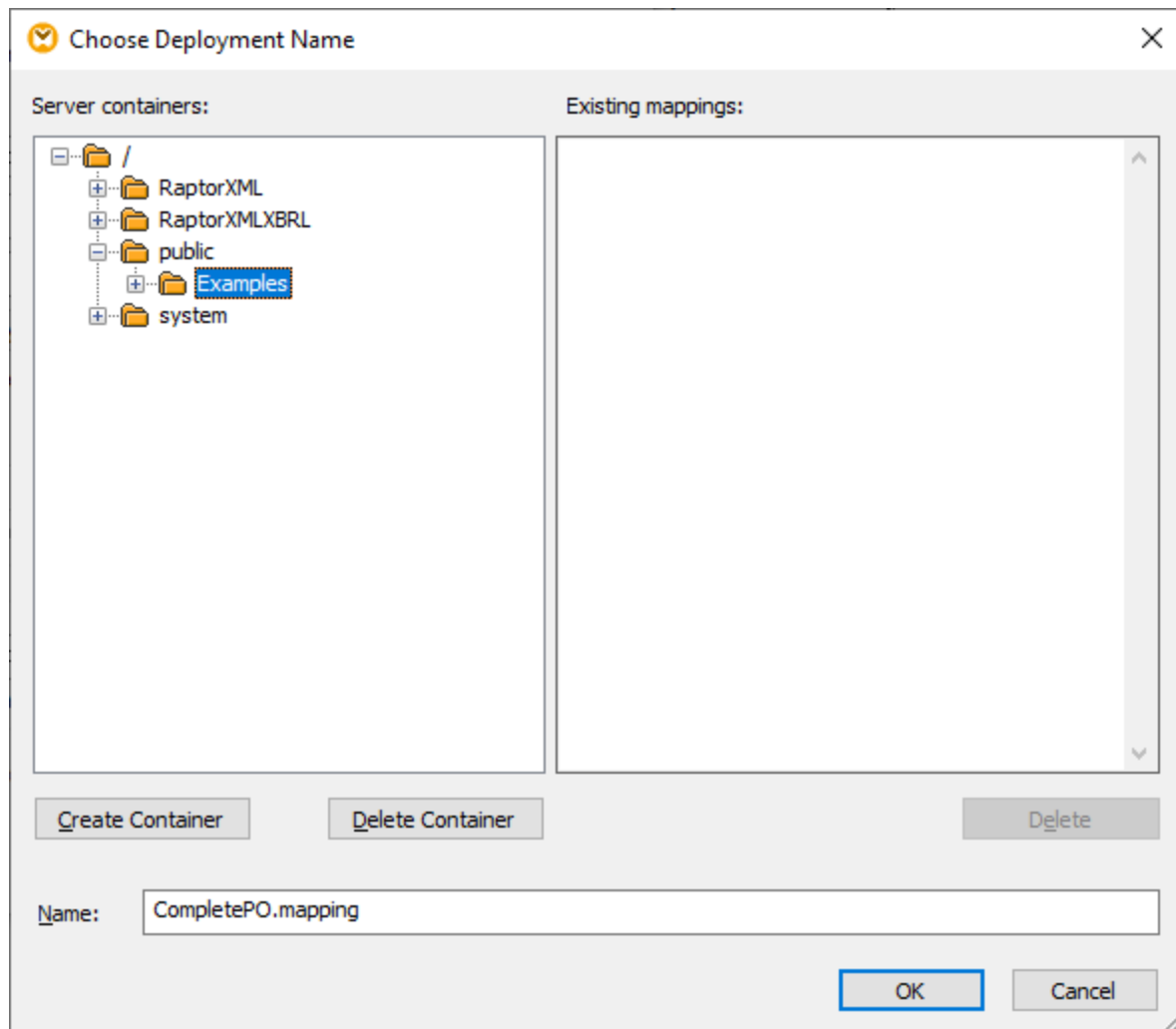
Enter the host name and port of a FlowForce Administration Interface to deploy the current mapping.

Server: localhost Port: 8082

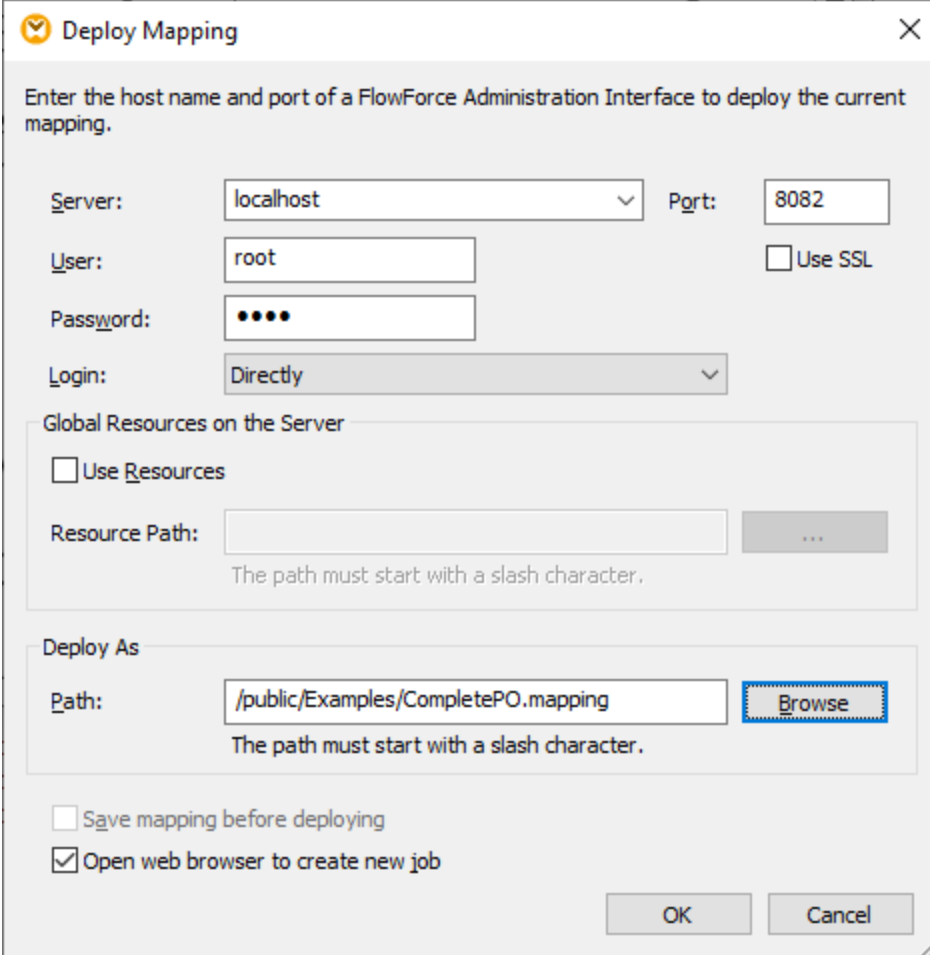
User: root Use SSL

Password: ●●●●●

7. For consistency with other examples, we will be deploying the mapping to the **/public/Examples** container. Click **Browse** and change the container path to **/public/Examples**. The **/public/Examples** container must already exist if you followed the previous examples; otherwise, you can create it by clicking **Create Container** in the dialog box below:



8. Select the **Open web browser to create new job** check box.



The image shows a 'Deploy Mapping' dialog box with the following fields and options:

- Server:** localhost (dropdown)
- Port:** 8082 (text box)
- User:** root (text box)
- Use SSL:**
- Password:** masked with four dots (text box)
- Login:** Directly (dropdown)
- Global Resources on the Server:**
 - Use Resources
 - Resource Path:** (text box) with a browse button (...)
 - The path must start with a slash character.
- Deploy As:**
 - Path:** /public/Examples/CompletePO.mapping (text box) with a browse button
 - The path must start with a slash character.
- Save mapping before deploying
- Open web browser to create new job

Buttons: OK, Cancel

9. Click **OK** to deploy the mapping.

When deployment finishes, the FlowForce Server Administration Interface opens in your web browser, and a partially filled in job page is displayed. The mapping function itself is saved at the container path specified earlier. This concludes the deployment part.

Creating the job

After you have deployed the mapping file to FlowForce Server as described above, the browser displays a partially filled job page. The first execution step is created automatically with some pre-filled parameters.

Create job in /public

Job name:

Job description:

Job Input Parameters

+

Execution Steps

+

Execute function

Parameters:	Customers:	(input)		<input style="border: 1px solid #ccc; border-radius: 50%;" type="button" value="+"/>
	Articles:	(input)		<input style="border: 1px solid #ccc; border-radius: 50%;" type="button" value="+"/>
	ShortPO:	(input)		<input style="border: 1px solid #ccc; border-radius: 50%;" type="button" value="+"/>
	CompletePO:	(output)		<input style="border: 1px solid #ccc; border-radius: 50%;" type="button" value="+"/>
	Working-directory:			<input style="border: 1px solid #ccc; border-radius: 50%;" type="button" value="+"/>

= Assign this step's result to as CompletePO

You can also create the job by opening the function's page (</public/Examples/CompletePO.mapping>), and then clicking **Create job**.

To configure the job:

1. Change the default job name from "CompletePO.job" to something more descriptive, for example, "GeneratePurchaseOrder". This is an optional step, but it may be necessary if the name is already used by some other job in the same container.
2. Fill in the first execution step created by default as follows:

Execute function	This field points to the mapping function deployed earlier; leave it as is.
Parameters	<p>The Customers, Articles, and ShortPO fields contain the respective XML files pre-packaged into the job.</p> <p>The CompletePO field specifies the path of the output file. By default, it is CompletePO.xml. The path is relative to the working directory, as further described below.</p>

	<p>In this example, you can leave all the input and output options as is. For information about changing input and output instances, see Running Mappings and Transformations as Jobs⁴¹⁷.</p> <p>In the Working-directory box, enter the path to the job's working directory. This example uses C:\FlowForceExamplesMapping as working directory.</p> <p>A working directory is a parameter required by execution steps if the job needs a location to unpack any input files or save output files. FlowForce Server also uses the working directory to resolve any relative paths that occur during step execution. When asked to provide a working directory, you should supply a valid path on the operating system where FlowForce Server runs. If you do not supply a working directory when creating the step, FlowForce Server uses a temporary directory.</p>
Assign this step's result to	This field gives a name to the mapping result. In this example, you can leave it empty.

- Under "Triggers", click **new Timer**.
- Next to "Run", set the timer to run **Daily** every **1** days. Next to "Start", select a date and time when the job must start, for example:

- Under "Credentials", select an existing credential record or specify a local credential. For details, see [Credentials](#)¹⁶².

- Click **Save**.

Running the job

At the time and date specified in the trigger, FlowForce Server executes the mapping job. If the job executes successfully, the file generated as a result (**CompletePO.xml**) becomes available in the **C:**

\FlowForceExamples\Mapping directory. To see whether the job executed successfully, refer to the [job log](#)¹⁰⁴.

5.5 Use a Job as Step of Another Job

This example shows you how to use a previously defined job as a step of another job. Since this example requires a previously created job, you should complete the [Creating a Job from a MapForce Mapping](#)³⁰⁸ example before completing this example.

As you may recall from the [Creating a Job from a MapForce Mapping](#)³⁰⁸ example, the **GeneratePurchaseOrder** job generates an XML file in a temporary folder every time when it runs. This example shows you how to do the following:

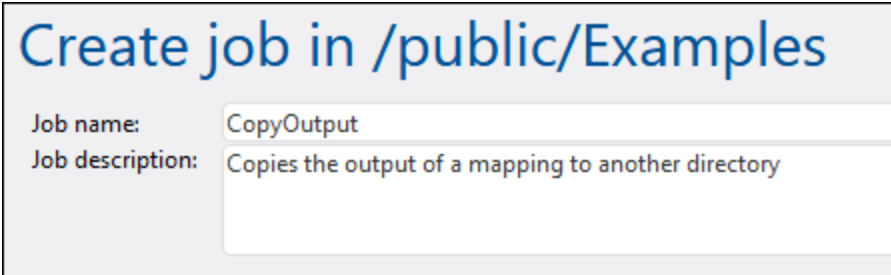
1. Create a job that copies the file generated by the mapping to another directory. We will call this job **CopyOutput**.
2. Modify the **GeneratePurchaseOrder** job to include the **CopyOutput** job as an additional execution step.

Prerequisites

- Required licenses: MapForce Enterprise or Professional edition, MapForce Server or MapForce Server Advanced Edition, and FlowForce Server
- The *FlowForce Web Server* and *FlowForce Server* services must be listening at the configured [network address and port](#)⁴⁷
- You have a FlowForce Server user account with permissions to one of the containers (by default, the **/public** container is accessible to any authenticated user).
- The mapping job created in this example copies files from one directory to another. Therefore, on the operating system where FlowForce Server runs, ensure that both directories exist and that you have rights to create files in both directories. This example uses the **C:\FlowForceExamples\Mapping** and **C:\FlowForceExamplesArchive** directories.
- Complete the steps described in the [Creating a Job from a MapForce Mapping](#)³⁰⁸ example.

Creating the job

1. Click **Configuration**, and then navigate to the **/public/Examples** container. The **public/Examples** container should already exist if you followed the previous examples; otherwise, create it using the **Create | Create Container** command.
2. Click **Create**, and then select **Create Job**.
3. Enter the name of the job (in this example, "CopyOutput").



Create job in /public/Examples

Job name:	CopyOutput
Job description:	Copies the output of a mapping to another directory

4. Under "Execution steps", add the first execution step, with the following settings:

Execution Steps

+

Execute function
/system/filesystem/copy

Parameters:	Source:	CompletePO.xml
	Target:	C:\FlowForceExamples\Archive\PurchaseOrder.xml
	Overwrite:	<input checked="" type="checkbox"/>
	Abort on error:	+
	Working directory:	C:\FlowForceExamples\Mapping

Execute function	Browse for the <code>/system/filesystem/copy</code> function.
Source	<p>CompletePO.xml</p> <p>We used a relative path because the Working Directory parameter is set, see below.</p>
Target	<p>This must be an existing file or directory path on the operating system where FlowForce Server runs. In this example, we would like to rename the file when it is copied, so we'll add the file name to the path, as follows:</p> <p>C:\FlowForceExamples\Archive\PurchaseOrder.xml</p>
Overwrite	Select this check box. This instructs FlowForce Server to overwrite any file with the same name found at the destination path.
Abort on error	<p>Leave this parameter as is.</p> <p>Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code>, job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code>, FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code>.</p>
Working directory	<p>FlowForce will look for all relative file paths in this directory. Set it to:</p> <p>C:\FlowForceExamples\Mapping</p>

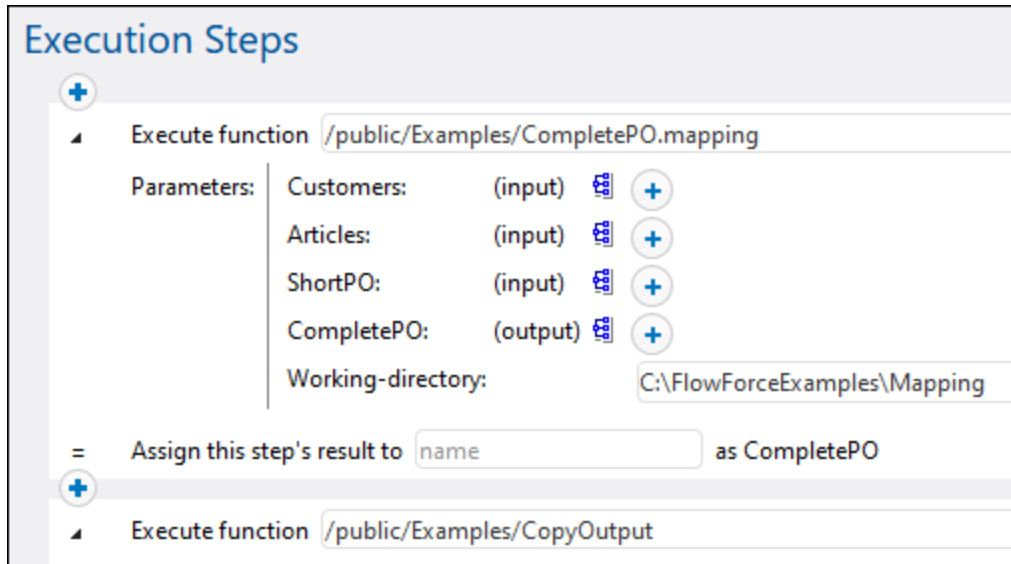
5. Under "Credentials", select an existing credential record or specify a local credential. For more information, see [Credentials](#)¹⁶².
6. Click **Save**.

As you may have noticed, the job we just created does not have any trigger. We did not define any trigger because we will call this job from another job.

Adding the "CopyOutput" job as a step of another job

1. Open the **GeneratePurchaseOrder** from the **/public/Examples** container.

2. Under "Execution Steps", click **new Execution step** to add a new step after the existing one.
3. Next to "Execute function", browse for the **CopyOutput** job created earlier. The execution steps should now look as follows:



4. Update the time trigger, and then click **Save**.
5. At the time entered in the trigger, FlowForce Server executes the job and copies the **CompletePO.xml** file to the specified directory and renames it to **PurchaseOrder.xml**. To see whether the job executed successfully, refer to the [job log](#)¹⁰⁴.

5.6 Create a Directory Polling Job

This example shows you how to monitor a directory for changes with the help of a file system trigger created in FlowForce Server (see also [File System Triggers](#)¹⁵⁴). Whenever a new XML file is added to the directory, FlowForce Server executes a mapping job that takes the XML file as input parameter. The output of the mapping job is then moved to an archive directory.

Prerequisites

- Required licenses: MapForce Enterprise or Professional edition, MapForce Server or MapForce Server Advanced Edition, and FlowForce Server
- The *FlowForce Web Server* and *FlowForce Server* services must be listening at the configured [network address and port](#)⁴⁷
- You have a FlowForce Server user account with permissions to one of the containers (by default, the **/public** container is accessible to any authenticated user).
- The mapping job created in this example copies files from one directory to another. Therefore, on the operating system where FlowForce Server runs, ensure that both directories exist and that you have rights to create files in both directories. This example uses the **C:\FlowForceExamples\DirPolling** and **C:\FlowForceExamples\Archive** directories.

Demo files used

- **ShortApplicationInfo.mfd** — the MapForce mapping from which the FlowForce Server job will be created
- **ApplicationsPage.xml** — the XML instance file to be supplied as input to the mapping.

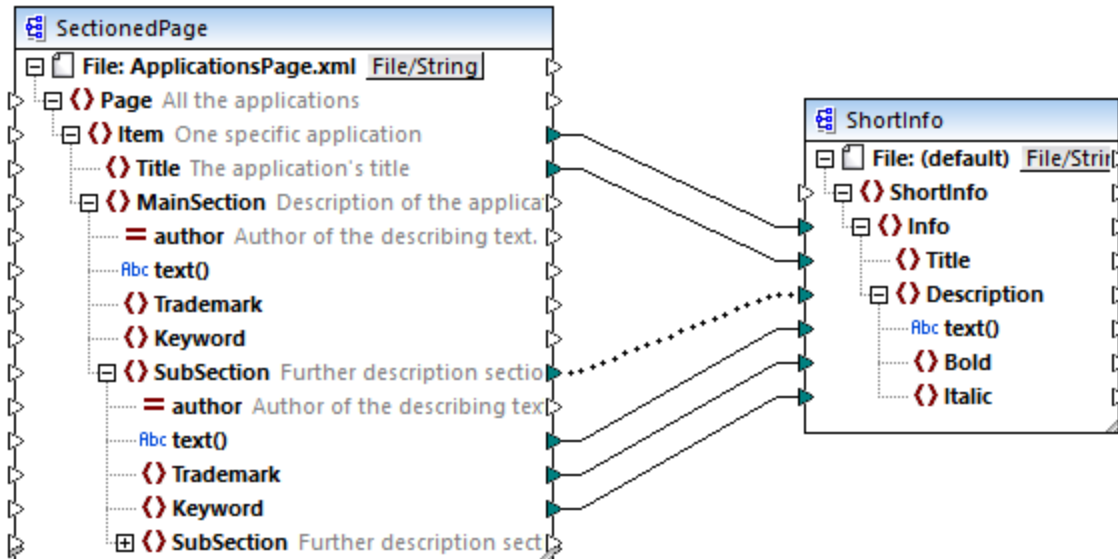
Both files are available at the following path on the machine where MapForce is installed:

<Documents>\Altova\MapForce2024\MapForceExamples\.

What the mapping does

The MapForce mapping used in this example (**ShortApplicationInfo.mfd**) is illustrated below. From a FlowForce Server perspective, the important thing is that the mapping takes an XML file as input, and produces another XML file as output.

Essentially, this mapping converts an XML file (**ApplicationsPage.xml**) to a different schema and saves it as **ShortInfo.xml**. The mapping is relatively easy to understand by looking at the topmost connection: for each **Item** found in the source, it creates an **Info** item in the target. The other connections are used to copy values from the respective child items. Of particular interest is the dotted connection; in MapForce, this connection is called "Source-driven (Mixed Content)" and it is used because **SubSection** contains mixed content.



ShortApplicationInfo.mfd

Notice the names of the source and target XML schemas are **SectionedPage** and **ShortInfo**, respectively. As you will see further below, the FlowForce job will have an input and output parameter with the same name, after the mapping is deployed to FlowForce Server.

Deploying the mapping to FlowForce Server

The mapping **ShortApplicationInfo.mfd** does not need any special preparation before it is deployed to FlowForce Server. Since both the source and target components are XML files, they will be included automatically in the package deployed to FlowForce Server.

To deploy the mapping to FlowForce, open it in MapForce and run the menu command **File | Deploy to FlowForce Server**.

Deploy Mapping

Enter the host name and port of a FlowForce Administration Interface to deploy the current mapping.

Server: localhost Port: 8082

User: root Use SSL

Password: ●●●●

Login: <Default>

Global Resources on the Server

Use Resources

Resource Path: ...
The path must start with a slash character.

Deploy As

Path: /public/Examples/ShortApplicationInfo.mapping
The path must start with a slash character.

Save mapping before deploying

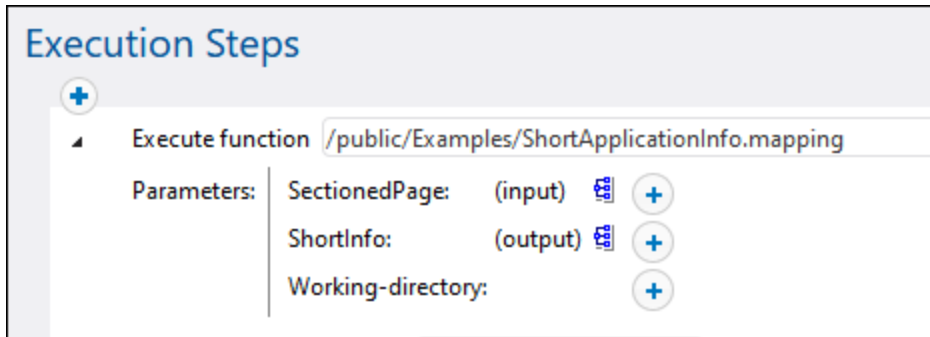
Open web browser to create new job

OK Cancel

If FlowForce Server runs on a different host and port, change the connection details above accordingly, see [Defining the Network Settings](#)⁴⁷. Also note that the path of the mapping is **/public/Examples/ShortApplicationInfo.mapping**; this is consistent with previous examples.

Creating the directory polling job

After the mapping is deployed to FlowForce Server, the browser opens and loads the job creation page. As illustrated below, the first execution step is created automatically and it calls the mapping function deployed previously. Notice that the input parameter has the same name as the source MapForce component (**SectionedPage**), while the output parameter has the same name as the target component (**ShortInfo**).

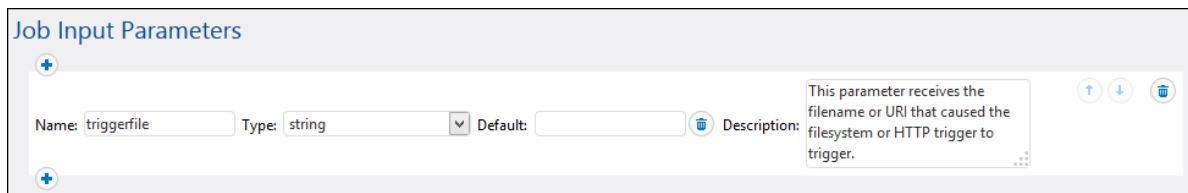


Configure the job as follows:

1. In the **Working-directory** box, enter the path to the working directory. This example uses **C:\FlowForceExamples\DirPolling** as working directory.

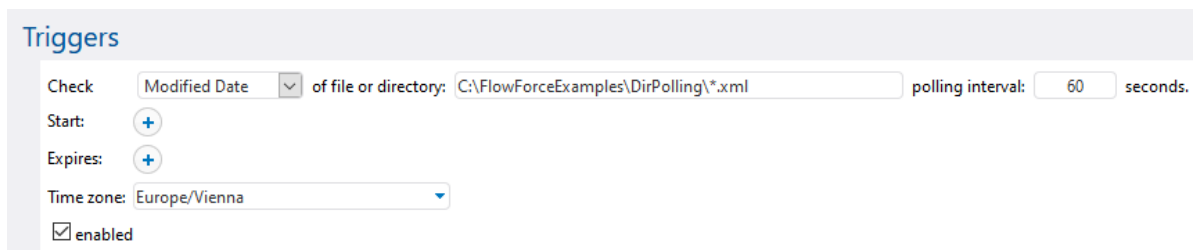
A working directory is a parameter required by execution steps if the job needs a location to unpack any input files or save output files. FlowForce Server also uses the working directory to resolve any relative paths that occur during step execution. When asked to provide a working directory, you should supply a valid path on the operating system where FlowForce Server runs. If you do not supply a working directory when creating the step, FlowForce Server uses a temporary directory.

2. Under "Triggers", click **new Filesystem trigger**. Notice that FlowForce Server automatically adds a new **triggerfile** parameter under "Input Parameters". You will need to refer to this parameter in a subsequent step.



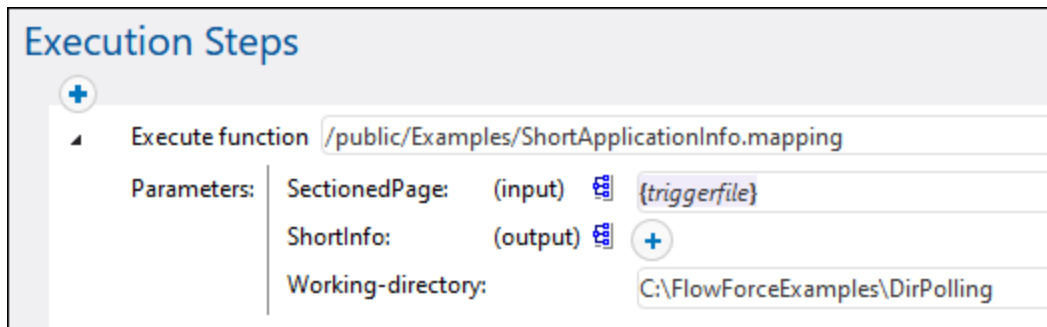
3. Set the following trigger values:

- Check: Modified Date
- File or directory: **C:\FlowForceExamples\DirPolling*.xml**
- Polling interval: 60 seconds



4. Under Execution Steps, supply the **triggerfile** parameter as input value to the **SectionedPage** parameter. To do this, click the **Set to** button next to the SectionedPage parameter, and then

select **triggerfile**. As a result, the value of the **SectionedPage** parameter changes to **{triggerfile}**. The curly braces denote a FlowForce expression and should not be removed.



With the configuration done so far, the trigger will fire whenever **ApplicationsPage.xml** is copied into the working directory. However, since the trigger uses a wildcard (*.xml), it would be fired also when any other XML file changes inside the directory, including the mapping output itself (**ShortInfo.xml**). This is not the intended behavior and could cause errors; therefore, let's add a second step that will move the generated output file to a new directory. Alternatively, you could rename the trigger to **C:**

\FlowForceExamples\DirPolling\ApplicationsPage.xml (in this case, a second step is no longer necessary).

To add the step which moves the output to a new directory, do the following:

1. Add a new execution step, immediately after the previous one.
2. Configure the step as follows (note that the source and destination fields are case-sensitive):

Execute function	Browse for the <code>/system/filesystem/move</code> function.
Source	ShortInfo.xml We used a relative path because the Working Directory parameter is set, see below.
Destination	This must be an existing file or directory path on the operating system where FlowForce Server runs. Set it to: C:\FlowForceExamples\Archive
Overwrite target	Select this check box. This instructs FlowForce Server to overwrite any file with the same name found at the destination path.
Abort on error	Leave this parameter as is. Optional parameter. This parameter determines the outcome of a job in which an error has occurred. If the <i>Abort on error</i> parameter is <code>true</code> , job execution will be terminated. If the <i>Abort on error</i> parameter is <code>false</code> , FlowForce Server will ignore errors and continue job execution. The default value is <code>true</code> .
Working directory	FlowForce will look for all relative file paths in this directory. Set it to: C:\FlowForceExamples\DirPolling

Execute function /system/filesystem/move		
Parameters:	Source:	ShortInfo.xml
	Destination:	C:\FlowForceExamples\Archive
	Overwrite target:	<input checked="" type="checkbox"/>
	Abort on error:	<input type="checkbox"/>
	Working directory:	C:\FlowForceExamples\DirPolling

Finally, add your operating system credentials with which the job will be executed:

1. Under "Credentials", select an existing credential record or specify a local credential. For more information, see [Credentials](#)¹⁶².
2. Click **Save**.

Running the job

You can now test the job by copying the file **ApplicationsPage.xml** to the working directory. When you do this, FlowForce Server executes the mapping job and copies the resulting output file to the archive directory.

To see whether the job executed successfully, refer to the [job log](#)¹⁰⁴.

5.7 Add Error Handling to a Job

This example illustrates how to add error handling to a job. The job used in this example lists the contents of a directory and will be invoked from the browser, as a Web service. You will learn how to configure FlowForce Server to handle the job outcome as follows:

- If the job execution is successful, display the job's output in the browser
- If the job fails to execute due to any reason, send an email notification to a named recipient.
- Whenever the job execution finishes, regardless of the execution status, log the job internal ID to a file on the local system.

In FlowForce Server terms, in this example you create a [protected block](#)¹³⁶ with two error handling conditions: "On Error" and "Always" (each will handle one of the scenarios mentioned above).



Prerequisites

- Required licenses: FlowForce Server
- The *FlowForce Web Server* and *FlowForce Server* services must be listening at the configured [network address and port](#)⁴⁷
- The FlowForce Server mail settings must be configured, see [Setting the Mail Parameters](#)¹¹⁸
- You need a FlowForce Server user account with permissions to one of the containers (by default, the **/public** container is accessible to any authenticated user).
- The job created in this example writes output to the disk. Therefore, on the operating system where FlowForce Server is installed, you need read and write access to some directory. This example uses **C:\FlowForceExamples\ErrorHandling**.

Tips


- Although this example uses Windows paths and commands, you can still test it on other operating systems, if you adapt the paths and the commands accordingly.

Creating the job

1. On the computer where FlowForce Server runs, create a directory that will be the job's working directory. This example uses **C:\FlowForceExamples\ErrorHandling**.
2. Log in to the FlowForce Server Web administration interface and go to the **/public/Examples** container. The **public/Examples** container should already exist if you followed the previous examples; otherwise, create it using the **Create | Create Container** command.
3. Click **Create Job** and enter a name for the job you are creating, for example "ListDirectory". The job's description is optional.
4. Under "Job Input Parameters", click the  button, and add a parameter of type "string". At job runtime, the parameter will provide the path of the directory to list. In this example, the name of the parameter is "inputDir"; it will be used in subsequent steps.
5. Under "Execution Steps", click **new error/success handling step**.
6. Under "Execute with error/success handling", click the  button, and choose to add a new execution step, with the following settings:


Execute function	Browse for the <code>/system/shell/commandline</code> function.
------------------	---


Command	<p>Enter the following shell command:</p> <pre>dir {inputDir}</pre> <p>Where <code>inputDir</code> is the name of the parameter created previously. The name is enclosed within curly braces because, at job runtime, its content will be replaced dynamically with the parameter value. For more information, see Embedding Expressions in String Fields¹⁷⁵.</p>
Abort on error	<p>Leave this option as is. For more information, see the description of the /system/shell/commandline²⁸⁵ function.</p>
Working directory	<p>Enter the path to the working directory created previously, for example C:\FlowForceExamples>ErrorHandling</p>

7. Under the "On error" condition, click the  button and choose to add a new execution step, with the following settings:

Execute function	Browse for the /system/mail/send ²⁷⁹ function.
From	Enter the email address of the sender, for example <code>flowforce@localhost</code> . Leave this field empty if you have configured the mail settings from the administration page.
To	Enter your email address.
Subject	<p>Enter the subject of the notification email as follows:</p> <pre>Job {instance-id()} has failed</pre> <p>The part between curly braces is a FlowForce expression which calls the instance-id²³² function to get the unique ID of the current (failed) job instance.</p>
Message body	<p>Type the following:</p> <pre>Exit Code: {string(exitcode(failed-step()))} Standard Error: {content(stderr(failed-step()))} Error message: {error-message(failed-step())}</pre> <p>The parts between curly braces are two FlowForce expressions. These expressions get the erroneous output and convert it to a string that will be the body of the email:</p> <ul style="list-style-type: none"> • The failed-step²²⁹ function returns the <code>result</code> of the failing step. This is an abstract FlowForce type that, in order to become more useful, must be supplied as argument to the <code>exitcode</code>, <code>stderr</code>, or <code>error-message</code> functions, see below. • The exitcode²⁰⁴ function gets the actual exit code of the error from the <code>result</code>, as a number, assuming that there is an exit code.

