# Contents

*Using This Book* ........................................................................................................... vii

**Chapter 1 / SAS Event Stream Manager Overview** ....................................................... 1
- What Is SAS Event Stream Manager? ................................................................. 2
- Key Concepts ......................................................................................................... 2
- Log On to SAS Event Stream Manager ............................................................... 5
- Understanding the User Interface ........................................................................ 6
- Overview of Managing Assets ............................................................................... 13
- Production Assets ................................................................................................ 14

**Chapter 2 / Working with Deployments** .................................................................... 17
- Create a Deployment ............................................................................................. 17
- Edit a Deployment ................................................................................................ 18
- Delete a Deployment ............................................................................................. 19

**Chapter 3 / Working with ESP Servers** ................................................................... 21
- Agent ...................................................................................................................... 21
- Add ESP Servers .................................................................................................. 22
- Connecting Directly to an ESP Server .................................................................. 24
- Edit an ESP Server ............................................................................................... 25
- Remove ESP Servers from a Deployment ............................................................ 26
- Remove an ESP Server from SAS Event Stream Manager ................................. 27

**Chapter 4 / Working with Projects** ........................................................................ 29
- Upload a Project ..................................................................................................... 29
- View a Project ........................................................................................................ 31
- Change the Production Status of a Project .......................................................... 37
- Upload a New Version of a Project ....................................................................... 37
- Delete a Project ...................................................................................................... 38

**Chapter 5 / Working with Filters** ........................................................................... 41
- Filters ...................................................................................................................... 41
Contents

Working with Filters for a Specific Deployment ........................................ 42
Managing Filters ......................................................................................... 44
Filter Expression Syntax ........................................................................... 46

Chapter 6 / Working with Job Templates ...................................................... 49
Job Template .............................................................................................. 50
Create a Job Template ................................................................................ 51
Upload a Job Template .............................................................................. 52
Edit a Job Template .................................................................................... 53
Change the Production Status of a Job Template ....................................... 55
View a Job Template ................................................................................... 56
Deploy a Job Template ............................................................................... 58
Download a Job Template ......................................................................... 59
Delete a Job Template ............................................................................... 60

Chapter 7 / Example Job Templates ............................................................... 63
Overview of Example Job Templates ........................................................... 63
Example Job Template: Stock Trade .......................................................... 64
Example Job Template: Stop a Project ....................................................... 74

Chapter 8 / Job Template Content ................................................................. 77
Overview of Job Template Content ............................................................. 78
job-template ............................................................................................... 78
localization-strings .................................................................................... 81
parameters .................................................................................................. 83
enumerations .............................................................................................. 93
server-filters .............................................................................................. 95
instructions .................................................................................................. 97
failure-instructions ..................................................................................... 111
initialization ............................................................................................... 114

Chapter 9 / Monitoring a Deployment .......................................................... 117
Monitor Deployments .................................................................................. 118
Monitor a Running Project ......................................................................... 120
View Sample Data ..................................................................................... 121
Update SAS Micro Analytic Service Modules ............................................ 123
Monitor ESP Servers ................................................................. 128
Monitor Jobs ........................................................................... 131
Delete a Job ............................................................................ 134
Rerun a Job .............................................................................. 134
Stop a Running Job ................................................................. 136
Stop a Running Project ............................................................ 136

Chapter 10 / Metering Servers ...................................................... 139
Metering .................................................................................. 139
Add a Metering Server ............................................................ 139
Monitor a Metering Server ....................................................... 140

Chapter 11 / Stock Trade Example ............................................... 143
Overview of the Stock Trade Example ....................................... 143
The Stock Trade Project .......................................................... 144
Stock Trade Example Steps ..................................................... 148

Recommended Reading ............................................................ 159
Glossary ................................................................................. 161
Contents
Using This Book

Audience

The target audience for *SAS Event Stream Manager: User’s Guide* is SAS Event Stream Processing application administrators. SAS Event Stream Manager enables application administrators to effectively manage the SAS Event Stream Processing environment, including deploying SAS Event Stream Processing projects to ESP servers, monitoring deployments, and managing change.