	<ul style="list-style-type: none"> • The stderr²⁰³ function gets the standard error output of the error from the <code>result</code>, as a stream, assuming that there is standard error output. • The error-message²⁰⁵ function gets the text of the FlowForce error message as it appears in the log. It may also return an empty string if there is no error or if it is not technically possible to retrieve the error text. • The string²²⁰ function converts the numeric exit code to a string (this must be done because the body of the email is of <code>string</code> type). • The content¹⁸³ function converts the error output from a stream to a string. <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p>The <code>exitcode</code> and <code>stderr</code> functions return a value only if execution produces an exit code and error output, respectively. This is typically the case for errors such as the ones produced by the command line. The <code>error-message</code> function is just for informational purpose and is not guaranteed to return the text of the error for every possible job configuration and error condition encountered.</p> </div>
--	--

8. Click **new error/success handler**, and then select **Always**.
9. Under the "Always" condition, click the  button and choose to add a new execution step, with the following settings:

Execute function	Browse for the  /system/shell/commandline ²⁸⁵ function.
Command	<p>Enter the following shell command:</p> <pre style="background-color: #f0f0f0; padding: 5px;">echo {instance-id()} >> JobLog.txt</pre> <p>On Windows, this command writes the job ID to a file called JobLog.txt. If the file contains data, the new text will be added after the existing data.</p>
Working directory	<p>Enter the path of the directory created previously (C:\FlowForceExamples\ErrorHandling).</p> <p>This directory will be used to resolve the path to the JobLog.txt file.</p>

At this stage, the job should look as follows (provided you did not use different paths or shell commands).

Job Input Parameters

Name: Type: Default: Description:

Execution Steps

Execute with error/success handling - on error, retry 0 times

Execute function /system/shell/commandline

Parameters: Command: Abort on error: Working directory:

= Assign this step's result to as result

On error

Execute function /system/mail/send

Parameters: From: To: Subject: Message body:

```
1 Exit Code: {string(exitcode(failed-step))}
2 Standard Error: {content(stderr(failed-step))}
3 Error message: {error-message(failed-step)}
```

 Attachment: Abort on error:

- To turn the job into a Web service, select the **Make this job available via HTTP...** check box and enter the name of the Web service, for example:

Service


Make this job available via HTTP at URL

Take notice of the service name; you will need it to call the Web service.

- Under "Credentials", select an existing credential record or specify a local [credential](#)¹⁶².
- Click **Save**.

Running the job

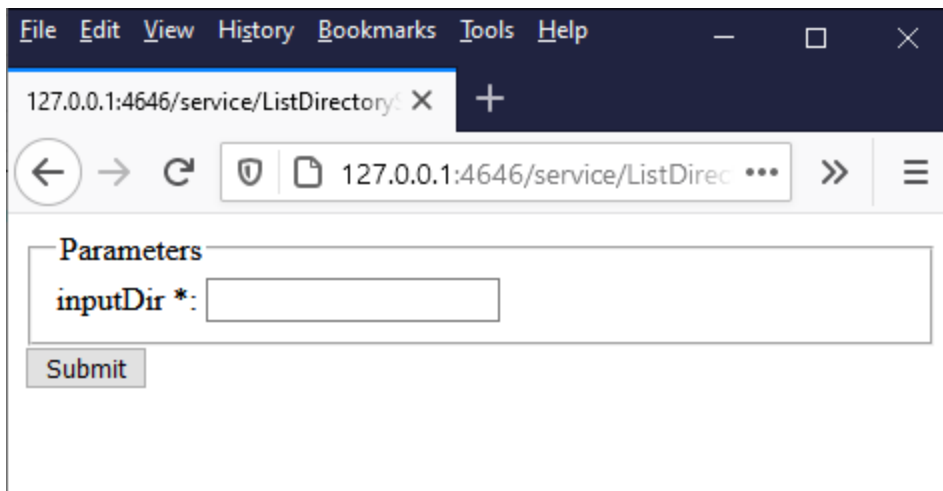
At this stage, you have completed the job configuration. Because this job is exposed as a Web service, you can run it in any of the following ways:

- Go to **Home**, and then click **Show all active triggers and services**. Next, click the job's URL displayed in the "Info" column.
- Enter <http://127.0.0.1:4646/service/ListDirectoryService> in the browser's address bar. Note that this URL works only if the *FlowForce Server* service listens at the default host address and port name. If you have defined other host and port settings in the [Configuration page](#)⁴⁷, change the address accordingly.
- If you set the optional **Host name** field of FlowForce Server from the [Setup Page](#)⁴⁷, you can execute the web service call directly from the job configuration page, by clicking the  button adjacent to the **Make this job available via HTTP** check box. The button is not displayed otherwise.

If prompted for credentials when accessing the Web service, supply the same credentials you use to log on to FlowForce Server.

Supplying your FlowForce Server user credentials for HTTP authentication is only for testing purposes. For production, it is recommended that you create a new FlowForce user, grant the **Service - Use** permission to this user on the container where the job is, and then access the Web service with the corresponding user account. To disable HTTP authentication and make the Web service public, grant the **Service - Use** permission to the user **Anonymous**, see [How Permissions Work](#)⁹².

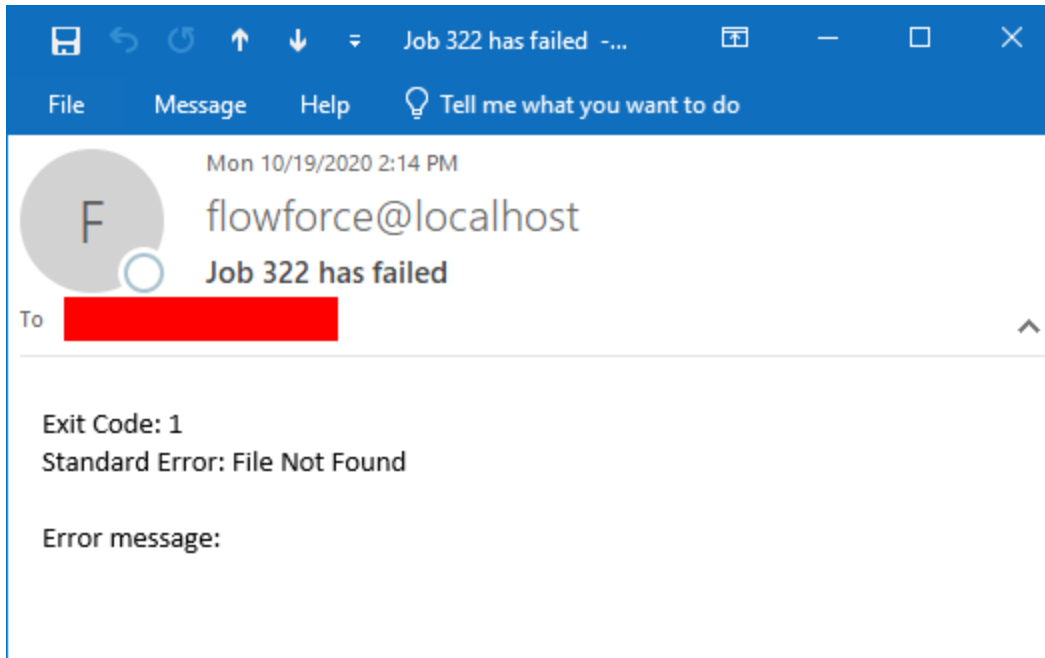
Since this job takes parameters, you will be prompted to supply a parameter value when you access the Web service from the browser.



If you enter a valid directory path like **C:**, for example, the job is executed, and the outcome is displayed in the browser.

Also, each time when you run the job, the ID of the job instance is appended to the contents of the **JobLog.txt** file, according to the "Always" condition configured previously.

To test the "On Error" condition, change the "inputDir" parameter to some deliberately incorrect value (for example, a path that does not exist). In this case, the browser will display an error and FlowForce Server will send an email to the address specified in the recipient field of the "On Error" handler. For example, the e-mail could look as follows:



As stated previously, the error functions used in this example are not guaranteed to return a value for each and every possible job configuration. Therefore, the level of detail provided by the e-mail depends on your job configuration and the kind of error encountered, and it should not be expected that the **Exit Code**, **Standard Error**, and **Error message** e-mail fields always contain text. The most authoritative reference for the cause of the error is the FlowForce Server [log](#)¹⁰⁴.

5.8 Expose a Job as a Web Service

This example illustrates how to create a FlowForce Server job exposed as a Web service. The job executes a mapping designed with Altova MapForce. The mapping queries data from a SQLite database and retrieves only records that match a value supplied as parameter when the Web service is called. You will learn how to deploy the existing mapping from MapForce to FlowForce Server and turn it into a Web service. After completing this example, you will be able to invoke the Web service from a browser.

Prerequisites

- Required licenses: MapForce Enterprise or Professional edition, MapForce Server or MapForce Server Advanced Edition, FlowForce Server
- The *FlowForce Web Server* and *FlowForce Server* services must be listening at the configured [network address and port](#) ⁴⁷
- You have a FlowForce Server user account with permissions to one of the containers (by default, the **/public** container is accessible to any authenticated user)
- The mapping job created in this example writes an XML file to a local directory. Therefore, on the operating system where FlowForce Server runs, a writeable directory must exist where the job output will be created. This example uses **C:\FlowForceExamples\GetPersonRecords**.
- The job used in this example reads data from a SQLite database and does not require installing any database drivers. However, if you would like to use a different database, then the database drivers must be installed not only on the computer where the mapping is designed but also on the server where the job runs. For example, in case of Microsoft Access databases, the Microsoft Access Runtime (<https://www.microsoft.com/en-us/download/details.aspx?id=50040>) must be installed on the machine where FlowForce Server runs.

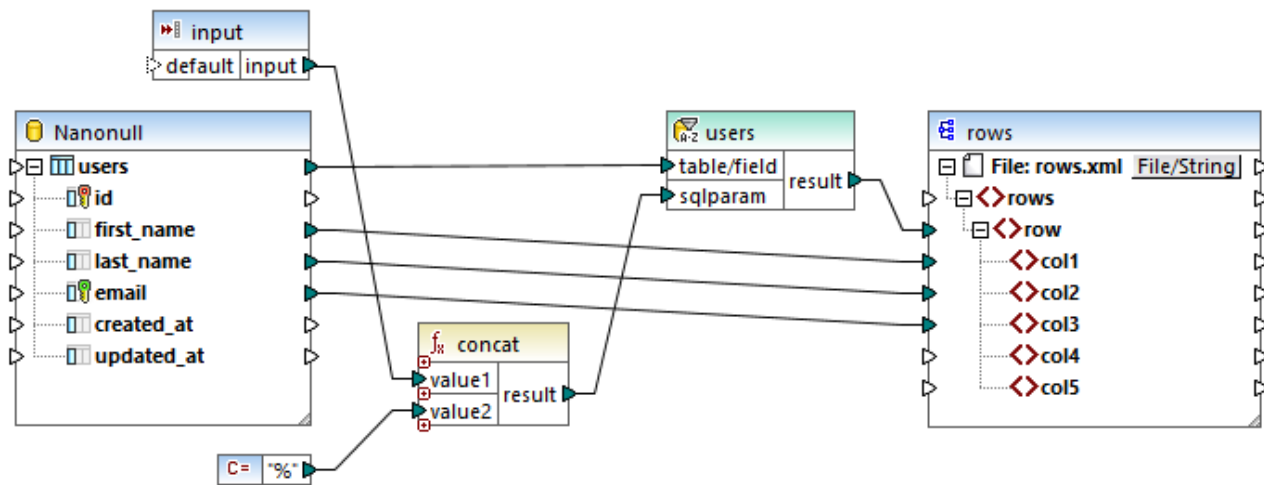
Demo files used

This example makes use of the following files, available at the following path on the computer where MapForce is installed: **..\Documents\Altova\MapForce2024\MapForceExamples\Tutorial**.

- **FilterDatabaseRecords.mfd** (the MapForce mapping design file)
- **Nanonull.sqlite** (the SQLite database from which the mapping reads data).

What the mapping does

The mapping discussed in this example is called **FilterDatabaseRecords.mfd** and is available in the "Tutorial" folder of MapForce (**..\Documents\Altova\MapForce2024\MapForceExamples\Tutorial**).



As illustrated above, the source component is a database which stores user records. The target component is an XML file. The connection from **users** to **row** creates one row for each database record extracted from the source. The mapping also contains an input parameter to be supplied at runtime. Double-click the title bar of the input parameter to view its properties:

📄 Create Input
✕

Name:

Datatype: string ▾

Input is required

Design-time Execution

Specify value

Value:

OK
Cancel

The mapping also contains a SQL-WHERE component placed between the source and the target. The goal of the SQL-WHERE component is to pass on to the target component only those database records that match the condition `last_name LIKE :sqlparam`. Again, this is configured from the component properties:

SQL-WHERE/ORDER Properties

To create a parameter, enter a ':' followed by its name in the WHERE condition.
Sample: Name = :Name AND Age >= :ValidAge

SELECT (fields) FROM users WHERE

```
last_name LIKE :sqlparam
```

Parameters:

Parameter	Type
sqlparam	string

ORDER BY

```
last_name DESC
```

OK Cancel

On the mapping, the value of `:sqlparam` is obtained by concatenating the input parameter with the % character. Therefore, if the caller supplies the input parameter "m" at runtime, then the mapping will retrieve user records whose last name begins with "m".

For more information about designing mappings such as the one discussed in this example, refer to MapForce documentation (<https://www.altova.com/documentation>).

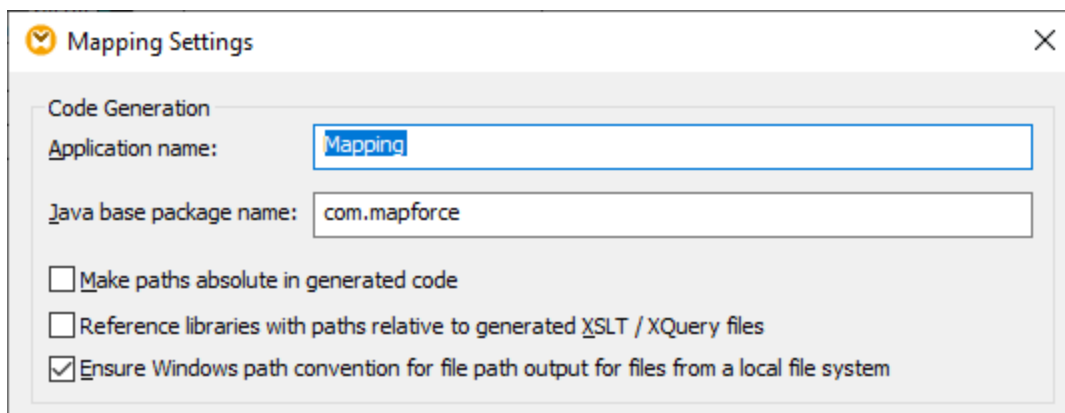
Preparing the mapping for deployment to FlowForce Server

In the instructions below, the term "source machine" refers to the computer where the MapForce is installed and the term "target machine" refers to the computer where FlowForce Server is installed (this may or may not be the same computer).

Before attempting to deploy the mapping to the target machine, do the following:

1. Make sure that the "FlowForce Web Server" service is configured to listen for client HTTP(S) requests, see [Defining the Network Settings](#)⁴⁷. For example, if FlowForce Server is installed on the same computer and is configured with the default settings, then you should be able to access it by typing **http://localhost:8082** in your browser. If FlowForce Server is running on a different computer, make sure that the incoming connections to the specified address and port are not blocked by the firewall.

2. Make sure that the "FlowForce Server" service is also configured to listen for client HTTP(S) requests. This service handles requests to jobs exposed as Web services, see also [How It Works](#)¹². Therefore, in order for the Web service to be accessible to HTTP clients outside of the local host, the "FlowForce Server" service must be configured to listen either on all interfaces, or on a specific address other than the local host. You can check whether this service is configured correctly by accessing the following URL from the browser: `http(s)://<host or IP address><port>/service/`. When prompted to enter a password, supply the password of your FlowForce Server user account. All jobs that are exposed as Web services (if any) should appear as links directly in the browser window.
3. Verify that the mapping is configured to use relative instead of absolute paths, as follows:
 - a) Open the **FilterDatabaseRecords.mfd** mapping in MapForce, right-click the mapping area, and select **Mapping Settings** from the context menu.
 - b) If applicable, clear the **Make paths absolute in generated code** check box.



Note: The check box **Ensure Windows path convention...** is not applicable in case of mappings designed in the BUILT-IN language, such as this one. It is relevant only when the mapping language is either XSLT or XQuery.

- c) Save the mapping.

File-based databases such as Microsoft Access or SQLite are not deployed to a target machine together with the mapping. Therefore, the SQLite database must be manually copied from the source machine to the target machine. Copy the **Nanonull.sqlite** database file from the directory ..

\Documents\Altova\MapForce2024\MapForceExamples\Tutorial on the source machine to some empty directory on the target machine. In this example, the target directory is **C:**

\FlowForceExamples\GetPersonRecords. Take notice of this path because it will be referenced later from the FlowForce job.

The mapping is now ready for deployment to FlowForce Server. For more information about deploying mappings which include database connections, see [Preparing Files for Server Execution](#)⁴⁰⁶.

Deploying the mapping

To deploy the mapping to FlowForce Server:

1. On the **File** menu, click **Deploy to FlowForce Server**. If you are deploying the mapping to FlowForce Server on a different machine, change the server address and port from "localhost:8082" to those configured from the FlowForce Server [network settings](#)⁴⁷.

Deploy Mapping

Enter the host name and port of a FlowForce Administration Interface to deploy the current mapping.

Server: localhost **Port:** 8082

User: root Use SSL

Password: ●●●●

Login: Directly

Global Resources on the Server

Use Resources

Resource Path: ...
The path must start with a slash character.

Deploy As

Path: /public/Examples/FilterDatabaseRecords.mapping
The path must start with a slash character.

Save mapping before deploying

Open web browser to create new job

2. For consistency with all other examples, we will choose to deploy the mapping to the **/public/Examples** container. Click **Browse** and change the container path to **/public/Examples**. The **/public/Examples** container must already exist if you followed the previous examples; otherwise, you can create it by clicking **Create Container**.
3. Select the **Open new browser to create new job** check box.
4. Click **OK**.

For reference to all deployment settings, see [Deploying Mappings to FlowForce Server](#)⁴¹².

Creating the FlowForce job

So far, you have deployed the mapping to FlowForce Server and have the job configuration page open in the browser (provided that you selected the check box **Open web browser to create new job** on the dialog box above). Otherwise, login to the FlowForce Server Web administration interface, open the previously deployed mapping function (it should be in the **/public/Examples** container), and then click **Create Job**.

To configure the job:

1. Under "Job Input Parameters", create a new input parameter of type **string**. This value will be provided by callers of the Web service when they invoke the job. Let's name it "LookupValue".

Create job in /public/Examples

Job name:

Job description:

Job Input Parameters

Name: Type: Default: Description:

2. Configure the execution step as follows:

- Set the value of the **input** parameter to the "LookupValue" input parameter created in previous step.
- Set the working directory to **C:\FlowForceExamples\GetPersonRecords**. Note that this directory must already exist on the file system, and it must already contain the source **Nanonull.sqlite** database if you followed the previous steps.

Execution Steps

Execute function

Parameters: input:

rows: (output)

Working-directory:

= Assign this step's result to as rows

3. To turn the job into a Web service, select the **Make this job available via HTTP...** check box and enter the name of the Web service, for example:

Service

Make this job available via HTTP at URL

Take notice of the service name; you will need it to call the Web service.


4. Under "Credentials", select an existing credential record or specify a local [credential](#)¹⁶².

Note: These are the credentials of your user account on the operating system and not the ones used to access the FlowForce Server Web administration interface. The user account must be able to access the **Nanonull.sqlite** database file from the working directory; otherwise, the job will fail to execute successfully.

5. Click **Save**.

Invoking the Web service

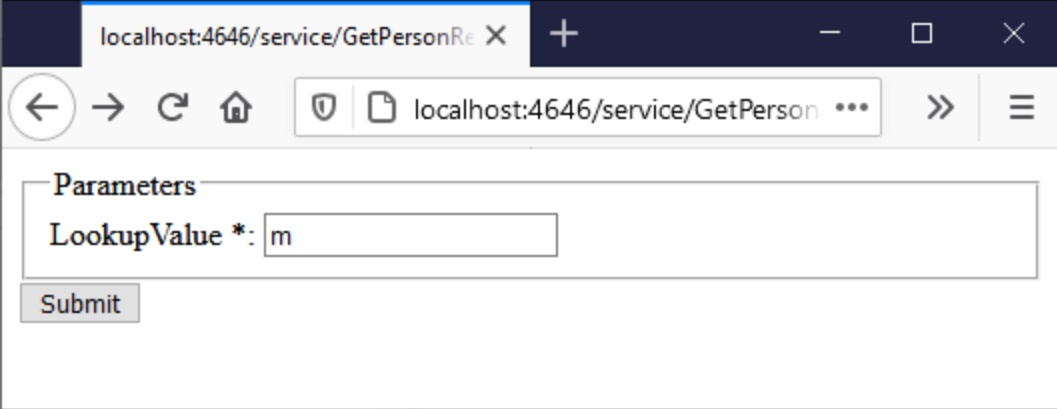
At this stage, you have completed the job configuration. Because this job is exposed as a Web service, you can run it in any of the following ways:

- Go to **Home**, and then click **Show all active triggers and services**. Next, click the job's URL displayed in the "Info" column.
- Enter <http://127.0.0.1:4646/service/GetPersonRecordsService> in the browser's address bar. Note that this URL works only if the *FlowForce Server* service listens at the default host address and port name. If you have defined other host and port settings in the [Configuration page](#)⁴⁷, change the address accordingly.
- If you set the optional **Host name** field of FlowForce Server from the [Setup Page](#)⁴⁷, you can execute the web service call directly from the job configuration page, by clicking the  button adjacent to the **Make this job available via HTTP** check box. The button is not displayed otherwise.

If prompted for credentials when accessing the Web service, supply the same credentials you use to log on to FlowForce Server.

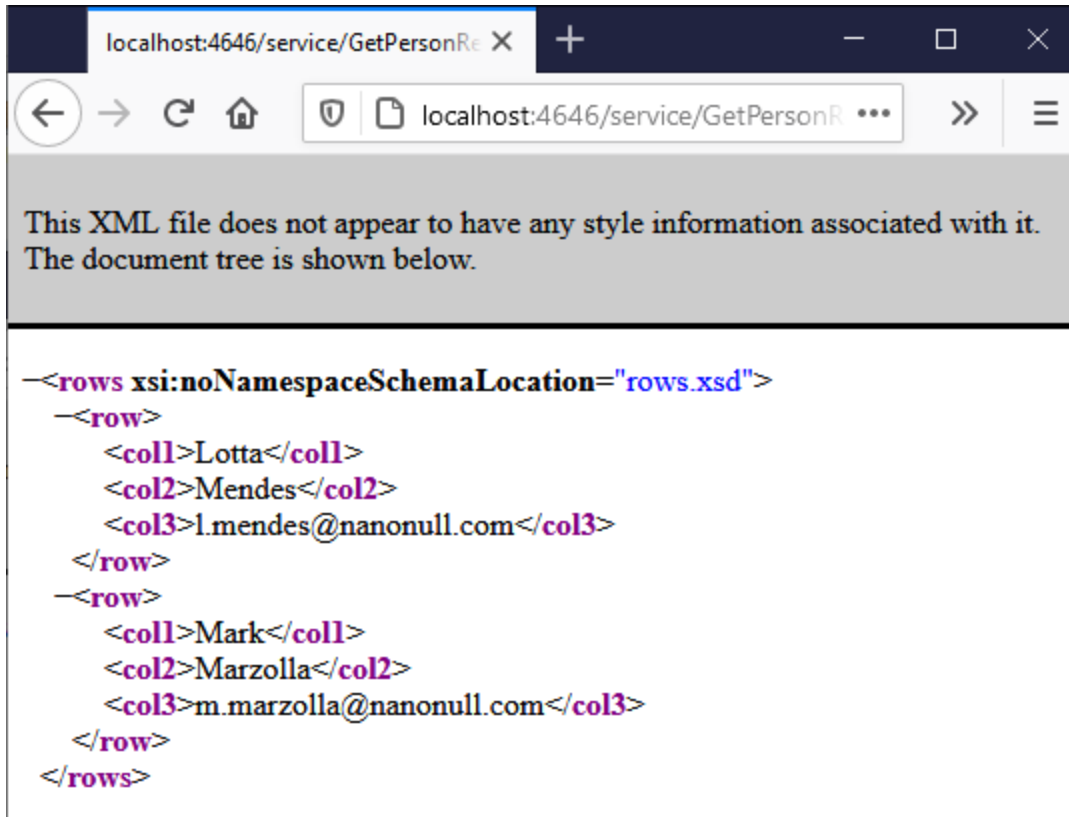
Supplying your FlowForce Server user credentials for HTTP authentication is only for testing purposes. For production, it is recommended that you create a new FlowForce user, grant the **Service - Use** permission to this user on the container where the job is, and then access the Web service with the corresponding user account. To disable HTTP authentication and make the Web service public, grant the **Service - Use** permission to the user **Anonymous**, see [How Permissions Work](#)⁹².

Since this job takes parameters, you will be prompted to supply a parameter value when you access the Web service from the browser.



The screenshot shows a web browser window with the address bar containing `localhost:4646/service/GetPersonRe`. Below the address bar, there is a "Parameters" section with a text input field labeled "LookupValue *:" containing the value "m". A "Submit" button is located below the input field.

If you enter a valid directory path like **M**, for example, the job will query the database and return only the rows where the person's last name begins with "M", for example:



On job failure, a "Service execution failed" error is displayed in the browser. If you see this error, check the [Job Log](#)¹⁰⁴ for further information.

5.9 Post JSON to FlowForce Web Service

This example shows you how to create a FlowForce Web service that accepts POST requests carrying JSON data in the HTTP request body. Secondly, it illustrates how to call the Web service from a client like MapForce.

In this example, the Web service will be configured to accept JSON data. You could also post XML or other content to a service created with FlowForce Server in a similar way as shown below. The Web service is intended to be very simple so it will merely accept JSON data and save it locally without any further processing. It is possible to further extend the job to validate the JSON data with RaptorXML Server, or process it, although this will not be done in this example.

This example specifically illustrates the case when data is posted in the body of the HTTP request, not as a parameter. For an example that invokes a Web service with parameters, see [Expose a Job as a Web Service](#)³³¹.

Prerequisites

- Required licenses: FlowForce Server, MapForce Enterprise Edition.

Remarks

FlowForce Server provides a quick way to create the Web service. MapForce Enterprise Edition acts as a client that calls the Web service created with FlowForce Server. You may also use a different client and achieve the same result.

- The *FlowForce Web Server* and *FlowForce Server* services must be listening at the configured [network address and port](#)⁴⁷
- You have a FlowForce Server user account with permissions to one of the containers (by default, the **/public** container is accessible to any authenticated user)
- This job saves input data received by the Web service to a local working directory, **C:\FlowForceExamples\PostJson**. This directory (or a similar one) must exist on the machine where FlowForce Server runs, and your operating system user account must have rights to write to this directory.

Creating the FlowForce job


Log in to the FlowForce Server Web administration interface, open the **/public/Examples** container, and then click **Create Job**. Next, enter a name and, optionally, a description for the Web service you are creating.

Note: The **public/Examples** container should already exist if you followed the previous examples; otherwise, create it using the **Create | Create Container** command.



Create job in /public/Examples


Job name:

Job description:

In order for the job to treat the POST data as arbitrary content, it must have exactly one parameter of type *stream*. To create the parameter, click **Add parameter** , enter a parameter name (in this example, "data"), and select *stream* as data type.


Job Input Parameters

 Type:  Description:




Next, add a new execution step and configure it as follows:

Execution Steps



Execute function

Parameters:	Source:	<input type="text" value="{as-file(data)}"/>
	Target:	<input type="text" value="file(instance-id()).json"/>
	Overwrite:	<input type="text" value=""/>
	Abort on error:	<input type="text" value=""/>
	Working directory:	<input type="text" value="C:\FlowForceExamples\PostJson"/>

The execution step above calls the FlowForce built-in [copy](#)²⁴¹ function. The expression shown in the "Source" text box converts the input received by the Web service to a file by using the [as-file](#)²¹⁴ expression function (recall that the input parameter was named **data** in a previous step). To obtain this expression automatically, click the **Set to**  button next to the "Source" text box and then select **data**.

The "Target" text box contains an expression that produces a unique file name each time when the job is invoked. To obtain the unique file name, the FlowForce [instance-id](#)²³² expression function is called; therefore, the JSON file name will look something like "file35.json", and the number will be different with each job call (corresponding to the ID of that FlowForce job instance). You could also enter a full path, but it is not necessary if the "Working directory" path is set, as it was done in this example. When you set the working directory path, any relative file name will be resolved relative to the working directory path.

The directory **C:\FlowForceExamples\PostJSON** (or a similar one if you changed the path) must exist and your operating system user account must have rights to write to it.

Under "Service", select the **Make this job available via HTTP** check box, and enter "PostJsonService" or a similar name for the new Web service. Take notice of the service name; you will need it to call the Web service.

Service

Make this job available via HTTP at URL

Under "Credentials", select an existing credential record or specify a local credential (see also [Credentials](#)¹⁶²). These must be the credentials of the user account on the operating system where FlowForce Server runs.

Credential


Run job using credential: Select existing credential:
 Define local credential: User name:
 Password:

Note: Do not confuse these credentials with the ones used to access the FlowForce Server Web administration interface.

Click **Save**. You are now ready to call the new Web service from a client.

Calling the Web service from a browser

You can call the Web service from a browser in any of the following ways:

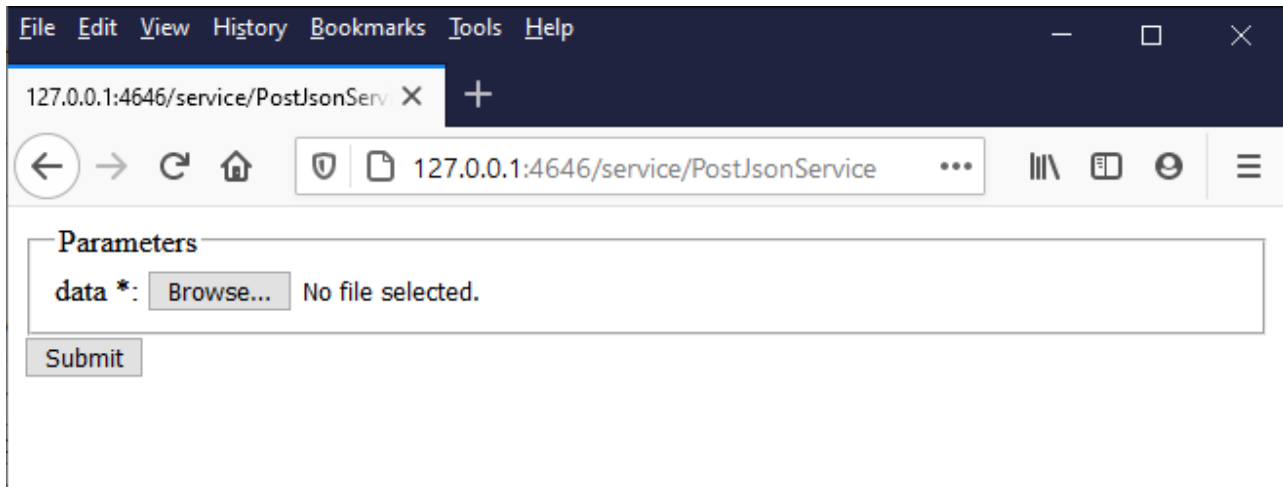
- Go to **Home**, and then click **Show all active triggers and services**. Next, click the job's URL displayed in the "Info" column.
- Enter <http://127.0.0.1:4646/service/PostJsonService> in the browser's address bar. Note that this URL works only if the *FlowForce Server* service listens at the default host address and port name. If you have defined other host and port settings in the [Configuration page](#)⁴⁷, change the address accordingly.
- If you set the optional **Host name** field of FlowForce Server from the [Setup Page](#)⁴⁷, you can execute the web service call directly from the job configuration page, by clicking the  button adjacent to the **Make this job available via HTTP** check box. The button is not displayed otherwise.

If prompted for credentials when accessing the Web service, supply the same credentials you use to log on to FlowForce Server.

Supplying your FlowForce Server user credentials for HTTP authentication is only for testing purposes. For production, it is recommended that you create a new FlowForce user, grant the **Service - Use** permission

to this user on the container where the job is, and then access the Web service with the corresponding user account. To disable HTTP authentication and make the Web service public, grant the **Service - Use** permission to the user **Anonymous**, see [How Permissions Work](#)⁹².

Because the job was configured to expect a stream as parameter, you are now prompted to enter the parameter value in the browser. Click **Browse** and select the JSON file to be submitted in the POST request.



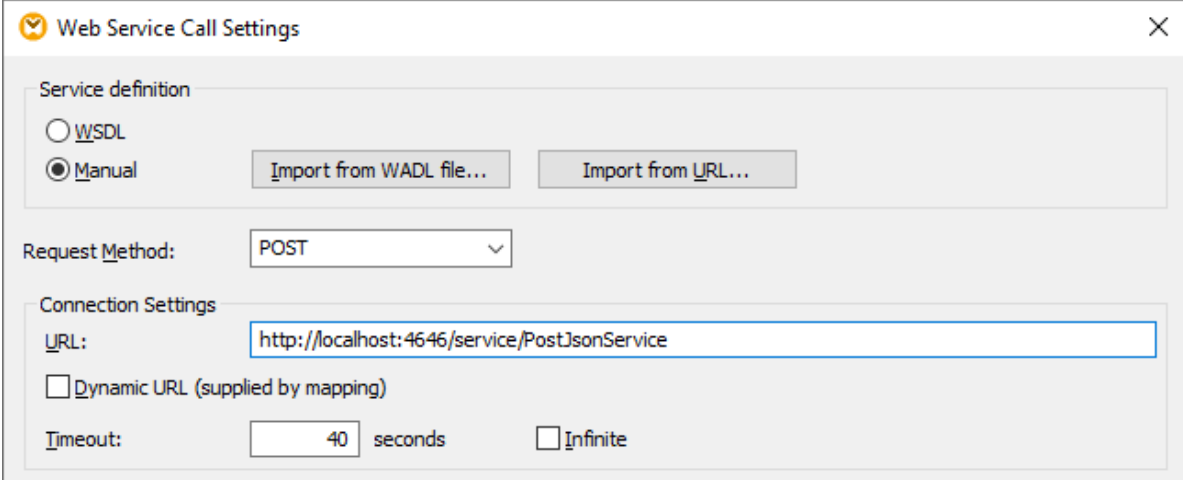
When you click **Submit**, FlowForce Server processes the job and outputs the response to the browser.

If the job executes successfully, the browser displays "true" and the JSON file is saved to the working directory **C:\FlowForceExamples\PostJson**. Otherwise, if you see an execution error, refer to the job log for more details, see [Viewing the Job Log](#)¹⁰⁴.

Calling the Web service from MapForce

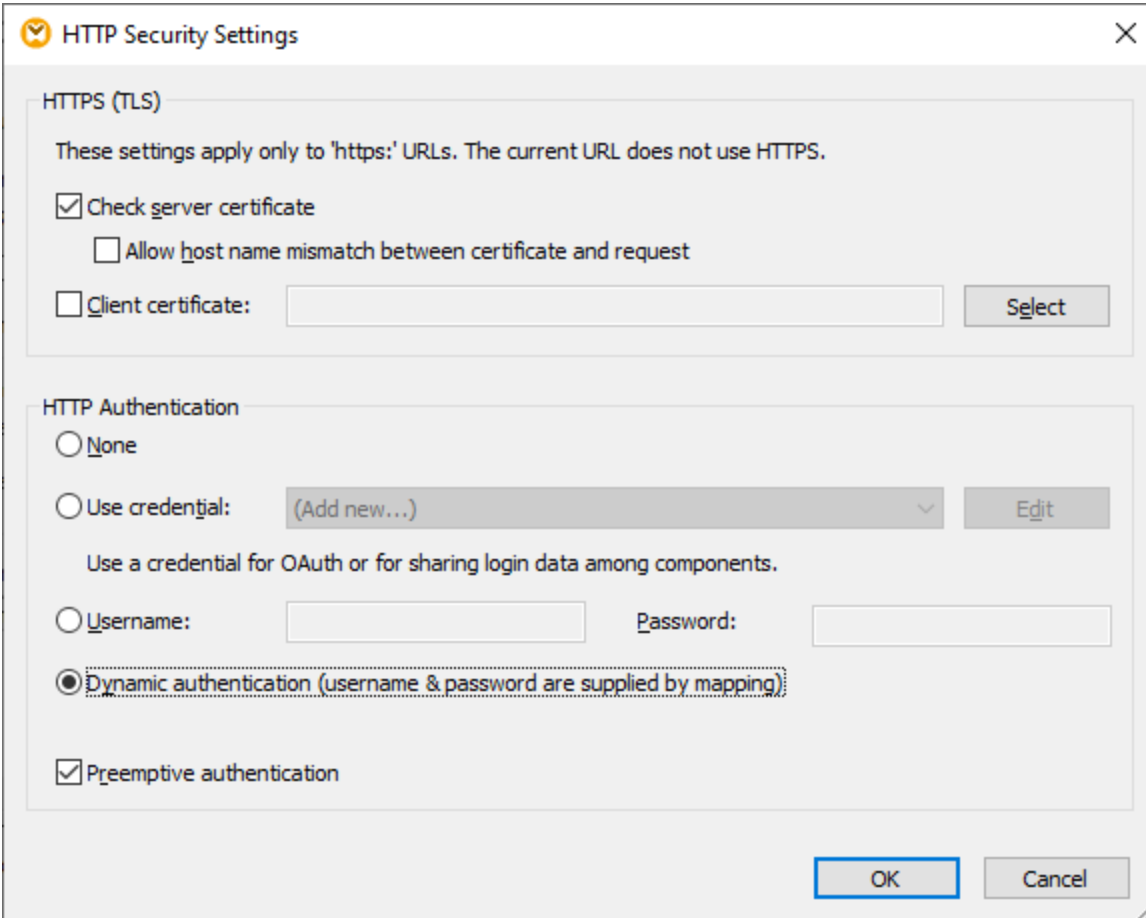
You can also call the Web service from a client other than the Web browser, for example, from MapForce Enterprise Edition.

1. On the **File** menu, click **New** to create a new mapping.
2. On the **Output** menu, click **Built-in Execution Engine**.
3. On the **Insert** menu, click **Web Service Function**. The Web Service Call Settings dialog box opens.
4. Click **Manual**, choose **POST** as request method, and enter the URL of the web service in the URL box. This is the same URL that was used to test the Web service from the browser.



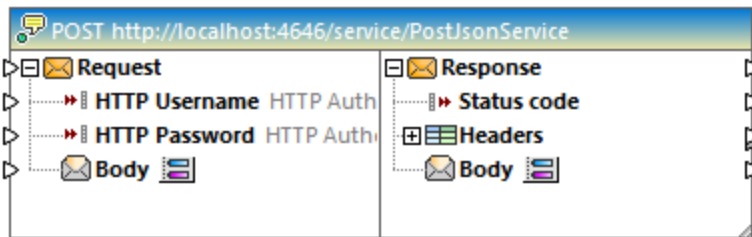
The "Web Service Call Settings" dialog box is shown. It has a title bar with a yellow icon and a close button. The "Service definition" section has two radio buttons: "WSDL" (unselected) and "Manual" (selected). There are two buttons: "Import from WADL file..." and "Import from URL...". The "Request Method:" dropdown is set to "POST". The "Connection Settings" section has a "URL:" text box containing "http://localhost:4646/service/PostJsonService". There is a checkbox for "Dynamic URL (supplied by mapping)" which is unchecked. The "Timeout:" section has a text box with "40" and the word "seconds", and an unchecked checkbox for "Infinite".

5. Click the **Edit** button next to "HTTP Security Settings", and select the **Dynamic authentication** check box. This makes it possible to supply the credentials interactively as input parameters to the mapping when the mapping runs. For information about the **Use credential** option, see [Credentials in Mapping Functions](#)⁴²¹. Entering the username and password directly in this dialog box is supported only for backward compatibility and is not recommended.

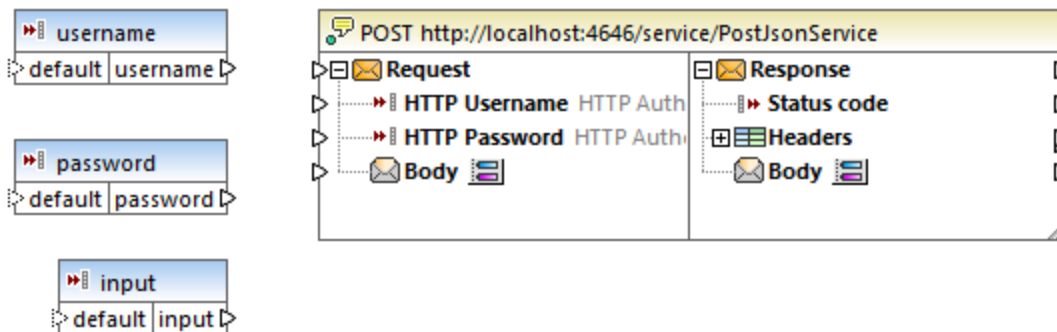


The "HTTP Security Settings" dialog box is shown. It has a title bar with a yellow icon and a close button. The "HTTPS (TLS)" section has a note: "These settings apply only to 'https:' URLs. The current URL does not use HTTPS." There are three checkboxes: "Check server certificate" (checked), "Allow host name mismatch between certificate and request" (unchecked), and "Client certificate:" (unchecked). There is a text box for the client certificate and a "Select" button. The "HTTP Authentication" section has three radio buttons: "None" (unselected), "Use credential:" (unselected), and "Dynamic authentication (username & password are supplied by mapping)" (selected). The "Use credential:" option has a dropdown menu with "(Add new...)" and an "Edit" button. Below it is a note: "Use a credential for OAuth or for sharing login data among components." There are two text boxes: "Username:" and "Password:". The "Preemptive authentication" checkbox is checked. At the bottom right are "OK" and "Cancel" buttons.

6. Click OK to close the dialog box. The mapping now looks as follows:



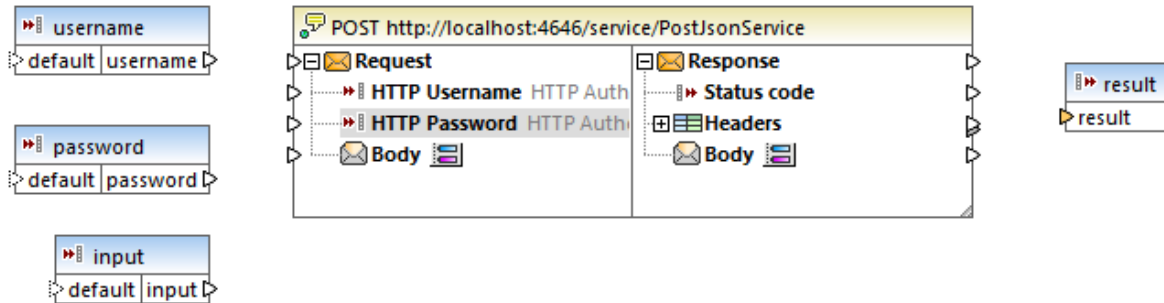
7. Add to the mapping three input parameters, by selecting the **Insert | Insert Input** menu command. The first two will supply the username and password, respectively. The third will supply the JSON data.



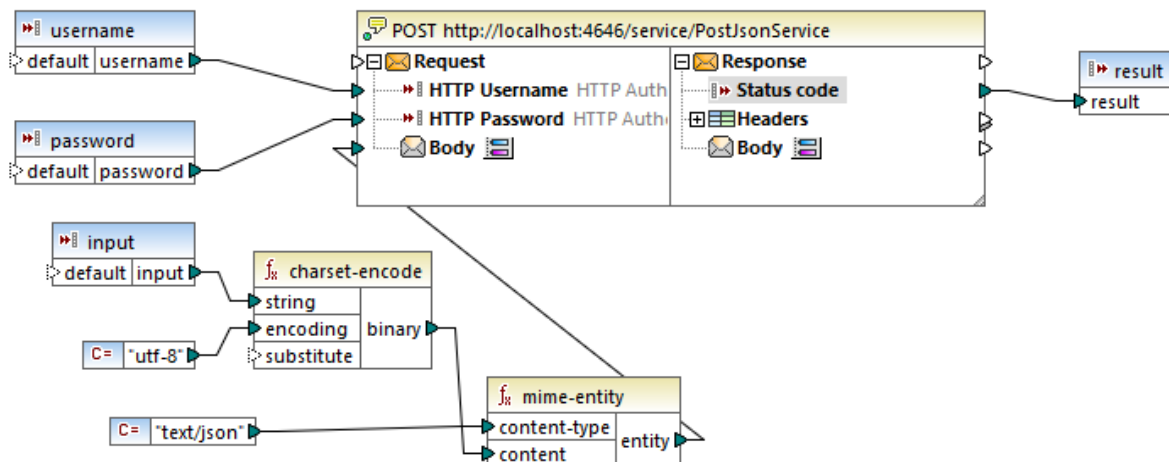
8. Double-click each of the input components above, and enter a design-time execution value to be used for previewing the mapping. For the first two parameters, enter the username and password required to access the Web service—these are necessary to run the mapping, and, for security reasons, it's not recommended to save them in the mapping file. For the parameter that will supply JSON data, enter some sample JSON data to be used for executing this mapping at design time, like the one shown below:

Note: The sample JSON data shown here is very short, for demo purposes. When MapForce Server runs the mapping, you can supply the JSON data as input parameter to the mapping from an actual JSON file.

9. Add the output of the mapping, by selecting the **Insert | Insert Output** menu command.



10. Drag the **charset-encode** and **mime-entity** functions from the Libraries window and make all the connections as shown below. You will also need to add two constants, by selecting the **Insert | Constant** menu command.



In the mapping above, the JSON input is provided to the mapping by means of a simple input component. The **charset-encode** and **mime-entity** functions are MapForce built-in functions that prepare the body of the HTTP request. The status code returned by the Web service is mapped to the result returned by the mapping.

Preparing the body of the HTTP request in an unstructured manner as shown above is just one of the ways to send data in the POST request. For JSON and XML structures, you can enter the JSON or XML schema of the request in the "Web Service Call Settings" dialog box instead. In this case, the body of the Web service component provides mapping inputs (connectors) based on the JSON/XML structure of the request.

You can now execute the mapping with MapForce, by clicking the **Output** tab. If an error occurs, it is displayed in the Messages window. To debug, you may need to check the FlowForce Server log as well (assuming that the POST request reached the server). Otherwise, if execution is successful, the following happens:

1. The HTTP status code "200" is displayed in the **Output** pane.
2. On the server side, the submitted JSON data is written to a file and saved to the **C:\FlowForceExamples\PostJson** directory.

The exact behavior of the mapping in case of an error can be further configured from MapForce. Also, the mapping can be run with MapForce Server, or be deployed to FlowForce Server, and turned into a job or even

another Web service. For further information, refer to MapForce documentation
<https://www.altova.com/documentation>.

5.10 Cache Job Results

This example shows you how to cache the result of a job (referred to as cache producer) and use it in another job (referred to as cache consumer). Both jobs will be exposed as Web services with the following behavior:

- When the cache producer Web service is invoked, it lists recursively the contents of the directory, creates or updates the cache, and then outputs the result in the browser;
- When the cache consumer Web service is invoked, it reads the cache created by the cache producer service and outputs the result in the browser.

Our goal is to compare the execution time of both jobs, and see that the second job executes significantly faster than the first job, since it consumes cached data.

Prerequisites

- Required licenses: FlowForce Server
- The *FlowForce Web Server* and *FlowForce Server* services must be listening at the configured [network address and port](#) ⁴⁷
- You have a FlowForce Server user account with permissions to one of the containers (by default, the **/public** container is accessible to any authenticated user).

Note: Although this example uses Windows paths and commands, you can still test it on other operating systems, if you change the paths and the commands accordingly.

Configuring the job

1. Click **Configuration**, and then navigate to the **/public/Examples** container. The **public/Examples** container should already exist if you followed the previous examples; otherwise, create it using the **Create | Create Container** command.
2. Click **Create**, and then select **Create Job**.
3. In the Job Name box, enter **CachedResult**.
4. Under "Execution Steps", add a new execution step with the following settings:

Execute function	Browse for the <code>/system/shell/commandline</code> function.
Command	Enter the following shell command: <code>dir /s</code> On Windows, this command lists <i>recursively</i> the contents of the working directory (see the next setting).
Working directory	Set the value to a directory on the machine where FlowForce Server runs, for example: <code>C:\</code> If you would like to use a different directory, choose one that is big enough so that it takes at least 20-30 seconds to list the directory contents recursively.

Assign this step's result to	We will need to refer to the value returned by the execution step in a subsequent step, so it must have a name. For the scope of this example, enter dir as value of this field.
------------------------------	---

5. Under Execution Steps, add a new execution step with the following settings:

Execute function	Browse for the /system/compute function.
Expression	Enter the following FlowForce Server expression: stdout(dir) The stdout function converts the raw result returned by the previous execution step into a stream of data (see Step Result Functions ²⁰²).

6. Under "Execution Result", set the return type to **stream**. As you might have noticed, we set it to the same data type as returned by the last execution step of the job. The job should now look as follows:

The screenshot displays the 'Execution Steps' configuration panel. It contains two steps:

- Step 1:** 'Execute function' set to `/system/shell/commandline`. Parameters include 'Command' (dir /s), 'Abort on error' (checked), and 'Working directory' (C:\). The result is assigned to 'dir' as 'result'.
- Step 2:** 'Execute function' set to `/system/compute`. The 'Expression' parameter is `stdout(dir)`. The result is assigned to 'name' as 'T0'.

Below the steps are buttons for 'new Execution step', 'new Choose step', 'new For-each step', 'new Error/Success handling step', and 'new Postpone step'.

The 'Execution Result' section shows 'Declare return type as:' set to 'stream'.

7. Under "Caching Result", select the **Cache the result** check box.
 8. Select the **Auto-create a new cache consumer job** check box, and then enter **DirectoryListingCachedService** as the name of the Web service.

Caching Result

Cache the result Cache is used whenever this job is called from another job.

- Add a time trigger to create and refresh the cached result or check "Initiated by consumer" option below.
- Create a job that will call this one and will benefit from the cache.
- If "Initiated by consumer" option is chosen then add Refresh or Purge Cache timers to prevent cache entries to become too old.
- In case of job input parameters present set "Initiated by consumer" option and set "Maximum number of cache entries" to expected number of

Initiated by consumer

Maximum number of cache entries:

Cache consumer job [/public/Examples/CachedResult.cached](#) is available via HTTP at URL:
 Delete consumer job

new Refresh Cache timer
new Purge Cache timer

9. Under "Service", click to select the **Make this job available via HTTP** check box, and enter **DirectoryListingService** as name of the service.


Service

Make this job available via HTTP at URL

10. Under "Credentials", select an existing credential record or specify a local credential, see [Credentials](#)¹⁶².
11. Click **Save**.

Running the job

At this stage, you have completed the configuration of both the cache producer and the cache consumer jobs. To test the performance of the non-cached service (**DirectoryListingService**) in the browser, run the job using any of the following approaches:

- Go to **Home**, and then click **Show all active triggers and services**. Next, click the job's URL displayed in the "Info" column.
- Enter <http://127.0.0.1:4646/service/DirectoryListingService> in the browser's address bar. Note that this URL works only if the *FlowForce Server* service listens at the default host address and port name. If you have defined other host and port settings in the [Configuration page](#)⁴⁷, change the address accordingly.
- If you set the optional **Host name** field of FlowForce Server from the [Setup Page](#)⁴⁷, you can execute the web service call directly from the job configuration page, by clicking the  button adjacent to the **Make this job available via HTTP** check box. The button is not displayed otherwise.

If prompted for credentials when accessing the Web service, supply the same credentials you use to log on to FlowForce Server.

Supplying your FlowForce Server user credentials for HTTP authentication is only for testing purposes. For production, it is recommended that you create a new FlowForce user, grant the **Service - Use** permission to this user on the container where the job is, and then access the Web service with the corresponding

user account. To disable HTTP authentication and make the Web service public, grant the **Service - Use** permission to the user **Anonymous**, see [How Permissions Work](#)⁹².

Note that, because the job was configured to list the contents of the C:\ directory recursively, it might take up to several minutes to complete. Refer to the job log to see how long it took for the job to complete, see [Viewing the Job Log](#)¹⁰⁴.

To test the performance of the cache consumer service (**DirectoryListingCachedService**), enter <http://127.0.0.1:4646/service/DirectoryListingCachedService> (or the equivalent URL if your host name and port are configured differently) in the browser's address bar. Since this service consumes the cache rather than executing the directory listing, it is expected to take significantly less time to complete.

5.11 Create a Job from a StyleVision Transformation

This example shows you how to create a FlowForce Server job from a StyleVision transformation. The job will consist of three steps, namely:

1. The first step will execute the StyleVision transformation.
2. Because the transformation returns a list of multiple streams, the second step will access one of the several files created by the transformation, using a FlowForce Server expression.
3. The third step will copy the file to an archive folder.

Prerequisites

- Required licenses: StyleVision Enterprise or Professional edition, StyleVision Server, FlowForce Server
- The *FlowForce Web Server* and *FlowForce Server* services must be listening at the configured [network address and port](#) ⁴⁷
- You have a FlowForce Server user account with permissions to one of the containers (by default, the **/public** container used in this example is accessible to any authenticated user).
- The job created in this example copies files from one directory to another. Therefore, on the operating system where FlowForce Server runs, ensure that both directories exist and that you have rights to create files in both directories. This example uses the following directories:
 - **C:\FlowForceExamples\GenerateHtml** — this is the job's working directory where all processing happens and relative paths are resolved.
 - **C:\FlowForceExamples\Archive** — the destination directory to which the HTML file produced by the job will be copied.

Demo files used

The StyleVision Power Stylesheet (.sps) file used in this example processes an XML file and produces output in multiple formats, including HTML. It is called **AutoCalc.sps**, and is available in the StyleVision "Examples" project, under **Examples > Basics > AutoCalc.sps**. To open the StyleVision examples project in StyleVision, click **Examples** on the **Project** menu.

Deploying the StyleVision transformation to FlowForce Server

First, let's deploy the demo .sps file from StyleVision to FlowForce Server. Deploying an .sps file means that StyleVision organizes the resources used by the transformation into an object and passes it through HTTP (or HTTPS if configured) to FlowForce Server. Once the transformation is deployed to FlowForce Server, you will create a server job from it.

To deploy the StyleVision transformation:

1. Open the **AutoCalc.sps** file in StyleVision.
2. On the **File** menu, click **Deploy to FlowForce...** If this option is disabled, make sure the **Design** tab is currently selected. When prompted to save the transformation as PXF file, leave the default settings as is, and click OK.

Deploy Transformation

Enter the host name and port of a FlowForce Administration Interface to deploy the current transformation.

Server: 127.0.0.1 Port: 8082

User: root Use SSL

Password:

Login: <Default>

Deploy As

Path: /public/AutoCalc.transformation

The path must start with a slash character.

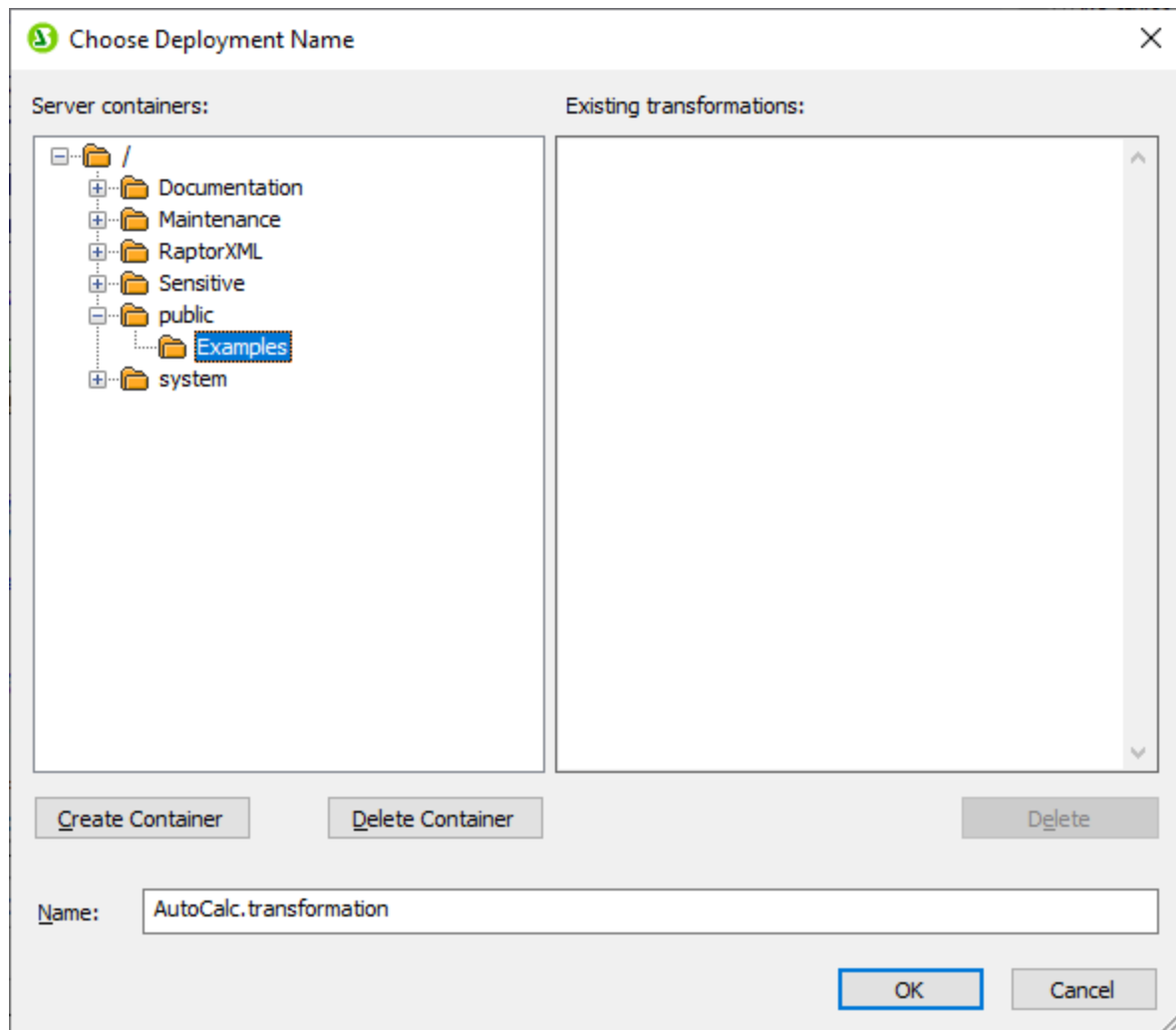
Save design changes before deploying

Open web browser to create new job

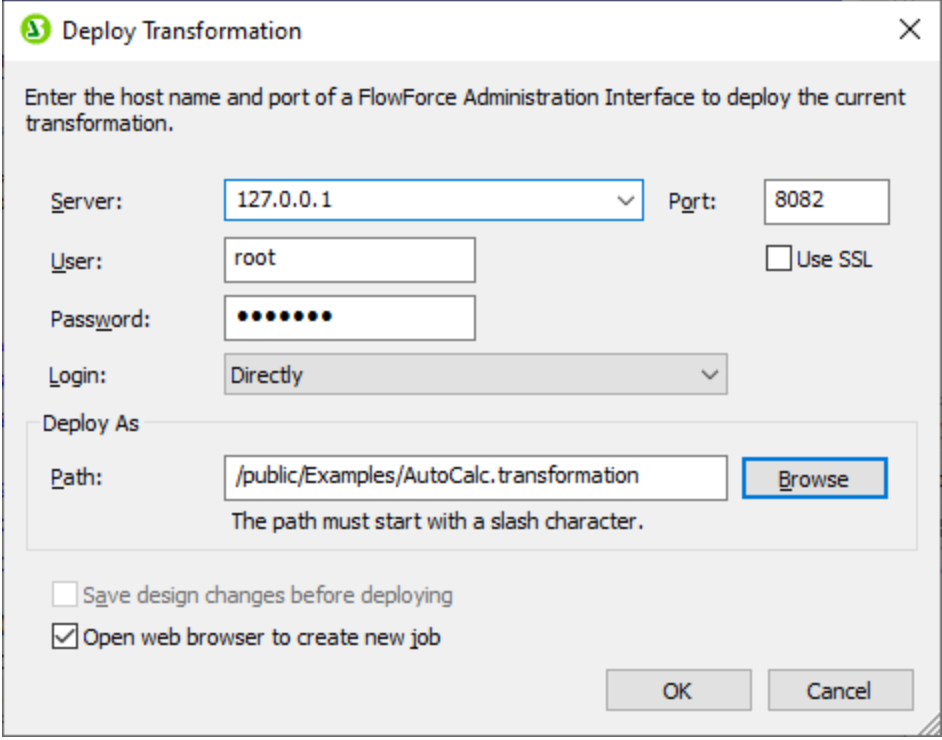
3. In the **Server** and **Port** text boxes, enter the server name and port of the Web administration interface (for example, 127.0.0.1 and 8082, if the *FlowForce Web Server* service is listening on the same machine at the default port). Change these values if you have configured a different address and port, see [Defining the Network Settings](#)⁴⁷.
4. In the **User** and **Password** text boxes, enter your FlowForce Server user name and password.
5. Select either **Directly** from the **Login** drop-down list, or leave the **<Default>** option as is.

If Directory Service integration is enabled, enter your domain user name and password, and then select your domain name from the **Login** drop-down list. For more information, see [Changing the Directory Service Settings](#)¹¹⁹.

6. The **Path** text box displays the default path where the transformation will be deployed. For consistency with other examples, click **Browse** and change the path to **/public/Examples/AutoCalc.transformation**. The **/public/Examples** container must already exist if you followed the previous examples; otherwise, you can create it by clicking **Create Container** in the dialog box below.



7. Click **OK**, and select the **Open web browser to create new job** check box on the "Deploy Transformation" dialog box.



The image shows a dialog box titled "Deploy Transformation" with a close button (X) in the top right corner. The dialog contains the following fields and options:

- Server:** A dropdown menu with "127.0.0.1" selected.
- Port:** A text box containing "8082".
- User:** A text box containing "root".
- Password:** A text box with ten dots representing a masked password.
- Login:** A dropdown menu with "Directly" selected.
- Use SSL:** An unchecked checkbox.
- Deploy As:** A section containing:
 - Path:** A text box containing "/public/Examples/AutoCalc.transformation".
 - Browse:** A button next to the path text box.
 - A note below the path: "The path must start with a slash character."
- Save design changes before deploying:** An unchecked checkbox.
- Open web browser to create new job:** A checked checkbox.
- OK** and **Cancel** buttons at the bottom right.

8. Click **OK** to deploy the transformation.

When deployment finishes, the FlowForce Server Administration Interface opens in your web browser, and a partially prefilled job page is displayed. The transformation function itself is saved at the container path specified earlier. This concludes the deployment part.

Creating the job


After you have deployed the .sps file to FlowForce Server as described above, the browser displays a partially filled job page. The first execution step is created automatically with some prefilled parameters.

You can also create the job by opening the function's page (**/public/Examples/AutoCalc.transformation**), and then clicking **Create job**.

To configure the job:

1. Change the default job name from "AutoCalc.transformaton.job" to something more descriptive, for example, "GenerateHtml". This is an optional step, but it may be necessary if the name is already used by some other job in the same container.

2. Fill in the first execution step created by default as follows:


Execute function	This field points to the StyleVision transformation function deployed earlier; leave it as is.
Parameters	<p>The InputXml field contains an XML file that is pre-packaged into the job (Data.xml). For the scope of this example, you can leave this option as is. For information about changing input instances, see Running Mappings and Transformations as Jobs⁴¹⁷.</p> <p>To declare the AutoCalc.html as output file, click the  button next to OutHtml.</p>

	<p>The option GenerateHtmlOutputAsMime is not set in this example. This option is typically necessary to prepare the HTML body of an e-mail message sent with FlowForce Server. For more information, see the send-mime²⁸⁰ function.</p> <p>The options OutRtf, OutFo, OutPdf, OutDocx declare transformation outputs in the respective formats (RTF, FO, PDF, Docx). These outputs are not used in the current example, so they can be left unchanged.</p> <p>In the Working-directory box, enter the path to a directory where FlowForce Server must save the job output. This example uses C:\FlowForceExamples\GenerateHtml as working directory.</p>
Assign this step's result to	Enter output . This explicitly gives the name "output" to the result returned by the first execution step. We will need to refer to this result subsequently.