Requirements for Solution Access and Use

Here are the requirements for accessing and using SAS Event Stream Manager:

- You have a user ID and password for logging on to SAS Event Stream Manager.
- A supported web browser has been installed.

  **Note:** For detailed information about supported browsers, in SAS Event Stream Manager click ![Help Icon](image) and then click **About**. The About window appears. Click **Supported Browsers and Platforms** to view supported browsers. SAS Event Stream Manager requires the use of cookies to maintain the session state.

- Your screen has a minimum screen resolution of 1,280 x 1,024.
- JavaScript has been enabled in your browser.
Purpose of This Document

This document provides usage and operational assistance for the typical user of SAS Event Stream Manager.

Note: Each deployment of SAS Event Stream Manager is configured to meet the needs of a specific customer. The information and windows, including the color and layout themes, shown in this document are intended to show the functionality of SAS Event Stream Manager and are only examples.
# SAS Event Stream Manager Overview

**What Is SAS Event Stream Manager?** ..................................................... 2

**Key Concepts** ................................................................. 2
- Asset ........................................................................... 2
- Deployment .................................................................. 3
- Project .......................................................................... 3
- ESP Server .................................................................... 4
- Job Template ................................................................. 4
- Job .............................................................................. 5

**Log On to SAS Event Stream Manager** .................................................. 5

**Understanding the User Interface** ......................................................... 6
- Pages ............................................................................ 6
- Panes ........................................................................... 7
- Tiles ............................................................................ 8
- Windows ....................................................................... 9
- Toolbars ....................................................................... 10
- Arrange Information in Tables ............................................ 12

**Overview of Managing Assets** ............................................................... 13

**Production Assets** ........................................................................... 14
What Is SAS Event Stream Manager?

SAS Event Stream Manager is a web-based client that enables you to manage your SAS Event Stream Processing environment.

You can use SAS Event Stream Manager to perform the following tasks:

- deploy SAS Event Stream Processing projects into production environments and test environments
- view the component parts of each deployment
- monitor the health of your deployments
- administer your deployments and manage change
- monitor your SAS Event Stream Processing metering servers

You can use SAS Event Stream Processing Studio to create the projects that you deploy to ESP servers using SAS Event Stream Manager.

Key Concepts

This topic introduces the key concepts in SAS Event Stream Manager.

Asset

An asset is any identifiable part of a SAS Event Stream Processing deployment that SAS Event Stream Manager can monitor or act upon. Examples of assets include projects and ESP servers.
**Deployment**

A *deployment* is a logical grouping of assets as a single unit for lifecycle management and monitoring in SAS Event Stream Manager. That is, a deployment is a group of assets that together does something useful.

SAS Event Stream Manager tracks changes to assets that it manages. Here are examples of details recorded by SAS Event Stream Manager for most assets:

- creation timestamp
- last update timestamp
- created by (user ID or name)
- last updated by (user ID or name)
- version
- comments

**Project**

A *project* is a data processing model that can be executed on an ESP server.

A project is stored as an XML document. You must use SAS Event Stream Processing Studio to create projects. If you create the project’s XML code manually, you must upload it to SAS Event Stream Processing Studio before you can use it in SAS Event Stream Manager.

Here are examples of project metadata captured by SAS Event Stream Manager:

- version
- project repository ID
- whether the project is a production asset (for more information, see “Production Assets” on page 14)
ESP Server

In SAS Event Stream Processing, the ESP server is an engine-executable program that instantiates and executes SAS Event Stream Processing projects.

An ESP server and the equipment on which it is running can have different states. For example, an ESP server might not be running even though the server on which it is located is available.

Here are examples of ESP server metadata captured by SAS Event Stream Manager:

- type
- usage type
- version number, build number, or both
- equipment on which the ESP server is running

Job Template

A job template is an XML document that contains a set of instructions to create a job. That is, a job template outlines the steps required to deploy a project on an ESP server.

You can create a job template by using an editor provided within SAS Event Stream Manager. A job template contains the following high-level elements:

- localizations
- parameters
- enumerations
- initializations
- instructions
- failure instructions

When you deploy a job template, a job is created.
Job

A job is a set of tasks to be executed by SAS Event Stream Manager on various assets associated with one or more deployments. A job is always created from a job template.