3. Click **new Execution step** and configure it as follows:

Execute function	Browse for the /system/compute ²³⁷ function.
Expression	<p>Enter the following FlowForce Server expression:</p> <pre>as-file(nth(results(output), 0))</pre> <p>This expression instructs FlowForce Server to do the following:</p> <ol style="list-style-type: none"> 1. Call the expression function results to get the list returned by output declared previously. 2. Pass this list to function nth to get the first item in the list. Since the list index is zero-based, we are using 0 as second argument of function nth. 3. Pass the value to the as-file function to declare it as a file.
Assign this step's result to	Enter html_file . This instructs FlowForce Server that the result returned by the step has the name html_file . We will need to refer to this result subsequently.

4. Click **new Execution step** and configure the step as follows:

Execute function	Browse for the /system/filesystem/copy ²⁴¹ function.
Source	Click  , and then select html_file .
Target	C:\FlowForceExamples\Archive\AutoCalc.html
Overwrite	Select the Overwrite check box.
Working directory	C:\FlowForceExamples\GenerateHtml

At this stage, the "Execution Steps" section of the job page should look as follows:

Execution Steps

+

Execute function

Parameters:	InputXml:	<input type="text" value="altova://packagedfile/Data.xml"/>
	OutHtml:	<input type="text" value="AutoCalc.html"/>
	GenerateHtmlOutputAsMime:	<input style="border: 1px solid #ccc; border-radius: 50%;" type="button" value="+"/>
	OutRtf:	<input style="border: 1px solid #ccc; border-radius: 50%;" type="button" value="+"/>
	OutFo:	<input style="border: 1px solid #ccc; border-radius: 50%;" type="button" value="+"/>
	OutPdf:	<input style="border: 1px solid #ccc; border-radius: 50%;" type="button" value="+"/>
	OutDocx:	<input style="border: 1px solid #ccc; border-radius: 50%;" type="button" value="+"/>
	Working-directory:	<input type="text" value="C:\FlowForceExamples\GenerateHtml"/>

= Assign this step's result to as ReturnTypeId, ReturnTypeIdDocx, ReturnTypeIdHtml,

+

Execute function

Parameters:	Expression:	<input type="text" value="as-file(nth(results(output), 0))"/>
-------------	-------------	---

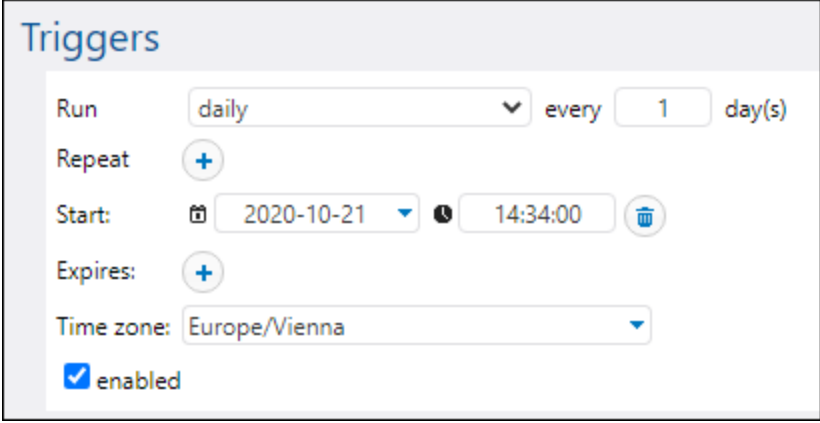
= Assign this step's result to as T0

+

Execute function

Parameters:	Source:	<input type="text" value="{html_file}"/>
	Target:	<input type="text" value="C:\FlowForceExamples\Archive\AutoCalc.html"/>
	Overwrite:	<input checked="" type="checkbox"/>
	Abort on error:	<input style="border: 1px solid #ccc; border-radius: 50%;" type="button" value="+"/>
	Working directory:	<input type="text" value="C:\FlowForceExamples\GenerateHtml"/>

5. Under "Triggers", click **new Timer**.
6. Next to "Run", set the timer to run **Daily** every **1** days. Next to "Start", select a date and time when the job must start, for example:



Triggers

Run: daily every 1 day(s)

Repeat: +

Start: 2020-10-21 14:34:00

Expires: +

Time zone: Europe/Vienna

enabled

9. Under "Credentials", select an existing credential record or specify a local credential. For details, see [Credentials](#)¹⁶².
10. Click **Save**.

Running the job

At the time and date specified in the trigger, FlowForce Server executes the StyleVision transformation job. If the job executes successfully, the **AutoCalc.html** file becomes available in the **C:\FlowForceExamples\Archive** directory. To see whether the job executed successfully, refer to the [job log](#)¹⁰⁴.

5.12 Validate a Document with RaptorXML

This example shows you how to create a job which validates an XML Schema file. This example shows probably the easiest way to validate a file, because it does not use conditional error handling and does not write the validation result to a custom log file or to the browser. The validation result will be available only in the FlowForce Server log. For a more complex validation job example, see [Validate XML with Error Logging](#)³⁶¹.

The validation job used in this example calls the **valany** function of RaptorXML Server. The **valany** function validates a document based on its type. It takes the file to validate as the only mandatory parameter, and it can be used to validate XML files, XML schemas, DTD schemas, and other file types. For a list of RaptorXML functions, see the RaptorXML documentation (<https://www.altova.com/documentation>).

Prerequisites

- Required licenses: FlowForce Server, RaptorXML (or RaptorXML+XBRL) Server
- The *FlowForce Web Server* and *FlowForce Server* services must be listening at the configured [network address and port](#)⁴⁷
- You have a FlowForce Server user account with permissions to one of the containers (by default, the **/public** container used in this example is accessible to any authenticated user).

Demo files used

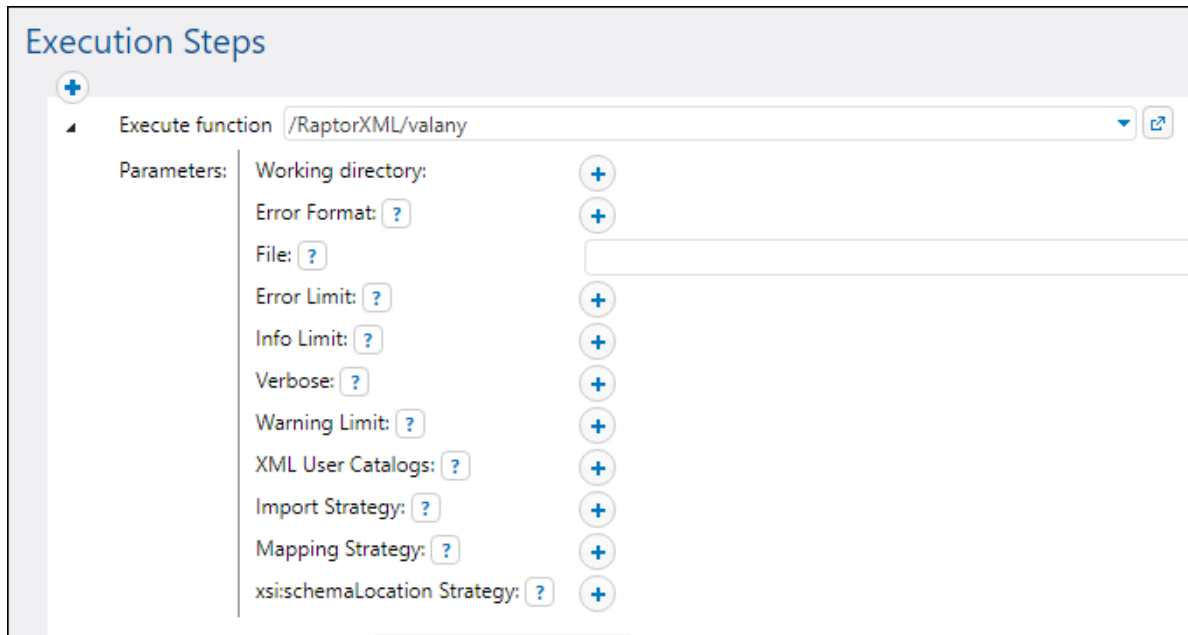
This example job validates the **address.xsd** file available in the RaptorXML Server installation folder, at the following path: **C:\Program Files\Altova\RaptorXMLServer2024\examples\address.xsd**.

On a 64-bit Windows running 32-bit RaptorXML Server, the path is **C:\Program Files (x86)\Altova\RaptorXMLServer2024\examples\address.xsd**, unless you installed RaptorXML Server in a different folder.

You can use any other XML schema file as well.

Creating the job

1. Log in to the FlowForce Server Web administration interface and open the **/public/Examples** container. The **public/Examples** container should already exist if you followed the previous examples; otherwise, create it using the **Create | Create Container** command.
2. Click **Create Job**. Next, enter a name and, optionally, a description for the job you are creating. This example uses "ValidateSchema" as job name.
3. Click **new Execution step**.
4. Next to "Execute function", browse for the **/RaptorXML/valany** function. Note that the mandatory parameter **File** is shown as an expanded field.



Note: The **valany** function exists directly under the "RaptorXML" container and also in any container that corresponds to a specific RaptorXML release, for example, "2024". For information about differences between the two, see [Generic versus release-specific RaptorXML functions](#)⁴³⁴.

5. In the **File** text box, enter the path to the schema file that you want to validate, for example, **C:\Program Files\Altova\RaptorXMLServer2024\examples\address.xsd**.
6. Under "Triggers", click **new Timer** and create a trigger that will run the job at a specific time in future. For details, see [Timer Triggers](#)¹⁵².
7. Under "Credentials", select an existing credential record or specify a local credential. For details, see [Credentials](#)¹⁶².
8. Click **Save**.

Running the job

The job will run at the date and time specified in the trigger. To see whether the job executed successfully, refer to the [job log](#)¹⁰⁴. Specifically, in the [Instance Log](#)¹⁰⁶ page, an entry like the one below indicates successful validation:

```
file:///C:/Program%20Files/Altova/RaptorXMLServer2021/examples/address.xsd: runtime="0ms"
result="OK" cmd="valxsd"
```

If the file did not validate, the job execution is considered failed (since at least one of the steps has failed), so an error is reported in the log. In this case, the logged entry displays `result="Fail"` along with details about the validation error.

5.13 Validate XML with Error Logging

This example shows you how to create a job which validates an XML file against a schema. If the job fails due to any reason, the error details will be written to a log file. For validation, we will use the `valxml-withxsd` function of RaptorXML Server running under FlowForce Server management. Note that, for the error logging part, the technique illustrated in this example is not dependent on RaptorXML Server and can be applied to other job types.

Note: The RaptorXML Server functions become available in FlowForce Server after RaptorXML Server is installed.

In this example, the job will be defined as a Web service, so that you can trigger it on demand, by accessing a URL from the browser. You can also add to the job a timer (or file system) trigger, similar to how this is done in other examples. You could even add to the same job a combination of a trigger and a Web service. This way, the job will run not only as defined by the trigger rules, but also on demand, when the Web service is called.

Prerequisites

- Required licenses: FlowForce Server, RaptorXML (or RaptorXML+XBRL) Server
- The *FlowForce Web Server* and *FlowForce Server* services must be listening at the configured [network address and port](#) ⁴⁷
- Your FlowForce Server user account has permissions to one of the containers (by default, the **/public** container used in this example is accessible to any authenticated user).
- The job created in this example generates a log file every time when it runs. Therefore, on the operating system where FlowForce Server runs, you must have rights to create files in some directory (this example uses the **C:\FlowForceExamples\ValidateXml** directory).

Demo files used

The XML file validated in this example is available in the RaptorXML Server installation folder, at the following path: **C:\Program Files\Altova\RaptorXMLServer2024\examples\NanonullOrg.xml**.

On a 64-bit Windows running 32-bit RaptorXML Server, the path is **C:\Program Files (x86)\Altova\RaptorXMLServer2024\examples\NanonullOrg.xml**, unless you installed RaptorXML Server in a different folder.

You could also use any other XML file for validation.


Creating the job

1. Log in to the FlowForce Server Web administration interface and open the **/public/Examples** container. The **public/Examples** container should already exist if you followed the previous examples; otherwise, create it using the **Create | Create Container** command.
2. Click **Create Job**. Next, enter a name and, optionally, a description for the job you are creating. This example uses "ValidateXml" as job name.



Create job in /public/Examples


Job name:


Job description:

- Under "Job Input Parameters", click the  button and create a new parameter of type "string as file", for example:






Job Input Parameters






 Name: Type: Default:  Description:





- Under "Execution Steps", click the  button, and then select **new error/success handling step**.


Execution Steps


 **Execute with error/success handling - on error, retry** times    

 On error    



- Under "Execute with error/success handling", click the  button, and choose to add a new execution step, with the following settings:


Execute function	Browse for the <code>/RaptorXML/valxml-withxsd</code> function.
Parameters	Next to the XML File parameter, click  and select the inputFile job input parameter declared earlier.

- Under the "On error" condition, click the  button and choose to add a new execution step, with the following settings:

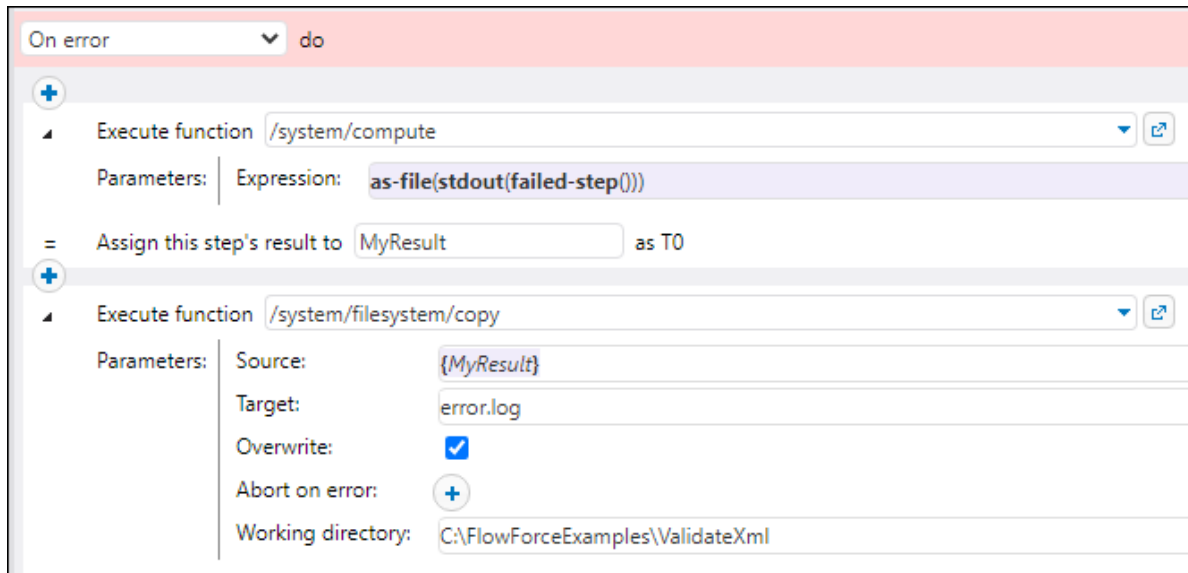
Execute function	Browse for the <code>/system/compute</code> function.
Parameters	Set the value of Expression to:

	<pre>as-file(stdout(failed-step()))</pre> <p>The parts between curly braces are two FlowForce expressions. This expression gets the output, converts it to a stream and then writes it to a file on the disk:</p> <ul style="list-style-type: none"> • The failed-step function returns the result of the failing step. This is an abstract FlowForce type that, in order to become more useful, must be supplied as argument to the exitcode, stderr, or error-message functions, see below. • The stdout function gets the standard output from the result, as a stream, assuming that there is standard output. • The as-file function creates a file from the stream. The path will be specified in a subsequent step.
Assign this step's result to	Enter a value which will uniquely identify the result of this job, for example, MyResult . By doing this, you are declaring this value as a variable, so that you can use it in a subsequent step.

7. Click the  button to add a new execution step after the previous one, with the following settings:

Execute function	Browse for the /system/filesystem/copy function.
Parameters	<p>Next to the Source parameter, click  and select the MyResult variable declared earlier.</p> <p>Next to the Target parameter, type the path where the log will be saved (in this example, the path is C:\FlowForceExamples\ValidateXml\error.log).</p> <p>Select the check box next to the Overwrite parameter. The log file is generated each time the job runs, so this ensures that the job does not fail when the log file already exists.</p> <p>Set the Working Directory parameter to C:\FlowForceExamples\ValidateXml.</p>


The "On Error" branch of the job should now look as follows:



8. Under "Service", select the **Make this job available via HTTP** check box, and enter **ValidateXmlService** as name of the service.
9. Under "Credentials", select an existing credential record or specify a local credential. For details, see [Credentials](#)¹⁶².
10. Click **Save**.

Running the job

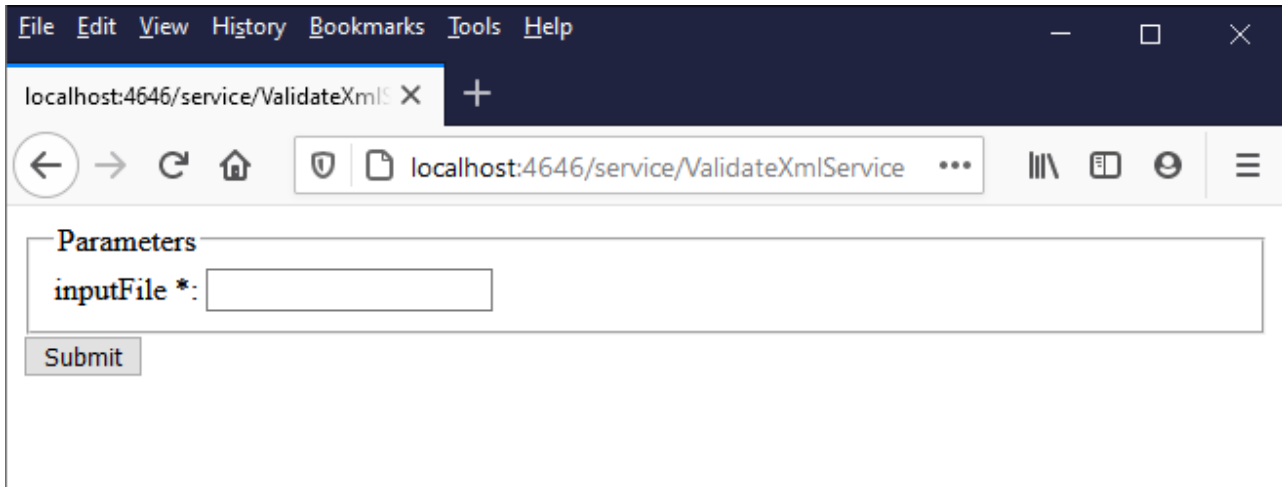
To run the job, do one of the following:

- Go to **Home**, and then click **Show all active triggers and services**. Next, click the job's URL displayed in the "Info" column.
- Enter <http://127.0.0.1:4646/service/ValidateXmlService> in the browser's address bar. Note that this URL works only if the *FlowForce Server* service listens at the default host address and port name. If you have defined other host and port settings in the [Configuration page](#)⁴⁷, change the address accordingly.
- If you set the optional **Host name** field of FlowForce Server from the [Setup Page](#)⁴⁷, you can execute the web service call directly from the job configuration page, by clicking the  button adjacent to the **Make this job available via HTTP** check box. The button is not displayed otherwise.

If prompted for credentials when accessing the Web service, supply the same credentials you use to log on to FlowForce Server.

Supplying your FlowForce Server user credentials for HTTP authentication is only for testing purposes. For production, it is recommended that you create a new FlowForce user, grant the **Service - Use** permission to this user on the container where the job is, and then access the Web service with the corresponding user account. To disable HTTP authentication and make the Web service public, grant the **Service - Use** permission to the user **Anonymous**, see [How Permissions Work](#)⁹².

Since this job was configured to expect a parameter as input, the browser displays a form where you can enter the path to the XML file that is to be validated.



Enter an XML file path in the text box (for example, **C:\Program Files\Altova\RaptorXMLServer2024\examples\NanonullOrg.xml**), and click **Submit**.

If the job executes successfully (that is, if it returns the exit code **0**), the browser displays the standard output of the job, for example:

```
file:///C:/Program%20Files/Altova/RaptorXMLServer/examples/NanonullOrg.xml: runtime="16ms"
result="OK"
```

If the job returns an exit code other than **0** (for example, due to an incorrect path, validation errors, and so on), the browser displays a "Service execution failed" message and the output is written to the **C:\FlowForceExamples\ValidateXml\error.log** file. In the event that the log file was not generated, check the [job log](#)¹⁰⁴ to identify the error. It may be the case, for example, that the `/system/filesystem/copy` function has failed because you have no permission to write to the target path.

5.14 Run XSLT with RaptorXML

This example shows you how to run an XSLT transformation with RaptorXML Server (or RaptorXML+XBRL Server) running under FlowForce Server management. The job will call the `xslt` function of RaptorXML Server. When you configure the job from the FlowForce Server configuration page, there are two ways to supply the parameters to the `xslt` function:

1. By typing key-value pairs (parameter name and value) in text boxes.
2. By entering a FlowForce Server expression.

Both ways are presented in more detail below.

Prerequisites

- Required licenses: FlowForce Server, RaptorXML (or RaptorXML+XBRL) Server
- The *FlowForce Web Server* and *FlowForce Server* services must be listening at the configured [network address and port](#) ⁴⁷
- Your FlowForce Server user account has permissions to one of the containers (by default, the **/public** container used in this example is accessible to any authenticated user).
- The job created in this example runs an XSLT stylesheet that processes an input XML file. Both files must exist in some directory on the operation system where FlowForce Server runs, and you must have rights to read and write files in this directory. This example uses the **C:\FlowForceExamples\RunXslt** directory.

Demo files

The job illustrated below will run an XSLT stylesheet called **transformation.xslt** which takes as input a file called **books.xml**, and two required parameters, "year" and "genre". The exact content of the files is shown in the code listings below. To use these files in the job, save both code listings with the indicated file names to the **C:\FlowForceExamples\RunXslt** directory.

The XSLT stylesheet looks as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:fn="http://www.w3.org/2005/xpath-
functions" exclude-result-prefixes="xs fn">
  <xsl:output method="xml" encoding="UTF-8" byte-order-mark="no" indent="yes"/>
  <xsl:param name="year" as="xs:string" required="yes"/>
  <xsl:param name="genre" as="xs:string" required="yes"/>
  <xsl:template match="/">
    <library>
      <xsl:attribute name="xsi:noNamespaceSchemaLocation"
namespace="http://www.w3.org/2001/XMLSchema-instance" select="'library.xsd'"/>
      <last_updated>
        <xsl:sequence select="xs:string(fn:current-dateTime())"/>
      </last_updated>
      <xsl:for-each select="(./books/book)[(fn:string(year) &gt; $year)]">
        <publication>
          <xsl:for-each select="@id">
```

```

        <id>
            <xsl:sequence select="fn:string(.)" />
        </id>
    </xsl:for-each>
    <author>
        <xsl:sequence select="fn:string(author)" />
    </author>
    <title>
        <xsl:sequence select="fn:string(title)" />
    </title>
    <genre>
        <xsl:sequence select="$genre" />
    </genre>
    <publish_year>
        <xsl:sequence select="xs:string(xs:integer(fn:string(year)))" />
    </publish_year>
</publication>
</xsl:for-each>
</library>
</xsl:template>
</xsl:stylesheet>

```

transformation.xslt

The input XML file looks as follows:

```

<?xml version="1.0" encoding="UTF-8"?>
<books xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="books.xsd">
    <book id="1">
        <author>Mark Twain</author>
        <title>The Adventures of Tom Sawyer</title>
        <category>Fiction</category>
        <year>1876</year>
    </book>
    <book id="2">
        <author>Franz Kafka</author>
        <title>The Metamorphosis</title>
        <category>Fiction</category>
        <year>1912</year>
    </book>
    <book id="3">
        <author>Herman Melville</author>
        <title>Moby Dick</title>
        <category>Fiction</category>
        <year>1851</year>
    </book>
    <book id="4">
        <author>Miguel de Cervantes</author>
        <title>Don Quixote</title>
        <category>Fiction</category>
    </book>
</books>

```

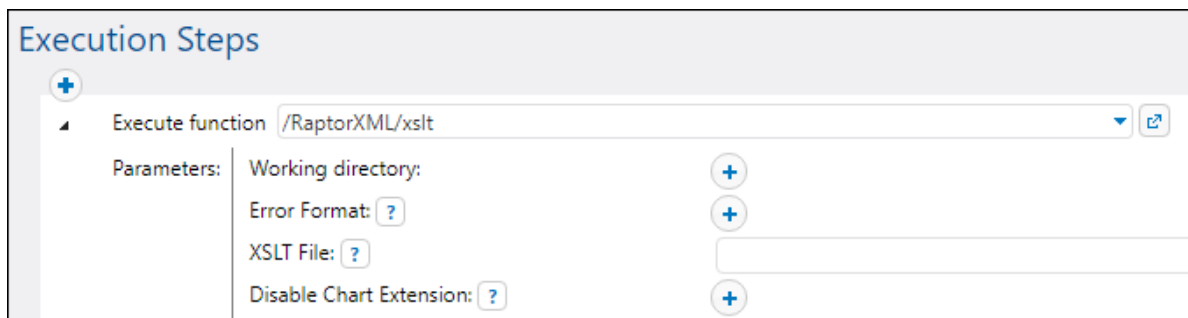
```
<year>1605</year>
</book>
</books>
```

books.xml

Creating the job

You can create a FlowForce Server jobs to run such an XSLT transformation as follows:

1. Log in to the FlowForce Server Web administration interface and open the **/public/Examples** container. The **public/Examples** container should already exist if you followed the previous examples; otherwise, create it using the **Create | Create Container** command.
2. Click **Create Job**. Next, enter a name and, optionally, a description for the job you are creating. This example uses "RunXslt" as job name.
3. Click **new Execution step**.
4. Next to "Execute function", browse for the **/RaptorXML/xslt** function.



Note: The **xslt** function exists directly under the "RaptorXML" container and also in any container that corresponds to a specific RaptorXML release, for example, "2024". For information about differences between the two, see [Generic versus release-specific RaptorXML functions](#)⁴³⁴.

5. Set the **Working directory** parameter to **C:\FlowForceExamples\RunXslt**.
6. Set the **XSLT File** parameter to **transformation.xslt**. This file must exist in the working directory.
7. Set the **XSLT Input** parameter to **books.xml**. This file must also exist in the working directory.
8. Set the **Parameters** parameter as follows:

- a. Click the **+** button next to **Parameters**. This expands a sub-section within the page, where you can add each parameter name and value individually.



- b. Click the **+** button for each new parameter that you need to add. To run the XSLT in this example, you will need to enter the parameters as follows:

The XSLT parameters are supplied to the job as key-value pairs. Notice the parameter name and value are entered in separate boxes. Also, the parameter value is enclosed within quotes.

9. Under "Service", select the **Make this job available via HTTP** check box, and enter **RunXsltService** as name of the service.
10. Under "Credentials", select an existing credential record or specify a local credential. For details, see [Credentials](#)¹⁶².
11. Click **Save**.

This concludes the job configuration part.

Supplying XSLT parameters as expression

In the job configuration above, you have supplied the parameter to the `xslt` function using text boxes. Note that there is a second way to do this, by entering a FlowForce Server expression in the **Parameters** text box, for example:


To use this second approach, click **Set to** next to **Parameters**, and then click **<Expression>**. Make sure that you type the expression very carefully in order to avoid parsing errors. The expression calls the [list](#)²¹⁰ expression function and builds a list of two key-value pairs. In each key-value pair, the key is the parameter name and the value is the parameter value. Importantly, the parameter values are again enclosed within single quotes.

To switch back to the text box layout, click **Set to** next to **Parameters**, and then click **<Value>**.

Running the job

To run the job, do one of the following:

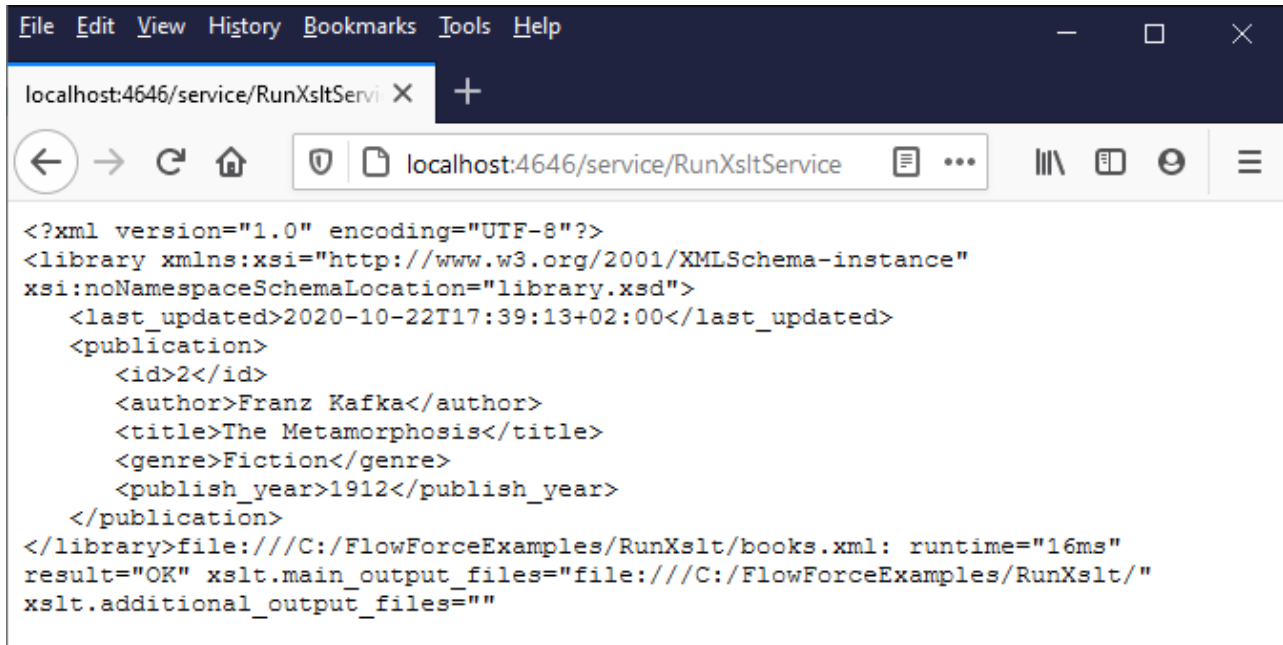
- Go to **Home**, and then click **Show all active triggers and services**. Next, click the job's URL displayed in the "Info" column.
- Enter <http://127.0.0.1:4646/service/RunXsltService> in the browser's address bar. Note that this URL works only if the *FlowForce Server* service listens at the default host address and port name. If you have defined other host and port settings in the [Configuration page](#)⁴⁷, change the address accordingly.

- If you set the optional **Host name** field of FlowForce Server from the [Setup Page](#)⁴⁷, you can execute the web service call directly from the job configuration page, by clicking the  button adjacent to the **Make this job available via HTTP** check box. The button is not displayed otherwise.

If prompted for credentials when accessing the Web service, supply the same credentials you use to log on to FlowForce Server.

Supplying your FlowForce Server user credentials for HTTP authentication is only for testing purposes. For production, it is recommended that you create a new FlowForce user, grant the **Service - Use** permission to this user on the container where the job is, and then access the Web service with the corresponding user account. To disable HTTP authentication and make the Web service public, grant the **Service - Use** permission to the user **Anonymous**, see [How Permissions Work](#)⁹².

If the job executes successfully, the browser displays the output of the job, for example:



```
<?xml version="1.0" encoding="UTF-8"?>
<library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="library.xsd">
  <last_updated>2020-10-22T17:39:13+02:00</last_updated>
  <publication>
    <id>2</id>
    <author>Franz Kafka</author>
    <title>The Metamorphosis</title>
    <genre>Fiction</genre>
    <publish_year>1912</publish_year>
  </publication>
</library>file:///C:/FlowForceExamples/RunXslt/books.xml: runtime="16ms"
result="OK" xslt.main_output_files="file:///C:/FlowForceExamples/RunXslt/"
xslt.additional_output_files=""
```

If the job fails, the browser displays a "Service execution failed" message. In this case, check the FlowForce Server [job log](#)¹⁰⁴ to identify the error.

5.15 Generate PDFs from XML Files

This example illustrates how to create a FlowForce Server job which takes as input multiple XML files and returns as output multiple PDF files. The FlowForce Server job will invoke both MapForce Server (to generate the XML output from multiple source XML files) and StyleVision Server (to convert the XML output to PDF).

This example requires a basic understanding of how MapForce mappings and StyleVision transformations work. If you are completely new to StyleVision and MapForce, it is recommended to read first the "Tutorials" chapters of MapForce and StyleVision documentation, respectively:

- [Quick Start Tutorial \(MapForce\)](#)
- [Quick Start Tutorial \(StyleVision\)](#)

Prerequisites

- Required licenses:
 - **MapForce Enterprise** or **Professional** edition. This tool enables you to design a mapping transformation (.mfd file) that converts (in this example) XML files from one schema to another.
 - **MapForce Server** or **MapForce Server Advanced Edition**. This tool enables you to run the mapping on a server, as a job.
 - **StyleVision Enterprise** or **Professional** edition. This tool enables you to design a stylesheet (.sps file) that converts an input XML file to a PDF file.
 - **StyleVision Server**. This tool enables you to run the transformation on a server, as a job.
 - **FlowForce Server**. This tool provides the means to run the transformations above as a scheduled or on demand job, change inputs if necessary, and monitor execution.
- The *FlowForce Web Server* and *FlowForce Server* services must be listening at the configured [network address and port](#) ⁴⁷
- Your FlowForce Server user account has permissions to one of the containers (by default, the **/public** container used in this example is accessible to any authenticated user).
- The job created in this example generates multiple files on the disk. Therefore, on the operating system where FlowForce Server runs, you must have rights to create files in some directory. This example uses the directory **C:\FlowForceExamples\GeneratePdfs**.

Demo files used

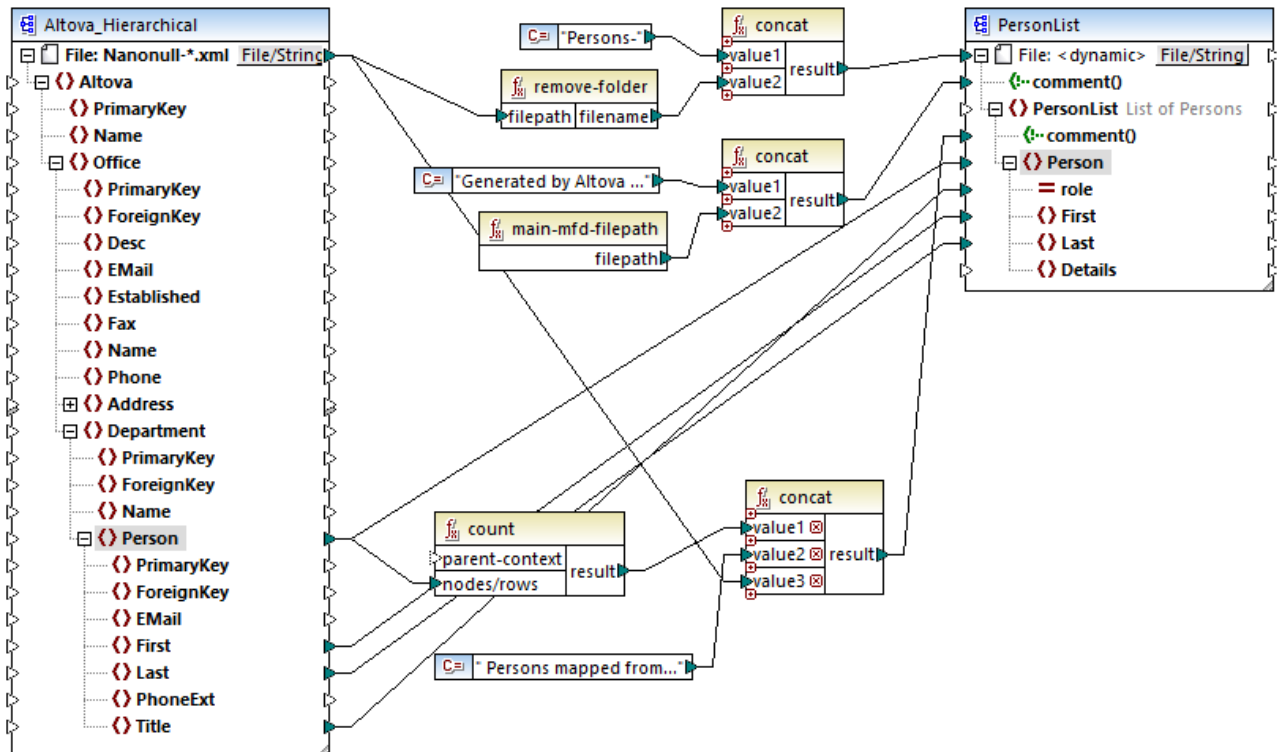
This example makes use of the following sample files, available at the following path:

<Documents>\Altova\MapForce2024\MapForceExamples

- **MultipleInputToMultipleOutputFiles.mfd** (the MapForce mapping file)
- **PersonListWithGrouping.sps** (the StyleVision transformation file)
- **Nanonull-Branch.xml**, **Nanonull-HQ.xml** (the input XML files)

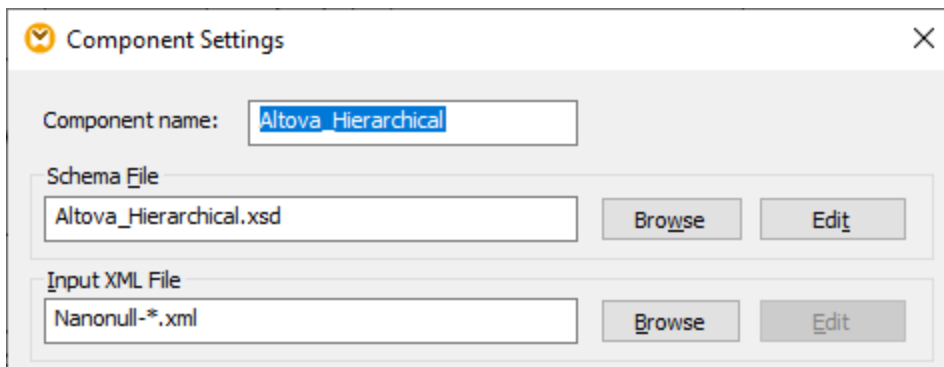
What the MapForce mapping does

As illustrated below, the mapping consists of a source component (**Altova_Hierarchical**), a target one (**PersonList**), and various intermediary MapForce built-in functions used to build miscellaneous strings to be written to the output.



MultipleInputToMultipleOutputFiles.mfd

The mapping takes as input any XML file that begins with "Nanonull-", from the directory `<Documents>\AltovaMapForce2024\MapForceExamples`. This is defined in the source MapForce component settings (in MapForce, right-click the header of the **Altova_Hierarchical** component illustrated below, and select **Properties** from the context menu). Notice that "Input File" is set to **Nanonull-*.xml**, where the asterisk is a wildcard. Literally, the input is any file which begins with "Nanonull-" and has the .xml extension.



The target component, **PersonList**, is configured to generate file names dynamically based on the file name of the source XML file. This is defined by right-clicking the **File/String** button at the top of the component, and then selecting **Use Dynamic File Names Supplied by Mapping** menu option. The connection to the "File <dynamic>" node means that a new file will be created for every value in the source. The **remove-folder**

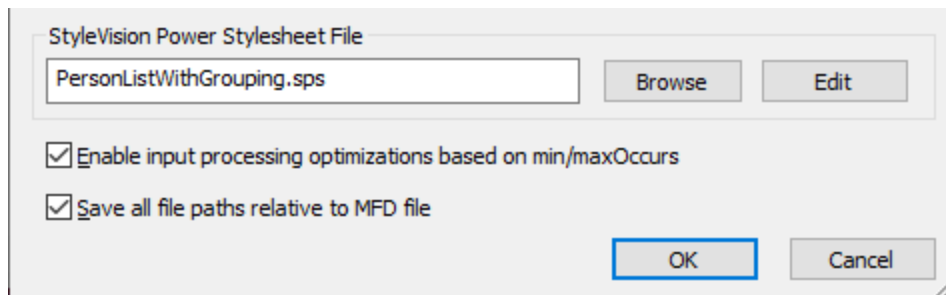
function is meant to get only the file name (without the folder) from the source path. This is then passed as value to the top **concat** function, which builds a string like *Persons-<Source filename>*.

The second **concat** function builds a string like *Generated by Altova...* followed by the complete path to the mapping file. The result is written as a comment in the target XML file.

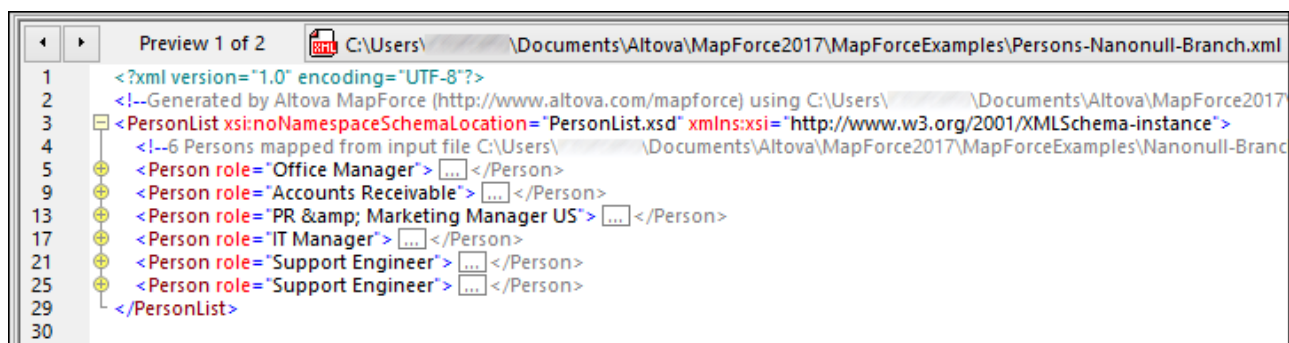
The third **concat** function uses the output of the **count** function to build a string that indicates how many person records have been mapped from the source. Again, the result is written as a comment in the target XML file.

Finally, the connection to the target **Person** node copies people data from the source to the target. An individual connection exists for each child element of **Person** that must be mapped.

In addition to this, the target component is configured to convert the generated output to PDF, for each XML generated file. Right-click the header of the target component, select **Properties**, and notice that the **StyleVision Power StyleSheet file** text box specifies a relative path to a StyleVision .sps stylesheet. The latter performs the actual conversion of XML to PDF (further discussed below).



To preview the output of this mapping directly in MapForce, click the **Output** tab available under the mapping pane. To preview the PDF result of the StyleVision transformation, click the **PDF** tab. You will notice that multiple XML's (or PDFs, respectively) are generated in the Output pane, for example:



At this stage, it is recommended to save one of the two output XML files to the disk (since, by default, MapForce generates temporary files). The file will act as a sample (working XML) if you would like to open and test the StyleVision power stylesheet in StyleVision (see next section). To save an output file, first click the **Output** tab, and then, on the **Output** menu, click **Save Output File**.

What the StyleVision transformation does

Run StyleVision and open the **PersonListWithGrouping.sps** transformation file. Recall that this file is in the same directory as the MapForce mapping discussed above, and it is referenced by the target MapForce component.

The screenshot displays the Design Overview pane for the **PersonListWithGrouping.sps** transformation. The initial document structure is as follows:


- Initial Document Section** (with [Edit Properties...](#) and [Add Header/Footer...](#) links)
- XML Root** (indicated by a green arrow icon)
- Heading**: **List of Employees** (indicated by **<h2>** and **</h2>** tags)
- Paragraph**: "This file contains all employees grouped according their role in the company." (indicated by **<p>** and **</p>** tags)
- Table**:

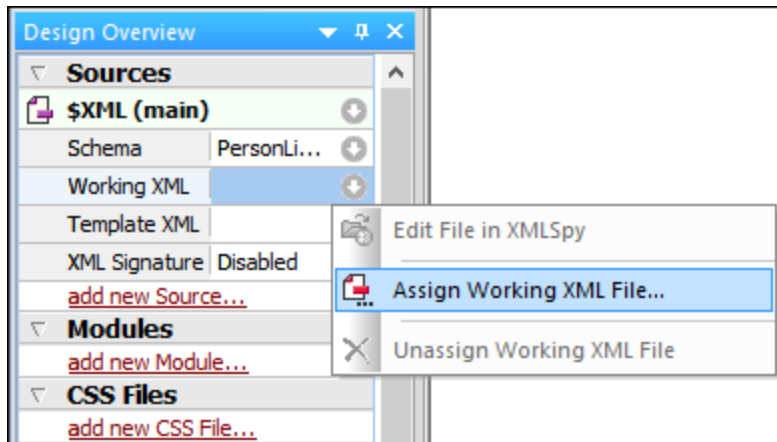
Role	First Name	Last Name
<= role >(content)< = role	<= First >(content)< = First	<= Last >(content)< = Last
- Final Paragraph**: (indicated by **<p>** and **</p>** tags)
- XML Root** (indicated by a green arrow icon)

PersonListWithGrouping.sps

The StyleVision .sps stylesheet illustrated above uses a single XML as source and creates a PDF document from it. The PDF document consists of a heading ("h2"), an introductory paragraph, a table populated dynamically, and an ending paragraph. The heading and the introductory paragraph contain static text, while the table and the ending paragraph are populated from the nodes of the source XML file, as indicated by the wrapping tags.

To preview this transformation directly in StyleVision, follow the steps below:

1. In the **Design Overview** pane, next to **Working XML**, click .



2. Select **Assign Working XML File** and browse for the XML output file saved previously from MapForce (see previous section).
3. Click the **PDF** tab.

Importantly, the .sps stylesheet is agnostic with the respect to the actual name or origin of the source XML file; it simply processes the XML file provided as input (as long as it conforms to the specified XML schema), and creates a PDF out of it. In order to automate this stylesheet so that it generates multiple PDF files, it will need to be deployed to FlowForce Server, as shown further below.

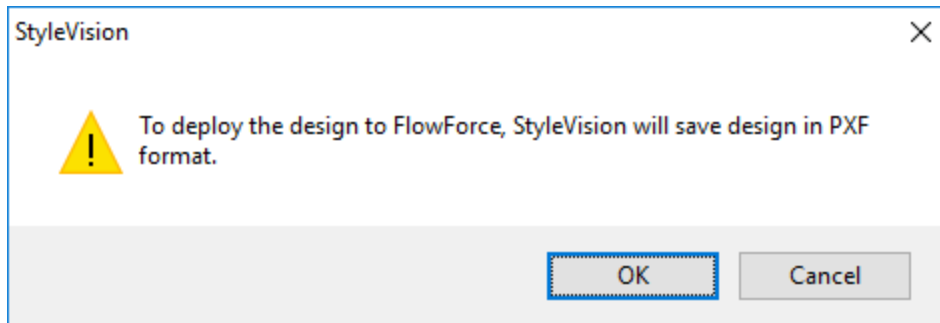
Deploy the files to FlowForce Server

So far, you have become familiar with the purpose of both the MapForce mapping and the StyleVision transformation used in this example. For more information about designing MapForce mappings and StyleVision stylesheets, refer to the documentation of these products (<https://www.altova.com/documentation.html>).

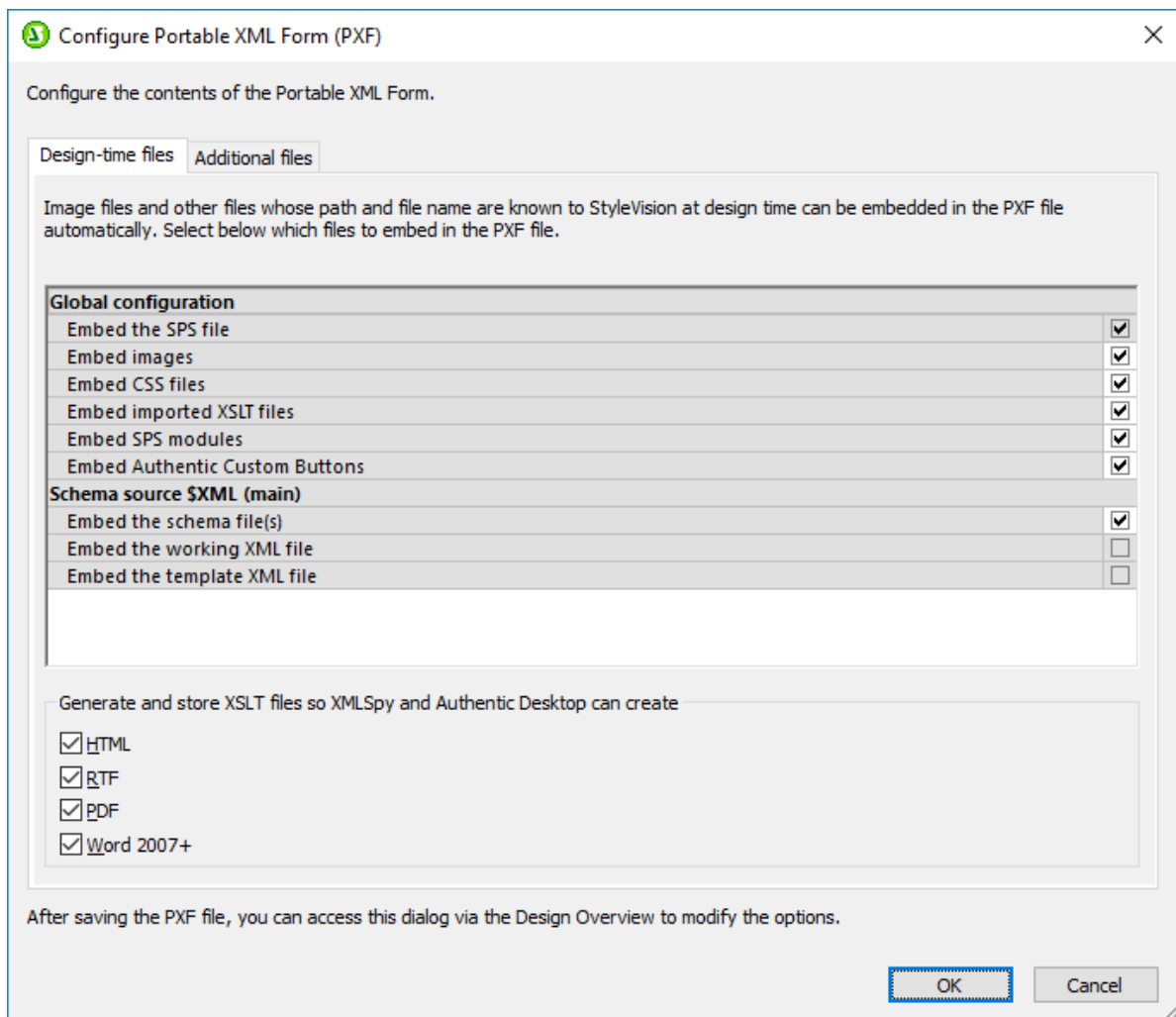
To make automation possible, both files must now be deployed to FlowForce Server. As specified in the "Prerequisites" section above, FlowForce Server must be licensed and running, and both MapForce Server and StyleVision Server must be licensed and running under FlowForce Server management. On Windows, you can use the [verifylicense](#) ⁴⁰⁴ command of each server product to check the status of its license. On other operating systems, the job execution will fail with an error message if the license is not found or valid.

To deploy the StyleVision stylesheet to FlowForce Server:

1. On the **File** menu, click **Deploy to FlowForce**. (If this command is grayed out, switch to the **Design** tab first.)
2. When prompted that the design file will be saved as PXF (Portable XML Form) format, click OK.



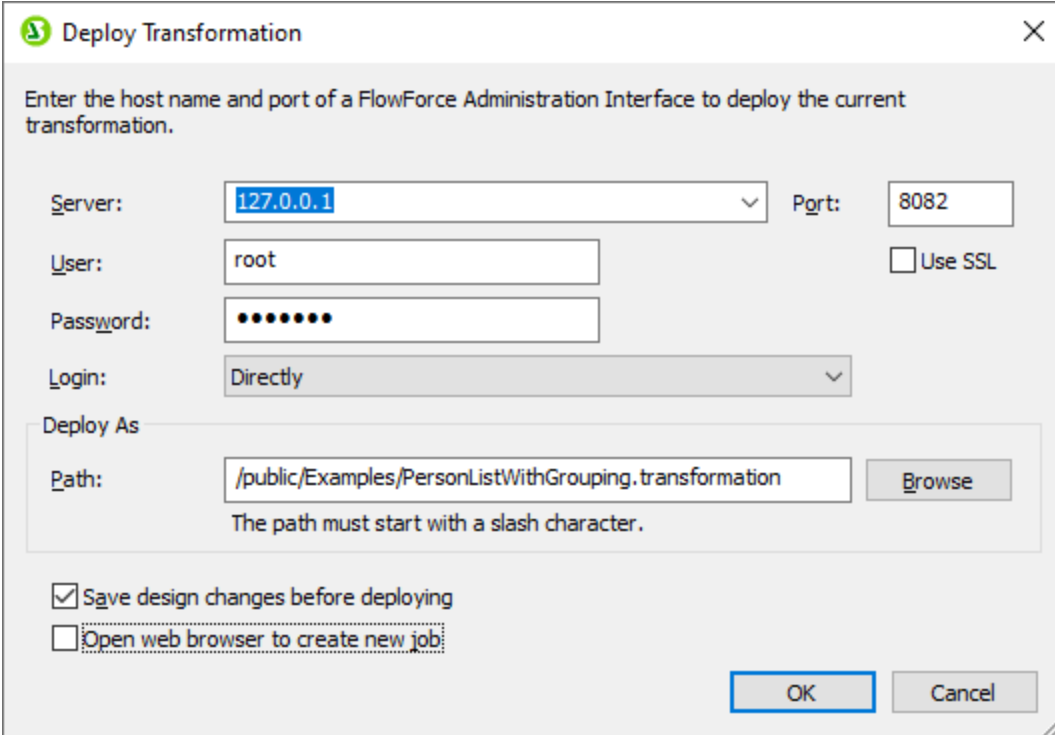
- When prompted to select the desired files to be included in the deployed package, leave the default settings as is. Although only PDF is generated in this example, including other outputs will save you time later if you change your mind and want to generate additional formats like HTML and RTF.



- When prompted, fill in the connection details to FlowForce Web Server. For simplicity, in the image below, the transformation is deployed to the local machine on port 8082, through plain HTTP. It is also possible to specify a remote address and deploy the files through an SSL-encrypted connection, provided that FlowForce Web Server has been configured to accept such connections, see [Defining the Network Settings](#)⁴⁷. The user and password values are illustrated below for the root FlowForce

account; however, any other FlowForce user account may also be used, provided that it has permissions to write data to the specified path. In this example, the **Open browser to create new job** check box has been deliberately left unchecked, because creating and configuring the job will be a separate step discussed further below.

5. For consistency with other examples, it is recommended to use the target path **/public/Examples/PersonListWithGrouping.transformation**.



Deploy Transformation

Enter the host name and port of a FlowForce Administration Interface to deploy the current transformation.

Server: 127.0.0.1 Port: 8082

User: root Use SSL

Password: ●●●●●●

Login: Directly

Deploy As

Path: /public/Examples/PersonListWithGrouping.transformation

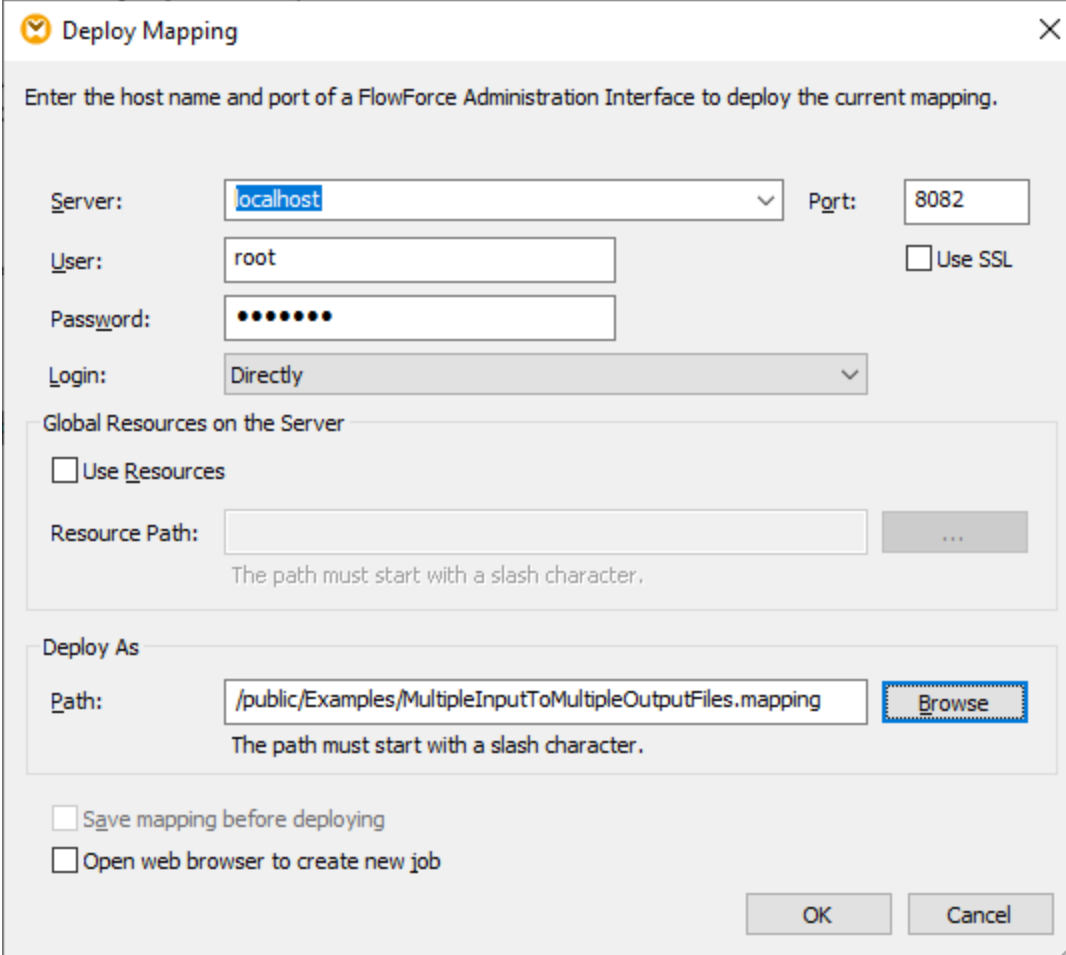
The path must start with a slash character.

Save design changes before deploying

Open web browser to create new job

To deploy the MapForce mapping to FlowForce Server:

- On the **File** menu, click **Deploy to FlowForce Server**. Filling in the connection details illustrated below works the same way as discussed above for StyleVision. Again, for consistency with other examples, it is recommended to use the target path **/public/Examples/MultipleInputToMultipleOutputFiles.mapping**.



Deploy Mapping

Enter the host name and port of a FlowForce Administration Interface to deploy the current mapping.

Server: localhost Port: 8082

User: root Use SSL

Password: ●●●●●●

Login: Directly

Global Resources on the Server

Use Resources

Resource Path: ...
The path must start with a slash character.

Deploy As

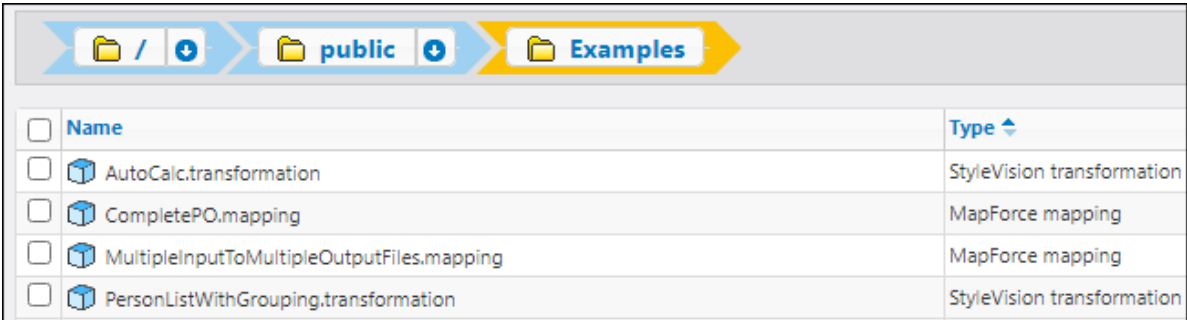
Path: /public/Examples/MultipleInputToMultipleOutputFiles.mapping
The path must start with a slash character.

Save mapping before deploying

Open web browser to create new job

OK Cancel

After the files were successfully deployed, the corresponding entries will appear in the specified FlowForce container (in this case, "/public/Examples") when you log on to FlowForce Server:



/ public Examples	
Name	Type
<input type="checkbox"/> AutoCalc.transformation	StyleVision transformation
<input type="checkbox"/> CompletePO.mapping	MapForce mapping
<input type="checkbox"/> MultipleInputToMultipleOutputFiles.mapping	MapForce mapping
<input type="checkbox"/> PersonListWithGrouping.transformation	StyleVision transformation

Note that the entries above are not jobs yet; they are now FlowForce functions from which actual jobs have yet to be created, as shown below.

Create and configure the FlowForce job

Now that the MapForce mapping and the StyleVision transformation have been deployed to FlowForce Server, they can be used to create the required job, as follows:

1. Navigate to the FlowForce **/public/Examples** container and click the function **MultipleInputToMultipleOutputFiles.mapping** deployed previously. Notice that the source component of the MapForce mapping discussed at the very beginning of this example has now become an input parameter to the FlowForce function. Also, it has a default value which is the path to the instance XML files processed by the mapping. This value can be overridden later if necessary. The "Working-directory" parameter was added automatically by FlowForce; its role will be clarified in the next steps.

Function MultipleInputToMultipleOutputFiles.mapping in /public/Examples

Function Input Parameters

Name: Type: Default:

Name: Type: Default:

The function will be executed with 'MapForce' version '2021'.

2. Click **Create Job**.
3. Enter a name and optionally a description for the job you are creating.
4. Configure the "Execution Steps" part of the job as shown below.

Execution Steps

- Execute function `/public/Examples/MultipleInputToMultipleOutputFiles.mapping`
 - Parameters:
 - Altova_Hierarchical: `Nanonull-*.xml`
 - Working-directory: `C:\FlowForceExamples\GeneratePdfs`
 - Assign this step's result to `output` as `PersonList`
- For each `item` in sequence `results(output)`
 - Execute function `/system/compute`
 - Parameters:
 - Expression: `as-file(item)`
 - Assign this step's result to `file` as `T0`
 - Execute function `/public/Examples/PersonListWithGrouping.transformation`
 - Parameters:
 - InputXml: `{file}`
 - OutHtml: `+`
 - GenerateHtmlOutputAsMime: `+`
 - OutRtf: `+`
 - OutFo: `+`
 - OutPdf: `{filename(file)}.pdf`
 - OutDocx: `+`
 - Working-directory: `C:\FlowForceExamples\GeneratePdfs`

5. Under "Service", select the "Make this job available via HTTP..." check box and enter the name of the Web service that will trigger the job on demand, for example "GeneratePdfsService". If you prefer to run the job as a scheduled job, or as a file system trigger, set the appropriate triggers (see [Managing Triggers](#) ¹⁵¹).

Service

Make this job available via HTTP at URL `http://<FlowForce server>/service/` `GenerateMultiplePDFs`

6. Under "Credential", enter the username and password of the operating system user account (the job will be executed as this user). Be careful not to confuse this password with the password of the FlowForce Web administration interface (see also [Credentials](#) ¹⁶²).

7. Click **Save**.

To understand how the job actually works, let's have a closer look at the "Execution Steps" section of the job. The first execution step calls the mapping deployed previously. It looks for any XML file that begins with "Nanonull-" in the working directory. In this example, the working directory is **C:**

\FlowForceExamples\GeneratePdfs.

The output returned by the first execution step represents the data returned by the mapping. It has been explicitly named **output**, in order to make it possible to refer to it in a subsequent step.

The second step of the job is a "for-each" step. Notice how the "for-each" step uses a FlowForce expression **results(output)** to get access to the data returned by the first step (that is, the output returned by the mapping). Specifically, the expression calls the function **results()** which takes as argument the output returned by the previous step, see also [Step Result Functions](#)²⁰². For an introduction to FlowForce expressions, see [FlowForce Expressions](#)¹⁷¹.

The "for-each" step consists of two smaller execution steps:


1. The first step calls the [/system/compute](#)²³⁷ built-in function to convert the mapping output into an actual file (generically named **file**). Importantly, the output of the **results(output)** expression is a stream, not a file. If the mapping returns multiple outputs (as in this case), the mapping output represents a sequence of streams. For this reason, a FlowForce expression function (**as-file**) is used to convert the current stream (the one that is being iterated) into an actual file.
2. The second step calls, for each stream that is being iterated, the StyleVision transformation deployed previously. Namely, with each iteration, StyleVision Server is called, an XML file is supplied as input, and a PDF file is returned as output. The FlowForce expression **{filename(file)}.pdf** creates the actual PDF file name on the disk. This operation takes place in the working directory specified by the "Working-directory" parameter.

Note: In this example, the same working directory is used by both steps—the one which executes the mapping and the one which runs the StyleVision transformation. In some cases, it may be necessary to specify separate working directories, in order to avoid file name collision or job execution errors.

Running the job

To prepare the input data for the job, copy the **Nanonull-Branch.xml** and **Nanonull-HQ.xml** from **<Documents>\Altova\MapForce2024\MapForceExamples** to the working directory (**C:\FlowForceExamples\GeneratePdfs**). This way, the first step of the job gets some input XML files to read data from when the job runs.

To run the job, do one of the following:

- Go to **Home**, and then click **Show all active triggers and services**. Next, click the job's URL displayed in the "Info" column.
- Enter <http://127.0.0.1:4646/service/GeneratePdfsService> in the browser's address bar. Note that this URL works only if the *FlowForce Server* service listens at the default host address and port name. If you have defined other host and port settings in the [Configuration page](#)⁴⁷, change the address accordingly.
- If you set the optional **Host name** field of FlowForce Server from the [Setup Page](#)⁴⁷, you can execute the web service call directly from the job configuration page, by clicking the  button adjacent to the **Make this job available via HTTP** check box. The button is not displayed otherwise.

If prompted for credentials when accessing the Web service, supply the same credentials you use to log on to FlowForce Server.

Supplying your FlowForce Server user credentials for HTTP authentication is only for testing purposes. For production, it is recommended that you create a new FlowForce user, grant the **Service - Use** permission to this user on the container where the job is, and then access the Web service with the corresponding user account. To disable HTTP authentication and make the Web service public, grant the **Service - Use** permission to the user **Anonymous**, see [How Permissions Work](#)⁹².

On successful job execution, the PDF files generated by the job will be available in the working directory **C:\FlowForceExamples\GeneratePdfs**. The browser displays "Cannot output the job" even in case of successful execution (this is expected, since the job produces PDF files which cannot be output to the browser). If the job fails to execute for any reason, the browser will display a "Service execution failed" message. In this case, check the [error log](#)¹⁰⁴ of the job. To troubleshoot issues, you may need to verify again all the prerequisites listed at the top of this page.

6 Command Line

The FlowForce solution consists of two services: FlowForce Server and FlowForce Web Server. For each of these services, an executable is available that supports administrative commands that can be run at the command line. You can find both executables at the following path:

Linux	/opt/Altova/FlowForceServer2024/bin/
macOS	/usr/local/Altova/FlowForceServer2024/bin/
Windows	<ProgramFilesFolder>\Altova\FlowForceServer\bin\

The executable names are as follows:


Linux	flowforceserver flowforcewebserver
macOS	flowforceserver flowforcewebserver
Windows	FlowForceServer.exe FlowForceWebServer.exe

Available commands

The command line interface (CLI) can be used for administration purposes (such as licensing, troubleshooting, and internal database backup). The commands supported by the CLI are listed below. The abbreviations *FFS* and *FFW* in the table below stand for FlowForce Server and FlowForce Web Server, respectively.

Note that before using some of the commands, you must shut down the FlowForce Server and FlowForce Web Server services (*see details below*).

Command	FFS	FFW	Services must be shut down	Description
help ³⁸⁷	Yes	Yes		Displays help for the command supplied as argument.
assignlicense ³⁸⁸	Yes			This command is applicable to Windows platforms only. It can be used to upload and assign a license file to FlowForce Server.
compactdb ³⁸⁹	Yes		Yes	Reduces the size of FlowForce .db files if they contain deleted records.
createdb ³⁹⁰	Yes		Yes	Creates a new FlowForce database.
debug ³⁹¹	Yes	Yes	Yes	Starts the application in debug mode.

exportresourcestrings ³⁹²	Yes	Yes		Exports all application resource strings to an XML file
foreground ³⁹³	Yes	Yes	Yes	Starts the application in foreground mode.
initdb ³⁹⁴	Yes		Yes	Creates or updates the FlowForce database.
install ³⁹⁵	Yes	Yes		Installs the application as a Windows service.
licenseserver ³⁹⁶	Yes			Registers FlowForce Server with the Altova LicenseServer on the local network.
migratedb ³⁹⁷	Yes		Yes	Migrates FlowForce Server data from a previous version to the latest version.
repair ³⁹⁸	Yes		Yes	Starts the application in repair mode.
resetpassword ³⁹⁹	Yes			Resets the password of the  root user to the default value, and grants to the  root user all privileges.
setdeflang sdl ⁴⁰⁰	Yes	Yes		Sets the default language.
start ⁴⁰¹	Yes	Yes		Starts the application as a service.
uninstall ⁴⁰²	Yes	Yes		Uninstalls the application as a Windows service.
upgradedb ⁴⁰³	Yes		Yes	Upgrades the FlowForce Server database to the latest version.
verifylicense ⁴⁰⁴	Yes			This command is applicable to Windows platforms only. It can be used to verify whether FlowForce Server is licensed, and, optionally, whether a given license key is already assigned to FlowForce Server.

Conventions

By convention, this documentation omits the full path of the executable when describing a given command, and uses `flowforceserver` instead of the executable name, for example:

```
flowforceserver help
```

Where `flowforceserver` is the path or name of the executable. Note that, if you use an absolute path, you will be able to run commands regardless of the current directory that your command prompt window (terminal) is in. However, if you would like to call the executable just by typing its name, make sure to do one of the following first:

- Change the terminal's current directory to the FlowForce Server installation directory
- Add the directory where the executable is to the PATH environment variable.

Both of these scenarios are described in more detail below.

Tips and tricks

If you are new to command line, be aware of the following tips and tricks.

- To find out the current directory where you command line window is, enter `pwd` on Linux and macOS. On Windows, enter `echo %CD%`.
- Make use of the **Tab** key to quickly enter various file or directory paths without having to type them in full. For example, if you type `cd c:\prog` at the command line, and then press **Tab**, you will get `C:\Program Files` automatically pre-filled (or perhaps some other directory under C:\ whose name begins with "Prog").
- When entering paths that contain white space, such as `C:\Program Files` on Windows, enclose them within quotes.
- If you see a message similar to "This command is not recognized as an internal or external command, operable program or batch file", you have most likely mistyped a path or command.
- On Linux, make sure that you use the correct case for file or directory names. For example, typing a path such as `/home/nikita/downloads` will return an error if the directory name is actually `/home/nikita/Downloads`.
- When typing a path on Linux or macOS, use forward slashes, as opposed to back slashes on Windows.

How to run a command

1. Open a command prompt window.
 - a. To open a command prompt on Windows, press the **Windows** key and then start typing `cmd`. Click the **Command Prompt** suggestion that appears.
 - b. To open a terminal on Mac, click the **Finder** icon, and then select **Go > Utilities** from the menu. Double-click the **Terminal** icon in the Utilities window.
 - c. If you run Linux from a graphical user interface, locate and run the **Terminal** command as applicable to your Linux distribution. If you run Linux from a command line interface, ignore this step.
2. Enter the full path to the executable, followed by the command you want to run. For example, the command below provides help at the command line.

<i>Linux</i>	<code>/opt/Altova/FlowForceServer2024/bin/flowforceserver help</code>
<i>macOS</i>	<code>/usr/local/Altova/FlowForceServer2024/bin/flowforceserver help</code>
<i>Windows</i>	<code>C:\Program Files (x86)\Altova\FlowForceServer2024\bin\FlowForceServer.exe help</code>

In the example above, the command `help` was run without any options or arguments. Other commands may have arguments and options, and those arguments and options could be mandatory or optional. Check the reference section for details about each command.

Calling FlowForce Server in the installation directory

To call the executable without having to type the full path, change the current directory to the directory where the FlowForce Server executable was installed, for example:

<i>Linux</i>	<code>cd /opt/Altova/FlowForceServer2024/bin</code>
<i>macOS</i>	<code>cd /usr/local/Altova/FlowForceServer2024/bin</code>
<i>Windows</i>	<code>cd C:\Program Files (x86)\Altova\FlowForceServer2024\bin</code>

You can now run any command by typing just the executable name, for example:

<i>Linux</i>	<code>./flowforceserver help</code>
<i>macOS</i>	<code>./flowforceserver help</code>
<i>Windows</i>	<code>FlowForceServer.exe help</code>

Note: On Linux and macOS systems, the prefix `./` indicates that the executable is in the current directory.

Calling FlowForce Server from any directory

To call the executable from any directory, refer to it using the absolute path. Alternatively, if you want to call the program by typing just the executable name, you can edit the PATH environment variable of your operating system so that it includes the full path to the FlowForce Server installation directory. For ways to change the PATH environment variable, refer to the documentation of your operating system.

Note: After changing the PATH environment variable, you may need to close the terminal window and open a new one, in order for the changes to take effect.

6.1 help

Purpose

Provides help information about the command supplied as an argument.

Syntax

```
FlowForceServer help <command>
```

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

Arguments

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.

Using `--help` as option for other commands

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

```
FlowForceServer createdb --help
```

has the same result as:

```
FlowForceServer help createdb
```

6.2 assignlicense

Purpose

This command can be used to upload and assign a license file to FlowForce Server.

Syntax

```
FlowForceServer assignlicense [options] FILE
```

Arguments

FILE	Specifies the path of the license file to be uploaded.
------	--

Options

<code>--t, --test-only=true false</code>	<p>When set to <code>true</code>, the license is uploaded and validated.</p> <p>When set to <code>false</code>, the license is uploaded, validated, and assigned as well.</p> <p>If this option is not specified, the default value is <code>true</code>.</p>
--	---

6.3 compactdb

Purpose

Reduces the size of FlowForce `.db` files if they contain deleted records. This command is particularly useful after running the [archive-log](#)²⁸⁴ or [truncate-log](#)²⁸⁵ system maintenance functions.

You can also compact the database files through the [FlowForce Server Setup Page](#)⁴⁵.

Before compacting the `.db` files, you must [stop](#)⁶¹ the FlowForce Server and FlowForce Web Server services.

Syntax

```
FlowForceServer compactdb [options]
```

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

Options

```
--datadir=VALUE
```

VALUE is the path of the [instance-data directory](#)¹⁸ which contains the `.db` files to be compacted. If this option is not specified, the `/data` directory will be used by default.

6.4 createdb

Purpose

Creates a new database. If the database already exists, the command will fail.

Syntax

```
FlowForceServer createdb [options]
```

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

Options

```
--          VALUE is the path of the instance-data directory18.  
datadir=VA  
LUE
```

6.5 debug

Purpose

Not for general use. This command starts FlowForce Server in debug mode (that is, not as a service). To stop this mode, press **CTRL+C**.

Syntax

```
FlowForceServer debug [options]
```

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

Options

```
--          VALUE is the path of the instance-data directory18.  
datadir=VA  
LUE
```

6.6 exportresourcestrings

Purpose

Outputs an XML file containing the resource strings of FlowForce Server. 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. Valid export languages (with their language codes in parentheses) are: English (en), German, (de), Spanish (es), and Japanese (ja).

Syntax

```
FlowForceServer exportresourcestrings Language XMLOutput
```

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

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

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

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


6.7 foreground

Purpose

Not for general use. This command starts Altova FlowForce Server in the foreground. It is used internally by the startup scripts for Linux.

Syntax

```
FlowForceServer foreground [options]
```

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

Options

```
--          VALUE is the path of the instance-data directory18.  
datadir=VA  
LUE
```

6.8 initdb

Purpose

Creates a new database or updates an existing one to the latest version.

Syntax

```
FlowForceServer initdb [options]
```

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

Options

```
--          VALUE is the path of the instance-data directory18.  
datadir=VA  
LUE
```

6.9 install

Purpose

If you use the [Setup page](#)⁴⁵ for installing services, you do not need the `install` command. If you configure your server instance via [the configuration files and CLI](#)⁶⁴, use this command to install FlowForce Server and FlowForce Web Server as services. Note that installing both services is mandatory for successful server configuration.

Syntax

Use the command below to install FlowForce Server as a service:

```
FlowForceServer install [options]
```

Use the command below to install FlowForce Web Server as a service:

```
FlowForceWebServer install [options]
```

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

Options

```
--          VALUE is the path of the instance-data directory18.  
datadir=VA  
LUE
```

6.10 licenseserver

Purpose

Registers FlowForceServer with LicenseServer. You must have Administrator privileges (root) to register FlowForce Server with LicenseServer. For more information, see the [LicenseServer documentation](#).

Syntax

```
FlowForceServer licenseserver [options] SERVER
```

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

Arguments

SERVER	Specifies the name of the machine running LicenseServer or its IP address.
--------	--

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.

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

Example

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

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

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

6.11 migratedb

Purpose

Copies FlowForce Server data from a previous [instance-data directory](#) ¹⁸ to the current one, and also upgrades the FlowForce database to the latest version if necessary. This command is invoked by the FlowForce installation scripts when there is already a previous version of FlowForce Server installed, so you do not typically need to run it. Running this command may be useful when you migrate FlowForce Server to a new machine or when you restore the instance-data directory from a backup (see [Backup and Recovery](#) ⁷⁵).

If you only need to upgrade the FlowForce database version to the latest one, it is sufficient to run [upgradedb](#) ⁴⁰³.

You can also migrate data through the [FlowForce Server Setup Page](#) ⁴⁵.

Before using the `migratedb` command, you must [stop](#) ⁶¹ the FlowForce Server and FlowForce Web Server services.

Syntax

```
FlowForceServer migratedb [options] --olddatadir=VALUE
```

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

Options

<code>--datadir=VALUE</code>	VALUE is the path of the instance-data directory
<code>--olddatadir=VALUE</code>	VALUE is the old path of the instance-data directory

Example

To migrate data from the application data directory of FlowForce Server 2022 to FlowForce Server 2024, run:

```
"C:\Program Files(x86)\Altova\FlowForceServer2024\bin\FlowForceServer.exe" migratedb
--datadir=C:\ProgramData\Altova\FlowForceServer2024\data --olddatadir=C:\
\ProgramData\Altova\FlowForceServer2022\data
```

6.12 repair

Purpose

Starts FlowForce Server with all triggers and job execution processes disabled, to enable troubleshooting.

Syntax

```
FlowForceServer repair [options]
```

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

Options



```
--          VALUE is the path of the instance-data directory18.  
datadir=VA  
LUE
```

Example

```
FlowForceServer repair --datadir=C:\ProgramData\Altova\FlowForceServer2024\data
```

6.13 resetpassword

Purpose

Resets the password of the  **root** user to the default value, and grants to the  **root** user all privileges. It is recommended to stop the running instance of FlowForce Server before performing this operation (see instructions for starting or stopping services on Linux, macOS, and Windows).

You can also reset the password through the [FlowForce Server Setup Page](#)⁴⁵.

Syntax

```
FlowForceServer resetpassword [options]
```

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

Options

```
--          VALUE is the path of the instance-data directory18.  
datadir=VA  
LUE
```

Example

```
FlowForceServer resetpassword --datadir=C:\ProgramData\Altova\FlowForceServer\data
```

6.14 setdeflang (sdl)

Purpose

The `setdeflang` command (short form is `sdl`) sets the default language of FlowForce Server. To change the default language, run this command for both `FlowForceServer` and `FlowForceWebServer` services (see *Syntax*).

Syntax

```
FlowForceServer setdeflang | sdl LanguageCode
FlowForceWebServer setdeflang | sdl LanguageCode
```

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

The possible values of *LanguageCode* are as follows.

en	English
es	Spanish
de	German
fr	French
ja	Japanese

Example

```
FlowForceServer setdeflang de
```


6.15 start

Purpose

Starts FlowForce Server as a service. This command is used internally by the startup scripts or by the Windows service installation; it is not for general use.

Syntax

```
FlowForceServer start [options]
```

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

Options

-- VALUE is the path of the [instance-data directory](#)¹⁸.
datadir=VA
LUE

6.16 uninstall

Purpose

This command uninstalls the FlowForce Server and FlowForce Web Server services. You can also use the [Setup page](#)⁴⁵ for uninstalling the services.

Syntax

To uninstall the FlowForce Server service, use the following command:

```
FlowForceServer uninstall
```

To uninstall the FlowForce Web Server service, use the following command:

```
FlowForceWebServer uninstall
```

6.17 upgradedb

Purpose

Upgrades the database to the latest version. The default database is upgraded automatically at installation time; therefore, it is usually not necessary to run this command manually. You can also upgrade the database through the [FlowForce Server Setup Page](#) ⁴⁵.

Before upgrading the database, you must [stop](#) ⁶¹ the FlowForce Server and FlowForce Web Server services.

Syntax

```
FlowForceServer upgradedb [options]
```

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

Options

```
--          VALUE is the path of the instance-data directory 18.  
datadir=VA  
LUE
```

Example

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

6.18 verifylicense

Purpose

This command can be used to verify whether FlowForce Server is licensed, and, optionally, whether a given license key is already assigned to FlowForce Server.

Syntax