A job collates all assets affected by the job template. A job includes the tasks to be performed on the assets and the status of each task.

SAS Event Stream Manager records the following job metadata for a deployment:

- job ID
- name
- description
- tags

Projects can include placeholders where values are injected to customize project behavior at run time. These placeholder values can be recorded and displayed as part of the details of a job.

Jobs cannot be changed after they are created. If you want to alter a job, you must create a new job instead.

Log On to SAS Event Stream Manager

1. Open the following URL:

   https://host/SASEventStreamManager

   The host is the system on which SAS Event Stream Manager is installed.

   The Sign In to SAS window appears.

2. Enter your user ID and password, and click Sign In.
Understanding the User Interface

Pages

A page is the highest level container in the user interface. All other user interface elements are contained within the confines of a page.

When you access SAS Event Stream Manager, the Deployments page appears.

Figure 1.1  The Deployments Page with Two Deployments

SAS Event Stream Manager contains the following main pages:

- The Deployments page enables you to create deployments and then monitor and manage them.
- The Log page displays details of currently running jobs and historical jobs.
The **ESP Servers** page enables you to view available ESP servers that SAS Event Stream Manager has detected automatically as well as to connect directly to other ESP servers. ESP servers that already belong to a deployment are not displayed.

The **Projects** page enables you to upload projects for inclusion in job templates.

The **Job Templates** page enables you to create and upload job templates, which you can then deploy to create running jobs.

The **Filters** page enables you to manage filters that are available in SAS Event Stream Manager.

The **Metering** page enables you to monitor the metering servers that track usage data.

For more information about the main tasks that you perform using SAS Event Stream Manager, see “Overview of Managing Assets” on page 13.

**Panes**

SAS Event Stream Manager pages contain *panes*. The following figure shows the **Job Templates** page, which contains a bottom pane with two *tiles*: Identification and Version.
Figure 1.2  Example of a Page with a Pane

To resize a pane, drag the border upward or downward.

To hide a pane, click 📜. To display it again, click 📜.

Tiles

A tile is a self-contained block of information that resides within a pane or sometimes directly on a page. The same tile can appear on several pages. For example, the Running Projects tile appears on more than one page. (For more information, see “Monitor a Running Project” on page 120.)
Figure 1.3  Example of a Tile

ESP Servers

<table>
<thead>
<tr>
<th>Health</th>
<th>Name</th>
<th>Tags</th>
<th>Host</th>
<th>HTTP Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔</td>
<td>ESP_Server_1</td>
<td></td>
<td>server1.example.com</td>
<td>2002</td>
</tr>
<tr>
<td>✔</td>
<td>ESP_Server_2</td>
<td>primary</td>
<td>server2.example.com</td>
<td>3002</td>
</tr>
</tbody>
</table>

Windows

A *window* is a floating user interface element that often appears as a result of a user action. Windows generally provide a means by which to perform an action. Closing a window returns you to the page from which the window was launched. The following figure shows a window that can create a new deployment in SAS Event Stream Manager.
There are three main toolbars in SAS Event Stream Manager, as shown in the following figure. For information about each toolbar, see the subsequent table.
### Figure 1.5  SAS Event Stream Manager Toolbars

<table>
<thead>
<tr>
<th>Item Number</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| 1           | Application bar |   - Displays your display name or user ID.  
               |                                               | Note: If you have not set a display name, your user ID is displayed by default.                |
|             |                 |                                               | Provides access to Help and product information.                                             |
|             |                 |                                               | Enables you to sign out of SAS Event Stream Manager.                                       |
| 2           | Menu bar        |   - Provides access to the main SAS Event Stream Manager pages: **Deployments**, **Log**, **ESP Servers**, **Projects**, **Filters**, **Job Templates**, and **Metering**. |
|             |                 |                                               | Provides access to each deployment, ESP server, project, job template, or job that is currently open. The navigation overflow menu button displays the total number of these pages that are currently open, for example, ![Help icon](https://example.com/help). |
| 3           | Toolbar or tabs |   - Includes buttons or tabs specific to the open page. For example, the preceding figure shows toolbar buttons on the **Job Templates** page. The following figure shows tabs on a page for an open project. |
Arrange Information in Tables

Sometimes a large amount of information is displayed in tables. To make it easier to work with a large amount of information, you can arrange this information in different ways.

You can sort the contents of columns by ascending or descending order. To do this, click the heading of the column that you want to sort.

You can create filter criteria by which to display only a subset of information for a column. To create filter criteria, click \( \text{Filter} \) for the column that you want to apply filter criteria to, select \( \text{Filter} \), and enter your filter criteria.

You can configure the columns that you want to display. To do this, click \( \text{Columns} \) in any column, select \( \text{Columns} \), and deselect the columns that you do not want to appear.

You can re-order columns. To do this, click and hold the column heading and drag it to a different location.

In some tables, you can group information by column. Such tables have a horizontal bar at the top of the table, with the text \( \text{Drag a column heading here to group by that column} \). To group information by column, drag a column heading to the bar. If required,
you can drag additional columns to the bar. In the following example, information about the **Deployments** page has been grouped by the Production column:

*Figure 1.7  Example of Grouping Information by Column*

---

**Overview of Managing Assets**

Before SAS Event Stream Manager can interact with assets, you must do the following:

1. **Create a deployment.**
   
   For more information, see “Create a Deployment” on page 17.

2. **Associate an ESP server with the deployment.**
   
   For more information, see “Add ESP Servers” on page 22.

3. **Upload a project.**
   
   For more information, see “Upload a Project” on page 29.

4. **Create or upload a job template.**
   
   For more information about creating a new job template, see “Create a Job Template” on page 51. For more information about uploading a job template, see “Upload a Job Template” on page 52.
5  Deploy a job template.

   For more information, see “Deploy a Job Template” on page 58.

For more information about the meaning of the terms mentioned here, see “Key Concepts” on page 2.

This guide includes an end-to-end example of how to use SAS Event Stream Manager to complete the tasks listed above. For more information, see “Overview of the Stock Trade Example” on page 143.

Production Assets

You can mark certain assets as production assets to prevent assets meant for testing from accidentally being used for production (that is, being used in a live environment). The following assets can be marked as production assets:

- Deployments: You can mark a deployment as a production asset when you create it. You cannot change this status later. That is, you cannot change an existing non-production deployment to a production deployment. You must create a new deployment instead. The fixed status ensures that an existing deployment cannot accidentally be moved from test to production.

- Projects: You can mark a project as a production asset when you create it. You can change this status later: you can change a production project to a non-production project, or you can change a non-production project to a production project.

- Job templates: You can mark a job template as a production asset when you create it. You can change this status later: you can change a production job template to a non-production job template, or you can change a non-production job template to a production job template.

You can deploy a non-production job template only against a non-production deployment. In such a situation, SAS Event Stream Manager permits you to select either a production project or a non-production project to deploy. This choice enables you to test your current production projects or to test non-production projects to assess whether they are suitable for being marked as production projects in the future.
You can deploy a production job template against a production deployment or against a non-production deployment:

- If you deploy a production job template against a production deployment, SAS Event Stream Manager permits you to select only a production project to deploy.
- If you deploy a production job template against a non-production deployment, SAS Event Stream Manager permits you to select either a production project or a non-production project to deploy.

The following diagram illustrates this information:

*Figure 1.8 Options for Deploying Job Templates*
Working with Deployments

Create a Deployment ................................................................. 17
Edit a Deployment ................................................................. 18
Delete a Deployment ............................................................... 19

Create a Deployment

1 On the Deployments page, click .

The Deployment Properties window appears.

2 In the Name field, enter a unique name for the deployment.

3 In the Description field, enter a description for the deployment. For example, you can enter the purpose of the deployment to allow users to differentiate between deployments with similar names.

4 In the Tags field, you can attribute one or more tags to the deployment.

Tags can be used to group and filter deployments. Tags are single-term descriptors for the deployment. Tags cannot contain spaces. Duplicate tags are not permitted on a single deployment.

5 If you want to create a production deployment, select the Production deployment check box. For more information, see “Production Assets” on page 14.
6 Click **OK**.