```
FlowForceServer verifylicense [options]
```

Options

<code>--l, --license-key=VALUE</code>	<p>This option enables you to verify if a particular license key is already assigned to FlowForce Server.</p> <p>The value must be set to the license key that you wish to verify.</p>
---------------------------------------	--

7 Integration with Altova Products

In [How It Works](#)¹², you have seen an overview of Altova products working together. Essentially, mapping files created with Altova MapForce and transformation files created with Altova StyleVision can be automated with the help of the following server counterpart products: MapForce Server (or MapForce Server Advanced Edition) and StyleVision Server. In addition, functions available in RaptorXML Server can also be invoked from FlowForce Server jobs, if the latter runs under FlowForce Server management.

MapForce Server and StyleVision Server can run mappings and transformations across multiple platforms (Windows, macOS, Linux), either at the command line, or from an API call. If these products do not run alongside FlowForce, automation entails developing programs or writing scripts which call the API or invoke the command line of MapForce Server or StyleVision Server.

When MapForce Server and StyleVision Server run under FlowForce Server management, automation can be taken to the next level. Namely, you can deploy the mappings and transformations directly to FlowForce Server and run them as jobs. This way, the mapping or transformation will benefit from all the advantages of a FlowForce Server job: scheduled or on demand execution, execution as a Web service, configuration by means of FlowForce expressions, error handling, conditional processing, email notifications, and so on.

Once deployed to FlowForce Server, the mapping or transformation appears in the container to which you deployed it. As illustrated below, mappings have the **.mapping** extension while transformations have the **.transformation** extension.



<input type="checkbox"/>	Name	Type
<input type="checkbox"/>	AutoCalc.transformation	StyleVision transformation
<input type="checkbox"/>	CompletePO.mapping	MapForce mapping

From a FlowForce perspective, such objects are actually functions, and thus can be turned into new jobs. They can also be called from existing jobs, and accept various inputs (typically, files) as parameters. Note that FlowForce Server does not execute such mapping or transformation functions by itself; MapForce Server or StyleVision Server (or both, depending on the case) are invoked to perform the actual execution.

The RaptorXML functions are available in the RaptorXML container, see also [Integration with RaptorXML Server](#)⁴³³.

The next sections discuss how to prepare mappings and transformations for server execution, how to turn them into jobs and how to process their results in FlowForce Server.

7.1 Prepare Files for Server Execution

A mapping designed and previewed with MapForce may refer to resources which are outside of the current machine and operating system (such as databases). In addition to this, in MapForce, all mapping paths follow Windows-style conventions by default. Thirdly, the machine where MapForce Server runs might not support the same database connections as the machine where the mapping was designed. For this reason, running mappings in a server environment typically requires some preparation, especially if the target machine is not the same as the source machine.

Note: The term "source machine" refers to the computer where the MapForce is installed and the term "target machine" refers to the computer where MapForce Server or FlowForce Server is installed. In the most simple scenario, this is the same computer. In a more advanced scenario, MapForce runs on a Windows machine whereas MapForce Server or FlowForce Server runs on a Linux or macOS machine.

As best practice, always make sure that the mapping validates successfully in MapForce before deploying it to FlowForce Server or compiling it to a MapForce Server execution file.

If MapForce Server runs standalone (without FlowForce Server), the required licenses are as follows:

- On the source machine, MapForce Enterprise or Professional edition is required to design the mapping and compile it to a server execution file (.mfx).
- On the target machine, MapForce Server or MapForce Server Advanced Edition is required to run the mapping.

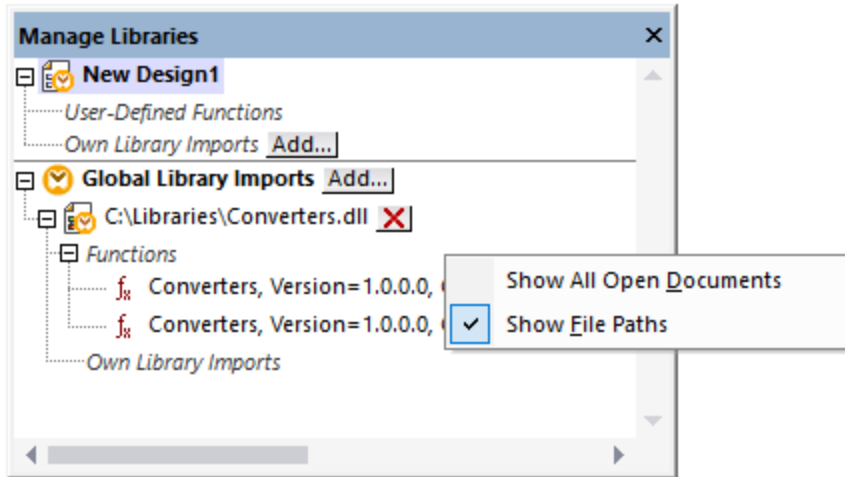
If MapForce Server runs under FlowForce Server management, the following requirements apply:

- On the source machine, MapForce Enterprise or Professional edition is required to design the mapping and deploy it to a target machine.
- Both MapForce Server and FlowForce Server must be licensed on the target machine. The role of MapForce Server is to run the mapping; the role of FlowForce is to make the mapping available as a job which benefits from features such as scheduled or on demand execution, execution as a Web service, error handling, conditional processing, email notifications, and others.
- FlowForce Server must be up and running at the configured network address and port. Namely, the "FlowForce Web Server" service must be started and configured to accept connections from HTTP clients (or HTTPS if configured) and must not be blocked by the firewall. The "FlowForce Server" service must also be started and running at the designated address and port.
- You have a FlowForce Server user account with permissions to one of the containers (by default, the **/public** container is accessible to any authenticated user).

General considerations

- If you intend to run the mapping on a target machine with standalone MapForce Server, all input files referenced by the mapping must be copied to the target machine as well. If MapForce Server runs under FlowForce Server management, there is no need to copy files manually. In this case, the instance and schema files are included in the package deployed to the target machine.
- If the mapping includes database components which require specific database drivers, such drivers must be installed on the target machine as well. For example, if your mapping reads data from a Microsoft Access database, then Microsoft Access or Microsoft Access Runtime (<https://www.microsoft.com/en-us/download/details.aspx?id=50040>) must be installed on the target machine as well.

- When you deploy a mapping to non-Windows platforms, ADO, ADO.NET and ODBC database connections are automatically changed to JDBC. Native SQLite and native PostgreSQL connections are preserved as such and require no additional configuration. See also "Database connections" below.
- If the mapping contains custom function calls (for example, to .dll or .class files), such dependencies are not deployed together with the mapping, since they are not known before runtime. In this case, copy them manually to the target machine. The path of the .dll or .class file on the server must be the same as in the "Manage Libraries" window in MapForce, for example:



- Some mappings read multiple input files using a wildcard path. In this case, the input file names are not known before runtime and so they are not deployed. For the mapping to execute successfully, the input files must exist on the target machine.
- If the mapping output path includes directories, those directories must exist on the target machine. Otherwise, an error will be generated when you execute the mapping. This behavior is unlike MapForce, where non-existing directories are generated automatically if the option **Generate output to temporary files** is enabled.
- If the mapping calls a Web service that requires HTTPS authentication with a client certificate, the certificate must be transferred to the target machine as well.
- If the mapping connects to file-based databases such as Microsoft Access and SQLite, the database file must be manually transferred to the target machine or saved to a shared directory which is accessible to both the source and the target machine and referenced from there, see "File-based databases" below.

Making paths portable

If you intend to run the mapping on a server, ensure that the mapping follows the applicable path conventions and uses a supported database connection.

To make paths portable to non-Windows operating systems, use relative instead of absolute paths when designing the mapping in MapForce:

1. Open the desired mapping design file (.mfd) with MapForce on Windows.
2. On the **File** menu, select **Mapping Settings**, and clear the **Make paths absolute in generated code** check box if it is selected.
3. For each mapping component, open the **Properties** dialog box (by double-clicking the component's title bar, for example), and change all file paths from absolute to relative. Also, select the **Save all file paths relative to MFD file** check box. For convenience, you can copy all input files and schemas into the same folder as the mapping itself, and reference them just by the file name.

For more information about dealing with relative and absolute paths while designing mappings, refer to MapForce documentation.

Importantly, both MapForce Server and FlowForce Server support a so-called "working directory" against which all relative paths will be resolved. The working directory is specified at mapping runtime, as follows:

- In FlowForce Server, by editing the "Working-directory" parameter of any job.
- In MapForce Server API, through the `WorkingDirectory` property of the COM and .NET API, or through the `setWorkingDirectory` method of the Java API.
- In MapForce Server command line, the working directory is the current directory of the command shell.

Database connections

Be aware that ADO, ADO.NET, and ODBC connections are not supported on Linux and macOS machines. Therefore, if the target machine is Linux or macOS, such connections are converted to JDBC when you deploy the mapping to FlowForce or when you compile the mapping to a MapForce Server execution file. In this case, you have the following options before deploying the mapping or compiling it to a server execution file:

- In MapForce, create a JDBC connection to the database
- In MapForce, fill the JDBC database connection details in the "JDBC-specific Settings" section of the database component.

If the mapping uses a native connection to a PostgreSQL or SQLite database, the native connection is preserved and no JDBC conversion takes place. If the mapping connects to a file-based database, such as Microsoft Access and SQLite, additional configuration is required, see "File-based databases" below.

Running mappings with JDBC connections requires that the Java Runtime Environment or Java Development Kit be installed on the server machine. This may be either Oracle JDK or an open source build such as Oracle OpenJDK.

- The `JAVA_HOME` environment variable must point to the JDK installation directory.
- On Windows, a Java Virtual Machine path found in the Windows registry will take priority over the `JAVA_HOME` variable.
- The JDK platform (64-bit, 32-bit) must be the same as that of MapForce Server. Otherwise, you may get an error with the reason: "JVM is inaccessible".

To set up a JDBC connection on Linux or macOS:

1. Download the JDBC driver supplied by the database vendor and install it on the operating system. Make sure to select the 32-bit version if your operating system runs on 32-bit, and the 64-bit version if your operating system runs on 64-bit.
2. Set the environment variables to the location where the JDBC driver is installed. Typically, you will need to set the `CLASSPATH` variable, and possibly a few others. To find out which specific environment variables must be configured, check the documentation supplied with the JDBC driver.

Note: On macOS, the system expects any installed JDBC libraries to be in the `/Library/Java/Extensions` directory. Therefore, it is recommended that you unpack the JDBC driver to this location; otherwise, you will need to configure the system to look for the JDBC library at the path where you installed the JDBC driver.

Oracle Instant Client connections on macOS

These instructions are applicable if you connect to an Oracle database through the **Oracle Database Instant Client**, on macOS. Prerequisites:

- Java 8.0 or later must be installed. If the Mac machine runs a Java version prior to Java 8, you can also connect through the **JDBC Thin for All Platforms** library, and disregard the instructions below.
- Oracle Instant Client must be installed. You can download the Oracle Instant Client from the Oracle official download page. Note that there are several Instant Client packages available on the Oracle download page. Make sure to select a package with Oracle Call Interface (OCI) support, (for example, Instant Client Basic). Also, make sure to select the 32-bit version if your operating system runs on 32-bit, and the 64-bit version if your operating system runs on 64-bit.

Once you have downloaded and unpacked the Oracle Instant Client, edit the property list (.plist) file shipped with the installer so that the following environment variables point to the location of the corresponding driver paths, for example:

Variable	Sample Value
CLASSPATH	/opt/oracle/instantclient_11_2/ojdbc6.jar:/opt/oracle/instantclient_11_2/ojdbc5.jar
TNS_ADMIN	/opt/oracle/NETWORK_ADMIN
ORACLE_HOME	/opt/oracle/instantclient_11_2
DYLD_LIBRARY_PATH	/opt/oracle/instantclient_11_2
PATH	\$PATH:/opt/oracle/instantclient_11_2

Note: Edit the sample values above to fit the paths where Oracle Instant Client files are installed on your operating system.

File-based databases

File-based databases such as Microsoft Access and SQLite are not included in the package deployed to FlowForce Server or in the compiled MapForce Server execution file. Therefore, if the source and target machine are not the same, take the following steps:

1. In MapForce, right-click the mapping and clear the check box **Make paths absolute in generated code**.
2. Right-click the database component on the mapping and add a connection to the database file using a relative path. A simple way to avoid path-related issues is to save the mapping design (.mfd file) in the same directory as the database file and to refer to the latter from the mapping just by file name (thus using a relative path).
3. Copy the database file to a directory on the target machine (let's call it "working directory"). Keep this directory in mind since it will be required to run the mapping on the server, as shown below.

To run such mappings on the server, do one of the following:

- If the mapping will be run by MapForce Server under FlowForce Server control, configure the FlowForce Server job to point to the working directory created previously. The database file must reside in the

working directory. For an example, see [Exposing a Job as a Web Service](#)¹⁵⁸.

- If the mapping will be run by standalone MapForce Server at the command line, change the current directory to the working directory (for example, `cd path\to\working\directory`) before calling the `run` command of MapForce Server.
- If the mapping will be run by the MapForce Server API, set the working directory programmatically before running the mapping. To facilitate this, the property `WorkingDirectory` is available for the MapForce Server object in the COM and .NET API. In the Java API, the method `setWorkingDirectory` is available.

If both the source and the target machines are Windows machines running on the local network, an alternative approach is to configure the mapping to read the database file from a common shared directory, as follows:

1. Store the database file in a common shared directory which is accessible by both the source and the target machine.
2. Right-click the database component on the mapping and add a connection to the database file using an absolute path.

Global Resources

If a mapping includes references to Global Resources instead of direct paths or database connections, you will be able to use Global Resources on the server side as well. When you compile a mapping to a MapForce Server execution file (.mfx), the references to Global Resources will be kept intact, so that you can provide these on the server side, at mapping runtime. When deploying a mapping to FlowForce Server, you can optionally choose whether it should use resources on the server.

For mappings (or mapping functions, in case of FlowForce Server) to run successfully, the actual file, folder, or database connection details that you supply as Global Resources must be compatible with the server environment. For example, files and folders paths must use the Linux convention for paths if the mapping will run on a Linux server. Likewise, Global Resources defined as database connections must be possible on the server machine.

For further information, see [Resources](#)⁴²³.

XBRL Taxonomy Packages

When you deploy a mapping that references XBRL Taxonomy Packages to FlowForce Server, MapForce collects all external references from the mapping and then resolves them using the current configuration and currently installed taxonomy packages. If there are resolved external references that point to a taxonomy package, then the taxonomy package is deployed together with the mapping. FlowForce Server will use that package—as it was during deployment—to execute the mapping. To refresh the taxonomy package used by FlowForce Server, you will need to change it in MapForce and redeploy the mapping.

Note that the root catalog of MapForce Server influences the way taxonomies are resolved on the target machine. The root catalog is found at the following path relative to the MapForce Server installation directory: **etc/RootCatalog.xml**.

Taxonomy packages that were deployed with a mapping will be used if the root catalog of MapForce Server does not already contain such a package or does not contain a package that is defined for the same URL prefix. The root catalog of MapForce Server has priority over the deployed taxonomy.

If MapForce Server runs standalone (without FlowForce Server), it is possible to specify the root catalog that should be used by the mapping as follows:

- At the command line, this is possible by adding the option `-catalog` to the `run` command.
- In the MapForce Server API, call the method `SetOption`, and supply the string `"catalog"` as first argument, and the path to the root catalog as second argument.

If a mapping uses XBRL components with table linkbases, the taxonomy package or the taxonomy package configuration file must be supplied to the mapping at runtime, as follows:

- At the MapForce Server command line, add the option `--taxonomy-package` or `--taxonomy-packages-config-file` to the `run` command.
- In the MapForce Server API, call the method `SetOption`. The first argument must be either `"taxonomy-package"` or `"taxonomy-packages-config-file"`. The second argument must be the actual path to the taxonomy package (or taxonomy package configuration) file.

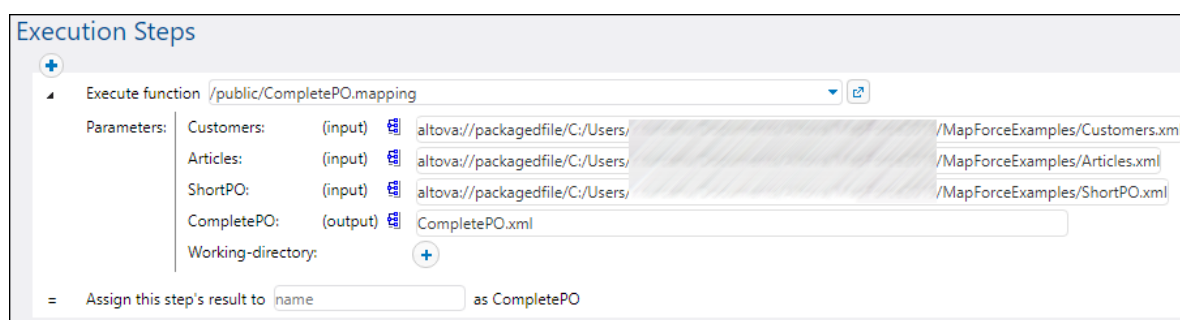
7.2 Deploy Mappings to FlowForce Server

Deploying a mapping to FlowForce Server means that MapForce organizes the resources used by the specific mapping into an object and passes it through HTTP (or HTTPS if configured) to the machine where FlowForce Server runs. MapForce mappings are typically deployed to FlowForce Server in order to automate their execution by means of FlowForce Server jobs. Once a mapping is deployed, you can create a full-featured FlowForce Server job from it, and benefit from all job-specific functionality (for example, define custom triggering conditions for the job, expose it as a Web service, and so on).

Note: The term "source machine" refers to the computer where the MapForce is installed and the term "target machine" refers to the computer where FlowForce Server is installed. In the most simple scenario, this is the same computer. In a more advanced scenario, MapForce runs on a Windows machine whereas FlowForce Server runs on a Linux or macOS machine.

The package deployed to FlowForce includes the following:

- The mapping itself. After deployment, the mapping becomes available in the FlowForce Server administration interface as a mapping function (.mapping), at the path you specify. Any source components become input arguments, and any target components become output arguments of this function.



- All kinds of input instance files (XML, CSV, Text) that are used by the mapping.

Prerequisites

See [Preparing Mappings for Server Execution](#) ⁴⁰⁶.

Deploying the mapping to FlowForce Server

1. Run MapForce and ensure that the transformation language is set to Built-In.
2. In the **File** menu, click **Deploy to FlowForce Server**. The **Deploy Mapping** dialog box opens (see *below*).

Deploy Mapping

Enter the host name and port of a FlowForce Administration Interface to deploy the current mapping.

Server: localhost Port: 8082

User: root Use SSL

Password: ●●●●

Login: <Default>

Global Resources on the Server

Use Resources

Resource Path: ...

The path must start with a slash character.

Deploy As

Path: /public/ChainedPersonList.mapping Browse

The path must start with a slash character.

Save mapping before deploying

Attach MFD file(s) for later retrieval

Open web browser to create new job

OK Cancel

3. Enter your deployment settings (as described below) and click OK. If you select the *Open web browser to create new job* check box, the FlowForce Server administration interface opens in the browser, and you can start creating a FlowForce Server job immediately.

The table below lists the mapping-deployment settings available in the **Deploy Mapping** dialog box.

Setting	Description
Server, Port, Use SSL	<p>Enter the server host name (or IP address) and port of FlowForce Server. These could be localhost and 8082 if FlowForce Server is running on the same machine at the default port. When in doubt, log on to FlowForce Server Web administration interface and check the I.P. address and port displayed in the Web browser's address bar.</p> <p>If you encounter connectivity errors, ensure that the machine on which FlowForce Server runs is configured to allow incoming connections on the designated address and port.</p>

Setting	Description
	To deploy the mapping through a SSL-encrypted connection, select the Use SSL check box. This assumes that FlowForce Server is already configured to accept SSL connections. For more information, refer to FlowForce Server documentation (https://www.altova.com/documentation).
User and Password	The user name and password to be entered depends on the value of the Login drop-down list (see next option). If the Login drop-down list is set to <Default> or Directly , enter your FlowForce Server user name and password. Otherwise, enter your domain user name and password, and select the domain name from the Login drop-down list.
Login	If Directory Service integration is enabled in FlowForce Server, select the domain name from this drop-down list, and enter your domain credentials in the User and Password fields (see previous option).
Use Resources, Resource Path	<p>Select the Use Resources check box if the mapping function should use Resources ⁴²³ after it is deployed to the server. If you select the check box, you must also enter the path of the respective resource on the server in the Resource Path text box. To select the resource, click the Ellipsis button.</p> <p>If there are no resources on the server yet to choose from, click Deploy Global Resources and deploy the required Global Resource to the server.</p> <p>If you do not select the Use Resources check box, any Global Resources will be resolved, based on the currently selected configuration. On the server, the mapping function will no longer require Global Resources, but will use the resolved value instead.</p>
Path	<p>Click Browse, and select the path where the mapping function should be saved in the FlowForce Server container hierarchy. By default, the path is set to the /public container of FlowForce Server.</p> <p>From the dialog box, you can also create new containers or delete existing containers and mappings, provided that you have the required FlowForce Server permissions and privileges.</p>
Save mapping before deploying	This option is available if you are deploying an unsaved mapping. Select this check box to save the mapping before deployment.
Attach MFD files for later retrieval	This option enables you to deploy the MFD file together with its dependent input files (e.g., source XML file(s)), except for structure-defining files (e.g., XSD schemas). When you open the deployed mapping in FlowForce Server, the <i>Deployed Files</i> section will list all the files that you can download.

Setting	Description
Open browser to create new job	If you select this check box, the FlowForce Server Web administration interface opens in the browser after deployment, and you can start creating a FlowForce Server job immediately.

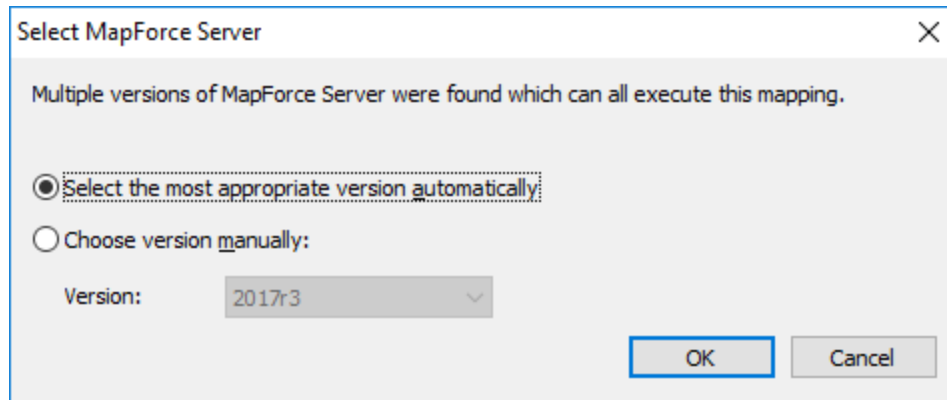
Troubleshooting

The following table lists problems that you might encounter when deploying a mapping, and their solution.

Problem	Solution
<p>Deploying the mapping returns the following error:</p> <pre>I/O operation on file ... failed. I/O Error 28: Failed to connect to <server> port 8082. Timed out System error 10060: A connection attempt failed because the connected party did not properly respond after a period of time, or established connection failed because connected host has failed to respond.</pre>	<p>Make sure that, on the target machine, the <i>FlowForce Web Server</i> service is running and configured to listen for connections on the specified port (8082, by default). Also, make sure that the firewall does not block incoming connections through this port.</p> <p>The <i>FlowForce Server</i> service must be running as well in order for the deployment to be possible.</p>
<p>Deploying the mapping returns the following error:</p> <pre>I/O operation on file ... failed. I/O Error 413: Payload Too Large</pre>	<p>This error may occur if an input file of the deployed mapping exceeds the maximum size limit of HTTP requests allowed by FlowForce Server (roughly 100 MB). You can increase the limit by setting the <code>max_request_body_size</code> option (in bytes) in the flowforcweb.ini and flowforce.ini files. For details, see Configuration File Reference.</p>

Selecting the server version (Windows only)

If the server where you deploy the mapping has multiple versions of MapForce Server running under FlowForce Server management (applicable to Windows servers only), then you are additionally prompted to specify the version of MapForce Server with which you want this mapping to be executed.



Note: The dialog box appears when the FlowForce Server installation directory contains .tool files for each MapForce Server version which runs under FlowForce Server management. By default, a MapForce Server .tool file is added automatically to this directory when you install MapForce Server as part of FlowForce Server installation. The path where the .tool files are stored in FlowForce is: **C:\Program Files\Altova\FlowForceServer2024\tools**. If you have additional versions of MapForce Server which you want to run under FlowForce Server management, their .tool files may need to be copied manually to the directory above. The .tool file of MapForce Server can be found at: **C:\Program Files\Altova\MapForceServer2024\etc**.

7.3 Run Mappings and Transformations as Jobs

You can create a FlowForce Server job from a MapForce mapping or StyleVision transformation as follows:

1. First, deploy the mapping or transformation to FlowForce Server. This step is done in MapForce (and StyleVision, respectively):
 - On the **File** menu, click **Deploy to FlowForce (Server)**.

For reference to the deployment settings, see [Deploying Mappings to FlowForce Server](#)⁴¹².

2. In FlowForce Server, navigate to the FlowForce container where you deployed the mapping or transformation (for example, the container "/public").

<input type="checkbox"/>	Name	Type
<input type="checkbox"/>	AutoCalc.transformation	StyleVision transformation
<input type="checkbox"/>	CompletePO.mapping	MapForce mapping

3. Click the required mapping or transformation, and then click **Create Job**. Alternatively, you can refer to the mapping or transformation from an existing job, by entering its path in the **Execute function** box:

You can now configure the job according to your needs. For example, you can run it as a [Web service](#)¹⁵⁸, or with the help of a [trigger](#)¹⁵¹. For a step-by-step example which illustrates deploying a StyleVision transformation and creating a job from it, see [Creating a Job from a StyleVision Transformation](#)³⁵¹. For a similar example for MapForce, see [Creating a Job from a MapForce Mapping](#)³⁰⁸. For an example job which calls both MapForce Server and StyleVision Server, see [Example: Generating Multiple PDFs from Multiple XMLs](#)³⁷¹.

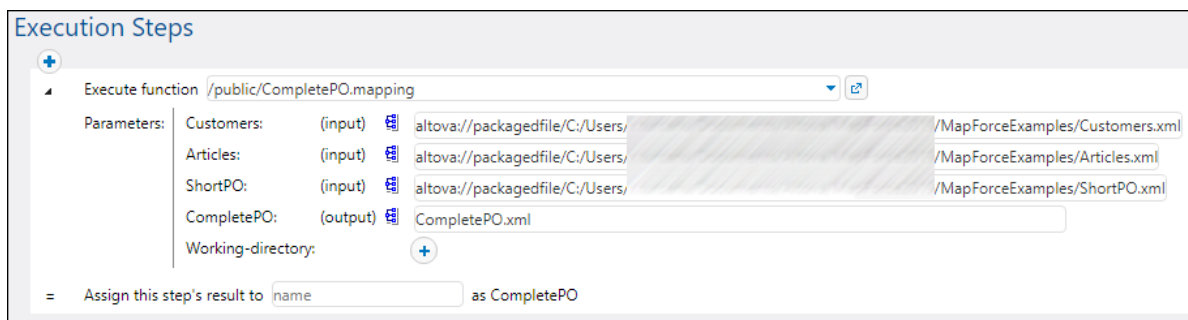
One of the most important parts of running a transformation or mapping job is handling the job input files. There are two approaches you can take: supply the input files statically to the job, or supply them dynamically at job runtime (for example, from a path). The exact approach to use depends on your needs. If your job needs to run

with the same input data every time, then the first approach is suitable. Otherwise, if you need your FlowForce jobs to pick up data from files supplied dynamically from a path, then the second approach must be used.

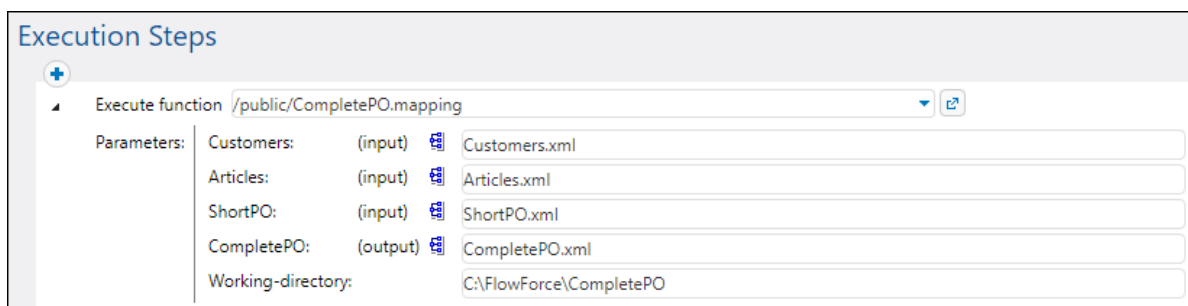
MapForce mappings

In case of mappings deployed from MapForce, any instance files (such as XML, CSV, JSON, Excel, and so on) are deployed together with the mapping and implicitly packaged as static. This means that, when the job runs, FlowForce will read data from the statically packaged files by default, which might not always be what you need. There are two scenarios here:

1. If you right-click the mapping in MapForce and select the **Make paths absolute in generated code** check box before deploying the mapping, all the input files explicitly appear with the prefix `altova://packagefile/` in FlowForce Server.



To instruct FlowForce Server not to read data from packaged files, remove the prefix `altova://packagefile` from the path. You can then refer to the file using either an absolute or a relative path. If using a relative path, the path is relative to the **Working Directory** parameter. For example, if you intend to provide as input some files from `C:\FlowForce\CompletePO`, then set the working directory to `C:\FlowForce\CompletePO` and enter just the name of the input files, as shown below.



2. If the **Make paths absolute in generated code** check box is NOT selected before deploying the mapping to FlowForce, the input files are shown with their relative path in FlowForce. Note that FlowForce will still read data from the packaged file in this case as well, even when there are files with the same name in the working directory. To instruct FlowForce not to read data from the packaged file, you can either make the file paths absolute or supply them as parameters to the job, as shown below:

Job Input Parameters

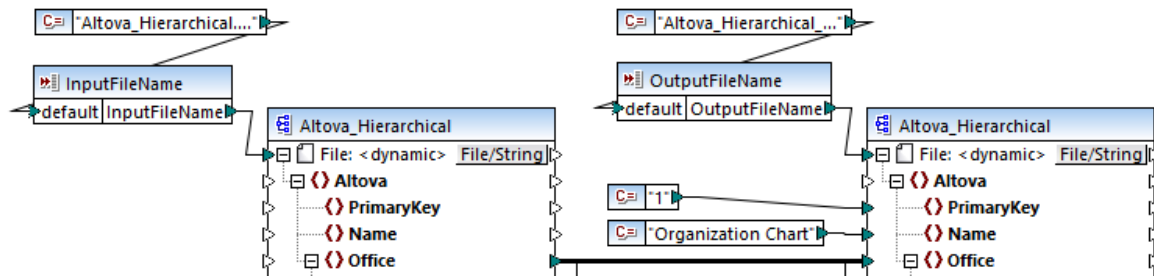
- Name: customers Type: string as file Default: C:\Customers.xml
- Name: articles Type: string as file Default: C:\Articles.xml
- Name: shortPO Type: string as file Default: C:\ShortPO.xml

Execution Steps

Execute function /public/CompletePO.mapping

Parameters:	Customers:	(input)	{customers}
	Articles:	(input)	{articles}
	ShortPO:	(input)	{shortPO}
	CompletePO:	(output)	CompletePO.xml
	Working-directory:		+

Alternatively, you can change the mapping design in MapForce so that the input file names are input parameters to the mapping. For example, the mapping illustrated below takes both the input and output file names as parameters.



When deployed to FlowForce Server, the parameters appear as such in the job configuration page (the files themselves are not packaged).

Execution Steps

Execute function /public/FileNamesAsParameters.mapping

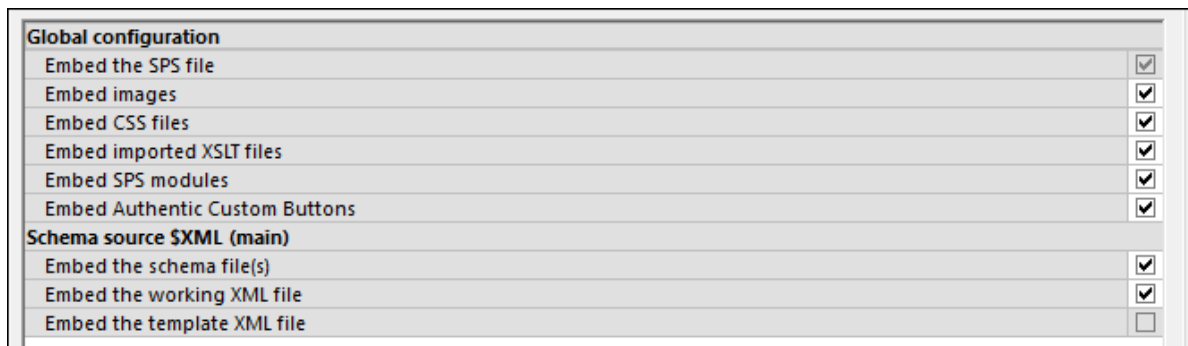
Parameters:	OutputFileName:	output.xml
	InputFileName:	Altova_Hierarchical.xml
	Working-directory:	C:\Users\... \MapForceExamples

The mapping illustrated above is called **FileNamesAsParameters.mfd** and is one of the example files that ship with MapForce. For information about how this mapping is designed, refer to the MapForce documentation.

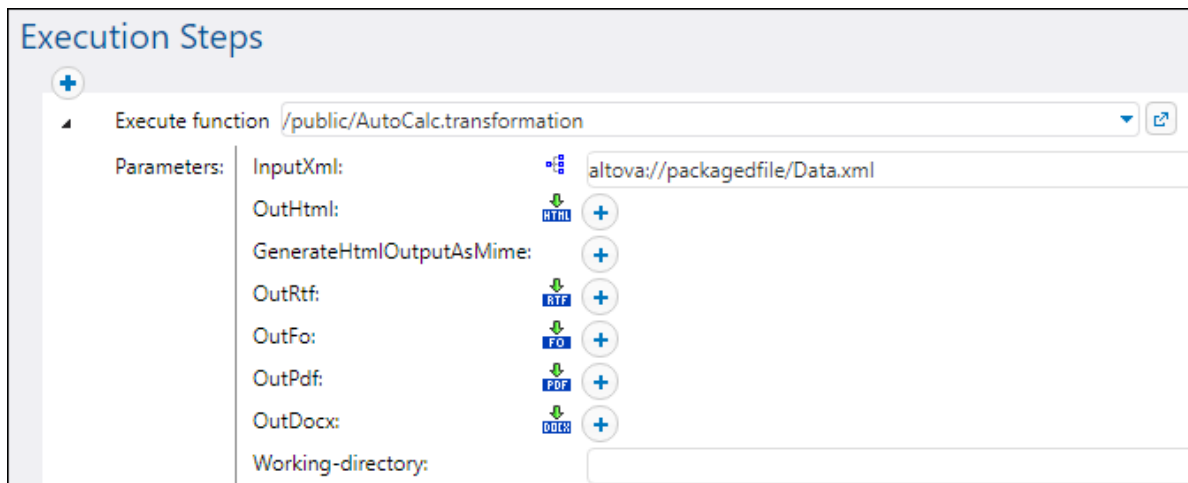
StyleVision transformations

In case of StyleVision transformations, you can handle input files as follows:

1. Open the PXF (Portable XML Form) file in StyleVision. If you have a SPS (StyleVision Power Stylesheet), StyleVision will prompt you to convert it to PXF format when you attempt to deploy it to FlowForce Server.
2. In the Design Overview window, click **Configure embedded files**. A dialog box appears.



3. Notice the option **Embed the working XML file**. If you select this check box, the working XML file will be part of the deployed package and, by default, FlowForce Server will read data from it each time when the job runs. A packaged file is indicated as such in FlowForce:



To supply the file dynamically to the job, remove the prefix `altova://packagedfile/` or change the path to an absolute one. If using a relative path, the path is relative to the **Working Directory** parameter. Alternatively, clear the **Embed the working XML file** check box before deploying the transformation to FlowForce Server.

If you clear the **Embed...** check box for resources like CSS files or images, FlowForce Server will look for them in the job working directory.

7.3.1 Credentials in Mapping Functions

Earlier in this documentation, you have seen an introduction to [Credentials](#)¹⁶². Recall that it is possible to create credentials not only in FlowForce Server, but also at mapping design time, in MapForce.

When you deploy a mapping containing credentials from MapForce to FlowForce Server, the credentials are deployed to the server as well. The deployed information will contain only the fields that you filled in when creating the credential record. For example, this may be an empty credential (if you chose to store only the credential name) or a credential object that contains both the username and password.

You can also deploy credential objects from MapForce to FlowForce Server as standalone objects, separately from the main mapping. You can choose directly from MapForce the target container where they should be deployed. For more information, refer to MapForce documentation (<https://www.altova.com/documentation>).

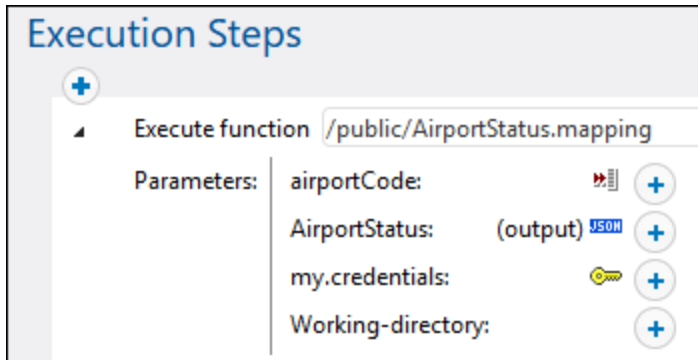
The field **Password** is considered sensitive data.

The sensitive data will be deployed only if you selected the **Include in MapForce Server Execution File and Mapping Deployment** check box at mapping design time in MapForce. This applies both when you deploy the mapping and when you deploy the standalone credentials.

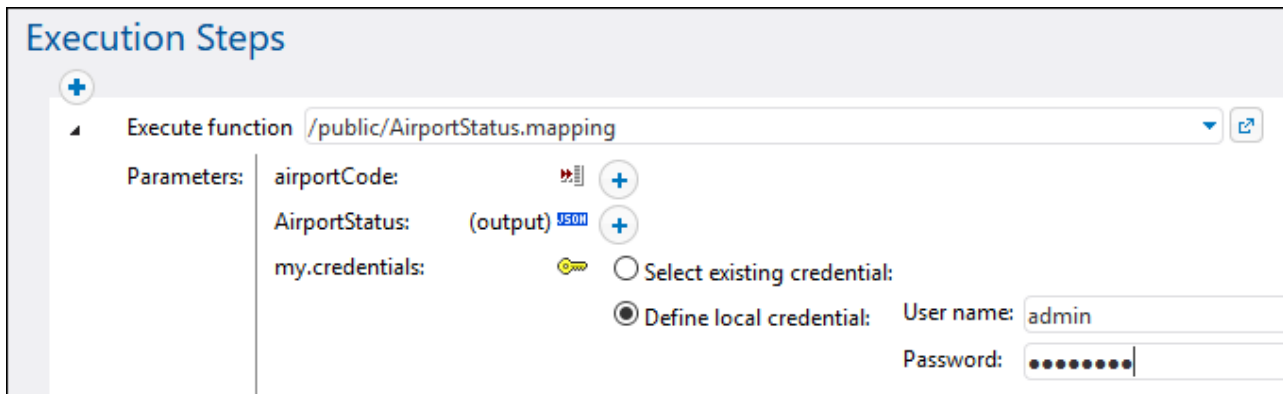
In FlowForce Server, you can see whether a mapping function needs credentials by opening the page of the respective mapping function, for example:

Name	Type	Other
airportCode	string	optional
AirportStatus	string	(output) JSON Default: AirportStatus.json
my.credentials	credential	
Working-directory	string as directory	Default: <input type="text"/>

If you selected the **Include in MapForce Server Execution File and Mapping Deployment** check box when creating the credential, then the job will use the credentials deployed together with the mapping. In this case, you don't need to specify them from the job configuration page. For example, the following execution step will run the mapping function with the stored credentials if such exist (notice that the "my.credentials" parameter is not expanded):



You can always override the stored credentials with any other credential object that was defined directly in FlowForce Server, or with some local credentials. To do this, click the "+" button and either select a credential object that already exists in FlowForce Server, or enter the username and password directly, for example:



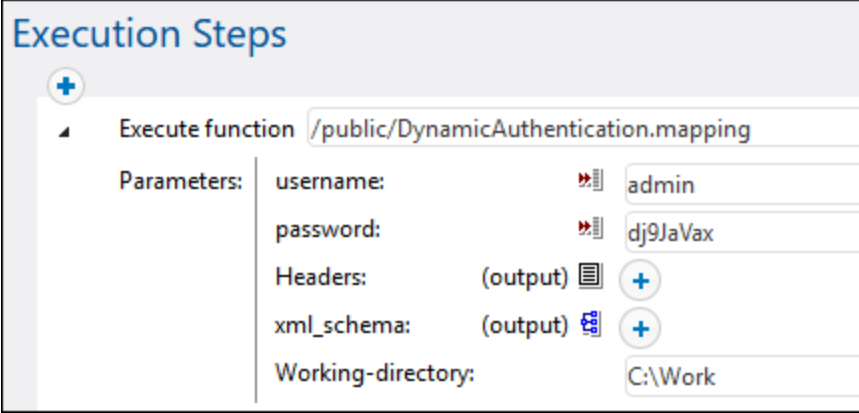
The credentials supplied as parameter to the execution step take precedence over credentials stored inside the mapping function.

If you did not select the **Include in MapForce Server Execution File and Mapping Deployment** check box when creating the credential in MapForce, it is mandatory to supply credentials as parameters to the execution step; otherwise, the job execution will fail.

7.3.2 Dynamic Authentication

In MapForce, it is possible to configure mappings that call Web services for basic HTTP authentication. Dynamic authentication is one of the ways to achieve this; it is an alternative to using credentials. Dynamic authentication means designing the mapping so that it accepts the username and password as input parameters. For details about configuring dynamic authentication, refer to MapForce documentation (<https://www.altova.com/documentation>).

When you deploy a mapping containing dynamic authentication to FlowForce Server, the username and password become input parameters to the mapping function. Any FlowForce Server job that calls such a mapping function will require the username and password before it can run successfully, for example:



Parameters:	
username:	admin
password:	dj9JaVax
Headers:	(output) +
xml_schema:	(output) +
Working-directory:	C:\Work

In the example illustrated above, the username and password are simply entered in the respective text boxes. However, you can also supply them as input parameters to the job, see [Managing Input Parameters](#).

7.3.3 Resources

Altova Global Resources are aliases for file, folder, and database resources. Each alias can have multiple configurations, and each configuration maps to a single resource. Therefore, when you use a global resource, you can switch between its configurations. For example, you could create a database resource with two configurations: `development` and `production`. Depending on your goals, you can switch between these configurations. In FlowForce Server, you can retrieve data from the `development` or `production` database by supplying the desired configuration to the mapping function.

Global resources can be used across different Altova applications (*see subsection below*).

Global resources in other Altova products

When stored as global resources, files, folders, and database connection details become reusable across multiple Altova applications. For example, if you often need to open the same file in multiple Altova desktop applications, you can define this file as a global resource. If you need to change the file path, you will need to change it only in one place. Currently, global resources can be defined and used in the following Altova products:

- [Altova Authentic](#)
- [DatabaseSpy](#)
- [MobileTogether Designer](#)
- [MapForce](#)
- [StyleVision](#)
- [XMLSpy](#)
- [FlowForce Server](#)
- [MapForce Server](#)
- [RaptorXML Server/RaptorXML+XBRL Server](#)

For more information about creating Global Resources, refer to the "Altova Global Resources" chapter of MapForce documentation.

Resources in FlowForce Server

In FlowForce Server, global resources are not stored in one XML file as in desktop applications. In FlowForce, each resource is a reusable object that may contain file or folder paths or database connection details. Resources can be copied, exported, and imported, and are subject to the same user access mechanism as other FlowForce Server objects. This means that any FlowForce user can use any resource in their mapping functions if they have the required permissions.

Once you have created a mapping with global resources in MapForce, you can deploy it to FlowForce Server. At deployment time, if you want your mapping to use global resources, select the **Use Resources** check box in the deployment dialog box. If you do not select the check box, any global resources used by the mapping will be resolved, based on the currently selected configuration. If you have selected the check box, the mapping function will require resources in FlowForce Server as well. The screenshot below is an example of a mapping function deployed to FlowForce that requires resources to run. Notice that the first parameter gets the default file path from a resource.

Function ReadJSON.mapping in /public

Function Input Parameters

Name: People	(input) <small>JSON</small> Type: string	Default: altova://file_resource/SourceFile
Name: Text file	(output) <small>Text file icon</small> Type: string	Default: Text file.csv
Name: Working-directory	Type: string as directory	Default: <input type="text"/>

Resources

Run function using resources:

In FlowForce Server, it is the mapping function that uses the global resources, not the job. The mapping function reads the path of the first input file from the resource. This means that all jobs using this function will use the same path unless you override the path from the job configuration page.

You can also deploy global resources to FlowForce Server as standalone objects. This means there is no need to deploy a mapping first in order to be able to deploy a global resource. For more information about deploying global resources to FlowForce Server, see [the MapForce documentation](#).

Structure of resources

In all Altova desktop applications, global resources are maintained as XML files. The default file is called **GlobalResources.xml**; you can find it in the **C:\Users\<username>\Documents\Altova** directory on the computer where MapForce is installed. A Global Resource file may contain multiple resources, also known as "aliases". An alias is either a file path, or a directory path, or a group of database connection details. Aliases, in their turn, can have multiple configurations. As described previously, configurations enable you to switch

paths or databases. This is best understood by looking at the structure of the following sample Global Resource file (note some data was omitted for simplicity):

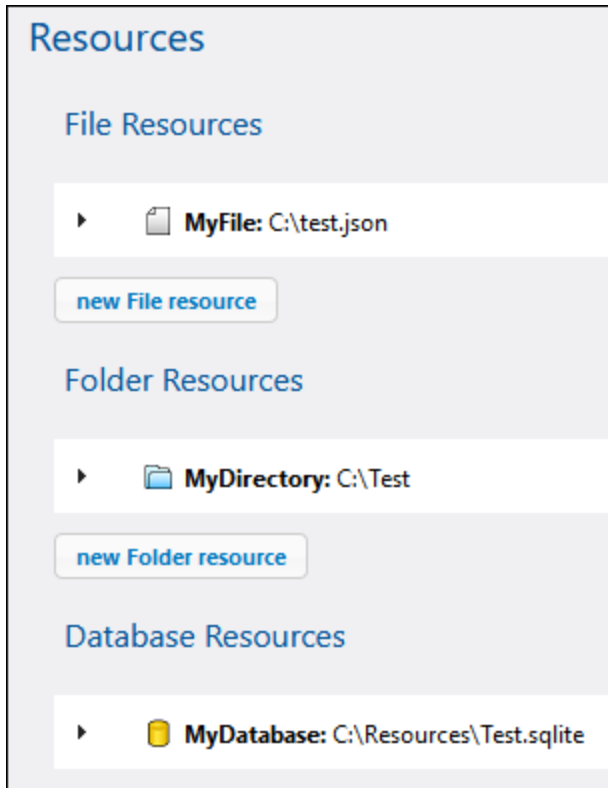
```
<Resources>
  <Resource Alias="MyFile">
    <Configurations>
      <Configuration Location="C:\test.json" ContentKind="File"
Configuration="Default"/>
      <Configuration Location="C:\production.json" ContentKind="File"
Configuration="Production"/>
    </Configurations>
  </Resource>
  <Resource Alias="MyDirectory">
    <Configurations>
      <Configuration Location="C:\Test" ContentKind="Folder"
Configuration="Default"/>
      <Configuration Location="C:\Production" ContentKind="Folder"
Configuration="Production"/>
    </Configurations>
  </Resource>
  <Resource Alias="MyDatabase">
    <Configurations>
      <Configuration ContentKind="DataSource" Configuration="Default">
        <DatabaseContextInfo vendor="sqlite" connection="C:
\Resources\Test.sqlite"/>
      </Configuration>
      <Configuration ContentKind="DataSource" Configuration="Production">
        <DatabaseContextInfo vendor="sqlite" connection="C:
\Resources\Production.sqlite"/>
      </Configuration>
    </Configurations>
  </Resource>
</Resources>
```

The file above defines three resources (aliases): a file path called "MyFile", a directory path called "MyDirectory", and a SQLite database called "MyDatabase". Each alias has two configurations: a default configuration used for testing, and a production configuration.

In FlowForce Server, because of the specifics of the multi-user server environment, resources work slightly differently. Specifically, an XML resource file such as the one above becomes a resource object in FlowForce. Inside the resource object, there can be multiple aliases, just like in desktop applications. However, each alias has only one configuration, and that is the configuration that you've selected upon deploying the resource from MapForce to FlowForce Server.


Whenever you deploy Global Resources from MapForce to FlowForce Server, only one of the configurations is deployed at a time.



For example, if you deployed the global resource file above, either the "Default" or "Production" configuration will be deployed to the server (not both at the same time). If you choose the "Default" configuration, the resource object would look as follows in FlowForce Server:



In FlowForce, any mapping function can consume one specific configuration of a global resource. Therefore, in this example, if you need the "Production" configuration on the server, you should deploy the same resource file once again, this time selecting the configuration "Production" from the deployment dialog box in MapForce. Alternatively, you can create a resource directly on the server, as described below, and change the mapping function to point to it instead of the "Default" resource. Note, however, that the alternative approach is possible with file and directory resources, not with databases.

Changing the resource of a mapping function

In FlowForce, resource objects are identified by the  icon. Therefore, if you've deployed both the "Default" and the "Production" configurations from the example above, the corresponding resources in FlowForce Server may appear as follows:

<input type="checkbox"/>	Name	Type
<input type="checkbox"/>	 ExampleGlobalResources_Default.resources	resources
<input type="checkbox"/>	 ExampleGlobalResources_Production.resources	resources

To change the resource used by a mapping function:

1. Go to the container where the mapping function was deployed and click to open the function.
2. Under "Resources", select a new resource path. Selecting resources works in the same way as with other FlowForce objects such as functions, credentials, and so on.



If the mapping function does not have a "Resources" section, this mapping was not configured for Global Resources in MapForce (or the **Use Resources** check box was not selected on deployment).

Any mapping function can use any resource, if the following requirements are satisfied:

- The resource kind is compatible with the function. For example, a "folder" resource is not suitable if the mapping function needs a "file" resource.
- The resource alias name is the one required by the mapping function. You normally select the alias name at mapping design time, in MapForce, but you can also override it in FlowForce, as further described below.

Resources and job configuration

As stated before, resources are consumed at mapping function level, not at job configuration level. When a job runs, it consumes those global resources that are defined in the function called by the job. Therefore, when you edit a job from the job configuration page, you have only very minimal configuration options with respect to resources, like "Overriding the resource alias" (further described below).

In some cases, it may be possible to reference a resource (like a folder or file) directly from the job configuration page. Please note that this may not work in all contexts and should be generally avoided unless you have a very good reason to use such references.

Note: It is not supported to refer to a resource from the "Working Directory" parameter of an execution step. This is because processing of resources requires that the MapForce Server process be already started, whereas the working directory is set *before* MapForce Server starts.

Overriding the resource alias

Even if a file or folder resource can have multiple aliases, only one of them is used at job runtime. The alias used at runtime is the one selected in MapForce while designing the mapping. For example, the following MapForce component is configured to generate **output.csv** to a directory alias called "MyDirectory". If you deploy this mapping to FlowForce Server, the mapping function on the server must also point to a resource that contains the "MyDirectory" alias.

As an alternative to editing the mapping in MapForce whenever you need to change the alias, you can also override the alias in FlowForce Server, from the job configuration page. To override file or folder aliases in a job, use the following syntax, replacing **MyFile** or **MyDirectory** with the required alias name:

Resource kind	Example
File	altova://file_resource/ MyFile
Directory	altova://folder_resource/ MyDirectory

For example, in the job configuration below, the directory alias was changed to "TestDir".

Note: Overriding the alias as shown above is not supported for database resources. If you have multiple databases aliases, switch to the required database alias in MapForce *before* deploying the mapping to FlowForce Server.

Creating resources

You can create only file or folder resources in FlowForce Server. To create a global resource in FlowForce Server, open a container of choice and click **Create | Create Resource**.

The screenshot shows a web-based configuration interface for resources. At the top, there are two input fields: 'Resources name:' with the value 'Resource1' and 'Resources description:' with the text 'Defines miscellaneous input or output paths used in mapping jobs.'. Below this is a section header 'Resources' in blue. Underneath, there is a sub-section 'File Resources'. A table-like structure shows a single entry with 'Resource alias:' set to 'SourceFile' and 'File:' set to 'C:\Resources\data.txt'. Below the table are two buttons: 'new File resource' and 'new Folder resource'. At the bottom of the interface is a 'Save' button.

Note: Creating database resources is not supported in a server environment. To create database resources, use the Global Resources editor of MapForce or any other Altova desktop application that supports Global Resources, and then deploy the resources from MapForce to FlowForce Server.

The resource alias should match the one required by the mapping function where you will use this resource. Otherwise, you will need to tweak jobs manually so that they point to the correct alias, as described above in "Overriding the resource alias".

Within the same resource object, you can create multiple aliases if required, by clicking the **New File Resource** or **New Folder Resource** buttons. This is optional, however. If you create multiple aliases, remember that you will need to modify jobs so as to indicate which alias it should use.

Editing resources

You can edit file or folder resources directly in FlowForce Server, as an alternative to doing this in MapForce and deploying them again. To edit a resource, click the respective record, update the paths (or the database connection details), and then click **Save**.

Note: In case of database resources, you can edit in FlowForce only certain fields such as the connection string or default database. It is, however, not possible to change the database vendor and connection method.

Updating a resource affects with immediate effect all of the following:

- All the mapping functions referencing that resource
- All the jobs that call the respective mapping functions.

7.4 Access the Mapping/Transformation Result

After a MapForce mapping or StyleVision transformation has been deployed to FlowForce Server, it becomes a FlowForce function which can be called from other execution steps. For example, in the first step of the job below, a mapping function called **SimpleTotal.mapping** is being executed.

The screenshot displays the 'Execution Steps' configuration window. It contains two steps:

- Step 1:** 'Execute function /public/SimpleTotal.mapping'. Parameters include 'ipo: (input)' and 'Working-directory:'. The result is assigned to the field 'output'.
- Step 2:** 'Execute function /system/compute'. The parameter 'Expression:' is set to 'stdout(output)'. The result is assigned to the field 'name' as 'T0'.

At the bottom, there are four buttons: 'new Execution step', 'new Choose step', 'new For-each step', and 'new error/success handling step'.

Notice that the job consists of two steps:

1. Step 1 calls MapForce Server to actually run the **SimpleTotal.mapping** function. Importantly, the **Assign this step's result to** field gives a name to the mapping result (in this case, it is `output`; however, it can be any name you choose).
2. Step 2 calls the `/system/compute` ²³⁷ function which converts the output of the mapping to a stream.

By default, the output of a mapping or transformation function is of generic type **result**. In order for the output to become useful, **result** must be converted to whatever data type you require (for example, string, stream, file). For this purpose, the `/system/compute` ²³⁷ built-in function is available, as well as various FlowForce expression functions. In the example above, the built-in function `/system/compute` ²³⁷ was called to perform the required data type conversion. Namely, the expression `stdout(output)` converts the result of the previous step to a stream.

The table below lists examples of FlowForce expressions that you will likely need to process the result of a mapping or a transformation function. Remember that, in all these examples, `output` is the name you entered in the **Assign this step's result to** field.

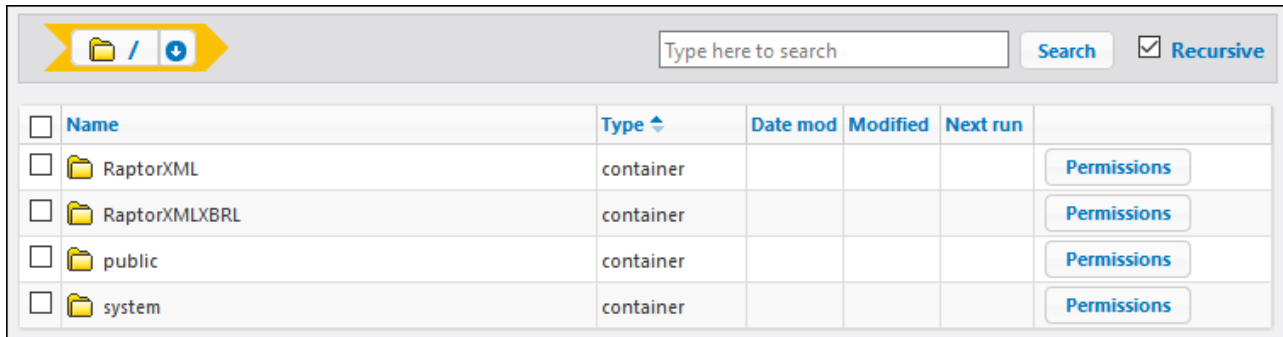
FlowForce Expression	Purpose
<code>stdout(output)</code>	Converts <code>output</code> to a stream.
<code>content(stdout(output))</code>	Converts <code>output</code> to string.

FlowForce Expression	Purpose
<code>as-file(stdout(output))</code>	Converts output to a file.
<code>as-file(nth(results(output), 0))</code>	This kind of expression is required if output consists of multiple files. This happens when the mapping or transformation function was designed (in MapForce or StyleVision) to generate not just a single output, but multiple outputs. The expression converts output to a sequence of streams, picks up the first stream from the sequence, and converts it to file. For an example, see Creating a Job from a StyleVision Transformation ³⁵¹ .
<code>as-file(nth(results(output, "CompletePO"), 0))</code>	Same as above, except that the file is retrieved from the sequence of streams not by its zero-based index as above, but by name (in this case, "CompletePO").

For complete reference to FlowForce expression functions that are available to handle the result of a step or job, see [Step Result Functions](#)²⁰². For an introduction to FlowForce expressions, see [FlowForce Expressions](#)¹⁷¹.

7.5 Integration with RaptorXML Server

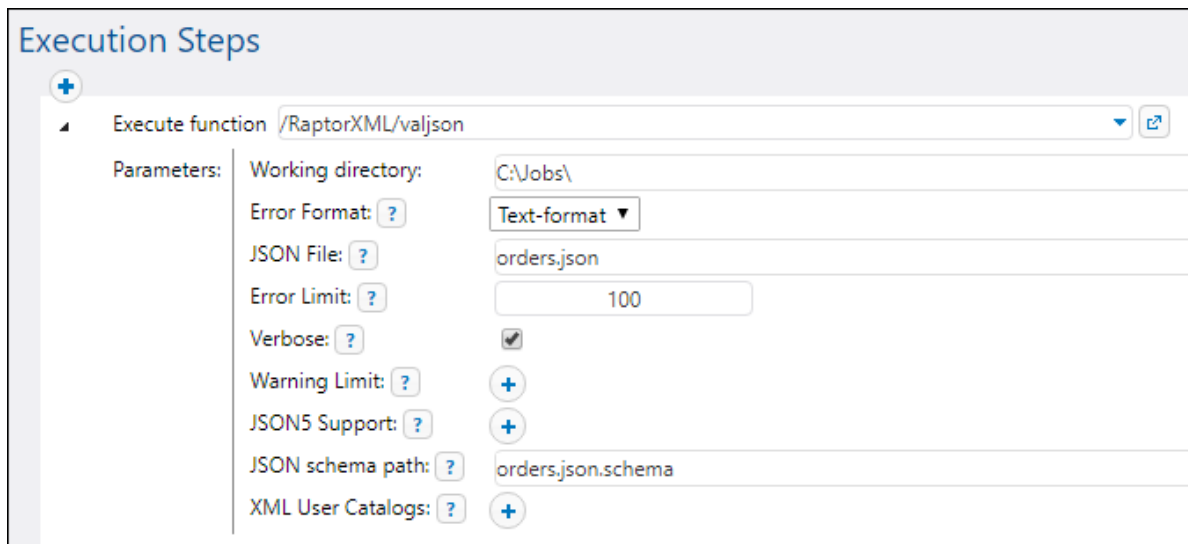
When RaptorXML is integrated into FlowForce, all the functions exposed by RaptorXML Server become available to FlowForce so that you can call them in jobs. More specifically, the RaptorXML functions exist in the **/RaptorXML** container of FlowForce. In case of RaptorXML+XBRL Server, the container name is **/RaptorXMLXBRL**.



<input type="checkbox"/>	Name	Type	Date mod	Modified	Next run	
<input type="checkbox"/>	RaptorXML	container				Permissions
<input type="checkbox"/>	RaptorXMLXBRL	container				Permissions
<input type="checkbox"/>	public	container				Permissions
<input type="checkbox"/>	system	container				Permissions

You can call the RaptorXML functions from jobs similar to calling FlowForce built-in functions:

- In the **/RaptorXML** (or **/RaptorXMLXBRL**) container, open the function of interest, and then click **Create Job**. You can either reference generic functions such as **/RaptorXML/valjson** or release-specific functions such as **/RaptorXML/2024/valjson**. The differences between the two are described below.
- Create a new execution step in a job, and call the desired RaptorXML function from an execution step. For example, the step below calls the **valjson** function:



Parameters:	Value
Working directory:	C:\Jobs\
Error Format: ?	Text-format ▼
JSON File: ?	orders.json
Error Limit: ?	100
Verbose: ?	<input checked="" type="checkbox"/>
Warning Limit: ?	+
JSON5 Support: ?	+
JSON schema path: ?	orders.json.schema
XML User Catalogs: ?	+

For examples of jobs that call RaptorXML Server, see:

- [Validate a Document with RaptorXML](#) ³⁵⁹
- [Validate XML with Error Logging](#) ³⁶¹
- [Use RaptorXML to Pass Key/Value Parameter Pairs](#) ³⁶⁶

For reference to all the RaptorXML functions, refer to the RaptorXML Server documentation (<https://www.altova.com/documentation>).

Manual integration

Integration between FlowForce Server and RaptorXML Server takes place automatically in many cases (for example, when you run the FlowForce Server installation on Windows and choose to install RaptorXML Server as well). However, there are also cases when manual integration between the two is necessary. Manual integration is typically required when FlowForce Server and RaptorXML Server of different versions were installed separately. For example, if the function definitions of a specific RaptorXML Server version are missing from the FlowForce Server interface even though that version of RaptorXML Server is installed, then manual integration is required.

To perform a manual integration, run the script available at the following path: **{RaptorXML installation directory}\etc\functions\integrate.bat**.

Note: On Unix systems, the script name is **integrate.cs**. Superuser privileges (sudo) are required to run this script.

This script takes two arguments: the path to the FlowForce Server installation directory and the path to the FlowForce Server data directory (see [FlowForce Server Application Data](#)⁶⁴). When you run the script, the following happens:

- All the release-specific functions of the integrated RaptorXML Server version become available to FlowForce Server so you can call them as jobs.
- The generic (release-agnostic) RaptorXML functions are updated to point to the release-specific functions of the integrated RaptorXML version.

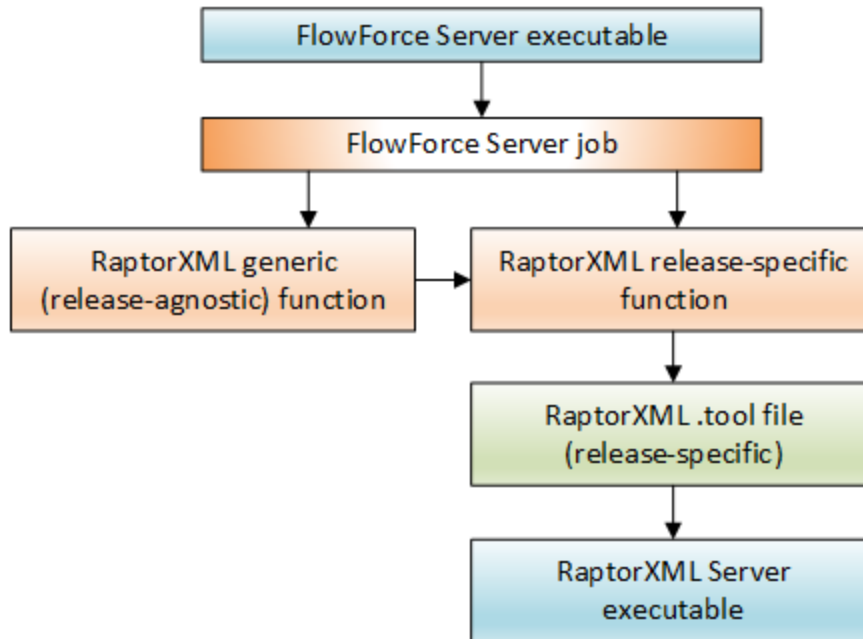
If the script returns errors, the function definitions of the integrated RaptorXML version are not compatible with FlowForce Server. In the unlikely event that this happens, please contact support.

Generic versus release-specific RaptorXML functions

The functions available in the **RaptorXML** or **RaptorXMLXBRL** containers are organized as follows:

- Functions from the **/RaptorXML** container are backward compatible down to the 2014 version of FlowForce Server (which is the first version supporting RaptorXML functions). These generic functions act as wrappers to the release-specific functions from the **/RaptorXML/{Release}** container. They are guaranteed to be compatible between releases but they do not provide all the features of the latest installed RaptorXML Server.
- Functions from the **/RaptorXML/{Release}** containers provide all the features of the corresponding RaptorXML release. These functions are compatible with FlowForce Server of the same release. However, any version of RaptorXML Server is not necessarily compatible with any version of FlowForce Server. You can check compatibility by running an integration script (as described under "Manual integration").

If a job calls a generic RaptorXML function, the function acts as a wrapper to the equivalent release-specific function of the RaptorXML Server. The selected RaptorXML release is the one that was most recently integrated into FlowForce, including manually-integrated releases. Still, as mentioned above, such calls will not benefit from the latest RaptorXML features (such as new arguments or even functions). To make use of the latest RaptorXML features from FlowForce jobs, call a release-specific function directly.



A release-specific function determines which RaptorXML .tool file should be used in order to look up the RaptorXML executable. A separate .tool file exists for each RaptorXML Server release. A .tool file instructs FlowForce Server about the location of the RaptorXML Server executable and can also be used to set environment variables, see [Setting Environment Variables](#)⁴³⁶.

If your FlowForce jobs refer to version-specific RaptorXML functions, and if you would like to upgrade to a newer version of FlowForce Server and RaptorXML Server, you can either modify all the jobs to point to the latest release-specific RaptorXML functions, or you can map the **Raptor.tool** file to a newer version of the RaptorXML Server executable, as follows:

1. Copy the **Raptor_<release>.tool** file from **{installation}\etc** directory of RaptorXML Server of the latest installed release to the **{configuration data⁶⁴}\tools** directory of FlowForce Server of the same release.
2. Rename the file to match the version of the old release (the Raptor release your jobs are pointing to). For example, if the old release is **RaptorXML 2017r3**, then rename the file to **Raptor_2017r3.tool**.

If you take the mapping approach, all the existing jobs will continue to look as if they call RaptorXML 2017r3 functions, whereas the .tool file will map in fact to the latest RaptorXML Server executable.

7.6 Tool Files

When you install other Altova servers alongside FlowForce Server, for example, by selecting the relevant server products in the FlowForce Server installation wizard or installing these server products using their stand-alone installer later, a `.tool` file is installed for each application that runs under FlowForce Server management. The following Altova products can run under FlowForce Server management: [MapForce Server](#), [StyleVision Server](#) and [RaptorXML Server](#). Usually, you do not need to configure `.tool` files unless you need to change environment variables such as `CLASSPATH` for MapForce Server and StyleVision Server.

FlowForce Server uses `.tool` files to locate and configure the execution of the other server applications under its management. FlowForce Server searches for `.tool` files in the instance-data directory, referred to as `INSTANCEDIR`, and the installation directory, referred to as `INSTALLDIR`. FlowForce Server first scans `INSTANCEDIR` and then `INSTALLDIR`. The tables below show the paths of these directories for different operating systems. Note that the directories shown for the `INSTANCEDIR` are default paths. During the configuration of FlowForce Server, you can [set your custom path to the INSTANCEDIR](#) ⁴⁵.

FlowForce Server instance-data directory (INSTANCEDIR)	
Linux	<code>/var/opt/Altova/FlowForceServer/data</code>
macOS	<code>/var/Altova/FlowForceServer/data</code>
Windows	<code>C:\ProgramData\Altova\FlowForceServer\data</code>

FlowForce Server installation directory (INSTALLDIR)	
Linux	<code>/opt/Altova/FlowForceServer2024/</code>
macOS	<code>/usr/local/Altova/FlowForceServer2024/</code>
Windows	<code>C:\Program Files\Altova\FlowForceServer2024\ C:\Program Files (x86)\Altova\FlowForceServer2024\</code>

The `INSTANCEDIR` is usually an empty directory, where you place any customized tool files. The `INSTALLDIR` directory is managed by the installation process, and the `.tool` files contained in it must not be edited.

Information messages

FlowForce Server groups running tool process instances and manages them as configured in the `.tool` files. When FlowForce enforces the rules regarding the lifetime of tool process instances, all these events may produce information messages in the log. For example:

```
Starting instance {id} of {tool} for {session}.
Starting {commandline}.
Instance {id} of {tool} for {session} is now idle.
Shutting down instance {id} of {tool} for {session}; sitting idle for too long.
Shutting down instance {id} of {tool} for {session}; maximum reuse count reached.
Instance {id} of {tool} for {session} unexpectedly ceased communication.
Instance {id} of {tool} for {session} attached to job instance {instanceid}.
```

The information messages listed above do not indicate licensing or queuing issues. Instead, they make it possible to track down potential problems, for example, by offering information about processes that were running at a particular time. If steps or jobs fail, this will generate a separate log message.

Tool file editing

Files with a `.tool` extension can be edited in a text editor (e.g., Notepad++). The following editing options are available:

1. The executable `path` under the `[Tool]` section. Changing this path might be necessary in certain cases, for example, when you need to make `.tool` files of older versions execute newer versions, or vice versa.
2. The `[Environment]` section. You can add or edit this section in order to define environment variables required by the tool. For more information, see the subsection below.

Important:

- When you edit a `.tool` file in `INSTANCEDIR`, changes take effect at once. You do not have to restart FlowForce Server.
- Do not change any `.tool` file settings other than the ones mentioned above, unless advised by Altova Support.
- It is not possible to define custom tools.

Environment variables

When MapForce Server mappings or StyleVision Server stylesheets run under FlowForce Server management, they may require setting environment variables. For example, you need to set `CLASSPATH` to specify the location of the JDBC drivers when connecting to a database. To set environment variables required by MapForce Server mappings or StyleVision Server transformations, edit the `.tool` file of the respective Altova server product. To edit the `.tool` file, first check if it already exists in the `INSTANCEDIR` directory. If the `.tool` file does not exist in `INSTANCEDIR`, copy it from `INSTALLDIR` of FlowForce Server.

You would find `.tool` files in the `INSTALLDIR` directory only if MapForce Server or StyleVision Server were installed after FlowForce Server. If the `.tool` file exists neither in `INSTANCEDIR` nor in `INSTALLDIR`, it is likely that FlowForce Server was installed after MapForce Server or StyleVision Server. In this case, you can find the `.tool` file in the `etc` directory relative to the MapForce Server or StyleVision Server installation directory.

You can add the required environment variables under the `[Environment]` section in the `.tool` file. The environment variables set in the `.tool` file override the environment variables defined by other means. The example of a `.tool` file (Linux) which sets the `CLASSPATH` variable is given below:

```
[Environment]
CLASSPATH=./usr/local/jdbc/oracle/ojdbc6.jar
```

Note: If you run the [migratedb](#)³⁹⁷ command while upgrading to a new major version of FlowForce, any `.tool` files from the application data directory of the previous version will be copied over to the application directory of the new version. This may have unwanted consequences. Therefore, make sure that the application data directory contains the `.tool` files that you actually need.

For information about executing shell commands or scripts as FlowForce Server jobs, see the [/system/shell/commandline](#)²⁸⁵ function.

Index

A

Active Directory,

integration with FlowForce Server, 119

Administration, 109

mail settings, 118
 parameters for system function /system/mail/send, 118
 password policies, 116
 reports, 114
 roles, 112
 Settings, 118
 test mail, 118
 test SMTP parameters, 118
 users, 109

Administration Tasks, 72

backup, 75
 data migration, 75, 76
 data recovery, 75, 76
 localize FlowForce Server, 79
 migratedb command, 76
 privileges, 72
 roles, 72
 upgradedb command, 76
 users, 72

ADO,

database connections, 406

ADO.NET,

database connections, 406

Application pools, 436

Authentication,

HTTP, 158
 Windows domain, 158

C

Command line interface,

getting help on, 387

Configuration via Configuration Files and CLI, 64

configuration data, 67
 configuration files, 64

export configuration data, 288
 export to another FlowForce Server instance, 288
 export to file, 288
 import configuration data, 288, 292
 import from file, 288
 instance parameters, 67
 missing dependencies, 292
 network settings, 67
 overview, 64
 set network address and port, 67
 SSL encryption, 67

Configuration via Setup page,

add an existing instance, 45
 bind address, 47
 certificate chain file, 47
 certificate file, 47
 create a new server instance, 45, 46
 create self-signed SSL certificates, 55
 default time zone, 47
 finish setup, 60
 host name, 47
 HTTPS connections, 47
 install services, 60
 Instances, 45
 master instance encrypted connection, 47
 network address and port, 47
 private key file, 47
 register with LicenseServer, 45
 settings, 47
 setup command, 47
 setup command options, 47
 Setup page, 47
 SSL encrypted connection, 47
 SSL encryption, 47, 50, 55
 start services, 60
 unencrypted connection, 47

Container,

restricting access to /public, 102

Containers,

creating, 97
 moving, 97
 overview of, 95
 renaming, 97
 setting permissions on, 100
 viewing permissions of, 99

Credentials,

add, 163
 create, 163

Credentials,

- define, 163
- OAuth 2.0, 162
- Password, 162, 163
- SSH key, 162
- types, 162

D**Data types,**

- in FlowForce, 177

Default time zone,

- configuration of, 118

Digital certificates,

- managing on Windows, 55
- trusting on Linux, 55
- trusting on Mac, 55
- trusting on Windows, 55

Directory polling job,

- example of, 319

E**Environment variables,**

- setting, 436

Error handling,

- adding to a job, 325

Error logging,

- adding to a job, 361

Error/success handling,

- Always, 136
- On-Error, 136
- On-Retry, 136
- On-Success, 136
- order, 136
- types, 136

Expression Functions, 180, 220

- Boolean, 185
- Execution State, 229
- File System, 214
- General Utility Functions, 180, 184
- List, 209
- MIME, 188
- Result, 202

Runtime, 232

Step Result, 202

Stream, 188

String, 219

F**Features,**

- new, 9

File System Triggers,

- check, 154
- enabled, 154
- expires, 154
- of file or directory, 154
- overview of, 154
- parameters, 154
- polling interval, 154
- start, 154
- time zone, 154
- wait N seconds for settle, 154

Filter by,

- Date, 104
- Instance ID, 104
- Object path, 104
- Severity, 104

FlowForce Server,

- changing the language of, 400
- command line interface, 383
- localization of, 392
- migrating to latest version of, 397

FTP, 162

- credentials, 164

FTPS, 162**G****General Utility Functions,**

- sleep-for, 184

Global Resources,

- using in FlowForce Server, 423

H

Home page,

- Active Timers, 85
- jobs info, 85
- Running Jobs, 85
- Statistics, 85

HTTP, 162

HTTP Triggers,

- check, 155
- enabled, 155
- expires, 155
- of URI, 155
- overview of, 155
- parameters, 155
- polling interval, 155
- start, 155
- time zone, 155
- wait N seconds for settle, 155

I

Input Parameters,

- add, 130
- built-in, 130
- default, 130
- description, 130
- fields, 130
- function parameters, 130
- job input parameters, 130
- name, 130
- remove, 130
- triggerfile, 130
- type, 130

Installation and Configuration, 20

- Installation of FlowForce Server, 21
- Licensing of FlowForce Server, 21
- Setup of FlowForce Server, 21
- Setup on Linux, 30
- Setup on macOS, 37
- Setup on Windows, 21

Instance,

- collapse, 106

- expand, 106

- log, 106

Introduction, 8

- APPDATADIR, 18
- application data directory, 18
- basic concepts, 15
- cache job result, 15
- credentials, 15
- execution result, 15
- expression functions, 15
- functions, 15
- how it works, 12
- important paths, 18
- input parameters, 15
- installation directory, 18
- INSTALLDIR, 18
- job, 15
- list of FlowForce instances, 18
- localize FlowForce Server, 12
- overview, 12
- queue settings, 15
- settings, 15
- step functions, 15
- steps, 15
- terms, 15
- triggers, 15
- web service, 15

item,

- as FlowForce data type, 177

J

Java,

- configuration, 406

JDBC,

- database connections, 406

Job statuses,

- Aborted, 89
- Aborting, 89
- Created, 89
- Failed, 89
- Finished, 89
- Finished successfully, 89
- Interrupted, 89
- Lost connection, 89
- Recovering, 89

Job statuses,

- Running, 89
- Starting, 89
- Superseded, 89
- Synchronizing, 89
- Untracked, 89
- Waiting, 89
- Waiting for slot, 89

Jobs, 162

- active timers, 85
- active triggers and services, 85
- all, 85
- as Web services, 158
- cache, 148
- cache results, 148
- caching results of, 347
- configuration, 128, 130, 132, 133, 135, 136, 140, 145
- copy, 128
- create, 128
- creating from MapForce mapping, 308
- creating from StyleVision transformations, 351
- credentials, 164
- duplicate, 128
- execution result, 145
- execution steps, 132, 133, 135, 136, 140
- global queues, 166
- input parameters, 130
- job result, 145
- local queues, 166
- log, 104
- monitor execution, 85
- prerequisites, 128
- queues, 166
- recently finished, 85
- result, 145
- return type, 145
- running, 85
- starting, 85
- statistics, 85
- statuses, 89
- step result, 145
- stop, 85

K**Key/value pairs,**

- passing to XSLT sheets, 366

L**LDAP,**

- integration with FlowForce Server, 119

Linux,

- trusting server certificates on, 55

Log,

- copy, 104
- export, 104, 106
- instance, 106
- log view, 104
- reducing the size of, 122
- settings, 122
- table, 104
- view, 104, 106

M**Mac,**

- trusting server certificates on, 55

MapForce Server,

- calling from a job, 371
- integration with, 405, 417

O**ODBC,**

- database connections, 406

P**PDF files,**

- generating, 371

Permissions,

- list of, 92
- overview of, 92

Protocols, 162

R

RaptorXML functions,

creating jobs with, 359, 361

RaptorXML Server,

integration with, 433

Resources,

creating, 423

editing, 423

result,

as FlowForce data type, 177

results,

as FlowForce data type, 177

S

Setup on Linux,

Assign a license to FlowForce Server on Linux, 34

Installation, 31

Installing LicenseServer on Linux, 32

License for FlowForce Server, 34

LicenseServer versions, 32

Licensing FlowForce Server on Linux, 33

Register FlowForce Server with LicenseServer on Linux, 34

Setup page, 36

Start FlowForce Server on Linux, 33

Start LicenseServer on Linux, 33

Setup on macOS,

Assigning a license to FlowForce Server on macOS, 41

Installation on macOS, 38

Installing LicenseServer on macOS, 39

LicenseServer versions, 39

Licensing FlowForce Server on macOS, 40

Register FlowForce Server with LicenseServer on macOS, 40

Setup page, 42

Start FlowForce Server on macOS, 40

Start LicenseServer on macOS, 40

Setup on Windows, 21

Altova ServiceController, 27

Assign a license to FlowForce Server on Windows, 28

Deinstallation, 22

Installation, 22

Installation on Windows Server Core, 24

Installing LicenseServer on Windows, 25

LicenseServer versions, 25

Licensing FlowForce Server on Windows, 26

Register FlowForce Server with LicenseServer on Windows, 28

Start FlowForce Server on Windows, 27

Start LicenseServer on Windows, 27

Uninstalling, 22

Severity types,

Error, 104

Info, 104

Verbose, 104

Warning, 104

SFTP, 162

SQLite,

as job data source or target, 406

Steps,

buttons, 132

Choose, 132, 133

collapse, 132

conditional, 133

create, 132

data types, 145

Error/Success Handling, 132, 136

example of using in jobs, 316

Execution, 132

expand, 132

For-Each, 132, 135

handle errors, 136

Postpone, 140

postponed, 132, 140

step result in other steps, 145

types, 132

StyleVision Server,

calling from a job, 371

integration with, 405, 417

System Functions, 235

/system, 235

/system/abort, 235

/system/compute, 237

/system/compute-string, 239

/system/create-file, 240

/system/filesystem, 241

/system/filesystem/copy, 241

/system/filesystem/delete, 242

/system/filesystem/mkdir, 243

/system/filesystem/move, 244

/system/filesystem/rmdir, 245

System Functions, 235

- /system/ftp, 245
- /system/ftp/delete, 246
- /system/ftp/delete-wildcard, 249
- /system/ftp/list, 252
- /system/ftp/mkdir, 255
- /system/ftp/move, 258
- /system/ftp/retrieve, 261
- /system/ftp/retrieve-wildcard, 264
- /system/ftp/rmdir, 267
- /system/ftp/store, 270
- /system/ftp/store-wildcard, 275
- /system/mail, 278
- /system/mail/send, 279
- /system/mail/send-mime, 280
- /system/maintenance, 284
- /system/maintenance/archive-log, 284
- /system/maintenance/cleanup-files, 284
- /system/maintenance/truncate-log, 285
- /system/shell, 285
- /system/shell/commandline, 285
- triggerfile, 245
- wildcards, 245

T**Timer Triggers,**

- enabled, 152
- expires, 152
- overview of, 152
- parameters, 152
- repeat, 152
- run, 152
- start, 152
- time zone, 152

Tool files,

- options, 436

Triggers,

- add, 151
- create, 151
- delete, 151
- disable, 151
- duplicate, 151
- enable, 151
- file system, 154
- HTTP, 155

- manage, 151
- re-enable, 151
- time, 152
- timer, 152
- triggerfile parameter, 151
- types, 151
- watch, 154

U**UNC,**

- syntax in paths, 128

Upgrade FlowForce Server, 44**W****Web Services,**

- authentication, 158
- configure, 158
- exposing jobs as, 331
- parameters, 158

Web UI Reference,

- Administration, 81
- Configuraiton, 81
- Help, 81
- Home, 81
- Log, 81
- log in, 81
- log on, 81
- overview, 81

Windows,

- network paths on, 235
- trusting server certificates on, 55

Windows network paths, 128**Working directory,**

- usage, 417