Your deployment appears on a new page. The following figure shows an example:

![Deployment Interface](image)

**Identification**

- **ID:** 73a9f6b0-7e34-469a-a85f-11fb82986d89
- **Name:** Warehouse Temperature
- **Production deployment:** False
- **Description:** A deployment for monitoring warehouse temperatures.

**Tags:**

- temperature

---

**Edit a Deployment**

1 Open the deployment if it is not already open: on the **Deployments** page, select the deployment that you want to open and click **.**

A separate page that displays the deployment opens.
2 Edit the Name, Description, and Tags fields in the Identification tile as required. For more information about these fields, see “Create a Deployment” on page 17.

Note: You cannot edit the ID and Production fields. Because the value of the Production field is fixed, you cannot change an existing non-production deployment to a production deployment. You must create a new deployment instead. The fixed value of the Production field ensures that a deployment cannot accidentally be moved from test to production.

3 If required, you can add or remove ESP servers, or filter ESP servers associated with the deployment:

- For more information about adding ESP servers, see “Add ESP Servers” on page 22.
- For more information about removing ESP servers, see “Remove ESP Servers from a Deployment” on page 26.
- For more information about filtering ESP servers, see “Working with Filters for a Specific Deployment” on page 42.

Delete a Deployment

You can delete a deployment if no ESP servers are associated with it. For more information, see “Remove ESP Servers from a Deployment” on page 26.

To delete a deployment:

1 On the Deployments page, select the deployment that you want to delete.

2 Click ![delete]

   The Delete Deployment window appears.

3 Click Delete.
Agent

An agent is a small executable program that SAS Event Stream Manager runs on equipment alongside ESP servers. (For an introduction to ESP servers, see “ESP Server” on page 4.)

Agents act as communication gateways between SAS Event Stream Manager and ESP servers. Agents relay operational metrics from ESP servers to SAS Event Stream Manager and perform actions on the ESP servers in response to the commands received from SAS Event Stream Manager.
SAS Event Stream Manager can also connect directly to an ESP server without using an agent. Connecting directly might be relevant when the agent cannot be installed on the same device as the ESP server (for example, due to disk space constraints).

Agents are typically installed and configured when SAS Event Stream Manager is deployed. One agent monitors all ESP servers on the machine on which the agent is installed. For more information about configuring an agent, see “Configure and Restart the SAS Event Stream Manager Agent” in SAS Viya for Linux: Deployment Guide.

Add ESP Servers

You can associate an ESP server with a deployment only if SAS Event Stream Manager is already aware of the ESP server. SAS Event Stream Manager detects some ESP servers automatically and communicates with these ESP servers by using agents. You can also connect directly to an ESP server (without using an agent). In this case, SAS Event Stream Manager becomes aware of the ESP server only after you have made the connection. For more information, see “Connecting Directly to an ESP Server” on page 24.

Each ESP server can belong to only one deployment. If you want to add an ESP server that already belongs to another deployment, you must first remove that ESP server from its original deployment. For more information, see “Remove ESP Servers from a Deployment” on page 26.

Each deployment can contain several ESP servers.

To add ESP servers to a deployment:

1. Open the deployment if it is not already open: on the Deployments page, select the deployment that you want to open and click .

   A separate page that displays the deployment opens.

2. Click .

   The Add and Remove ESP Servers window appears.
3 Move the required ESP servers from the **Available servers** table to the **Selected servers** table, using the buttons between the two tables:

- To move a single ESP server, select the ESP server in the **Available servers** table and click ➔.
- To move all ESP servers that are displayed in the **Available servers** table, click ➔.

4 Click **OK**.

The ESP servers appear in the table on the deployment page. The following figure shows an example:
Connecting Directly to an ESP Server

SAS Event Stream Manager detects some ESP servers automatically. SAS Event Stream Manager communicates with these ESP servers using agents. You can also connect directly to an ESP server without using an agent. Connecting directly might be relevant when the agent cannot be installed on the same device as the ESP server (for example, due to disk space constraints).

To connect directly to an ESP server:

1. On the ESP Servers page, click New.

   The ESP Server Properties window appears.

2. In the Name field, enter a name for the ESP server.

3. In the Host field, enter the host name of the server containing the ESP server.

4. In the HTTP port field, replace the default value with the port number used for HTTP administration requests and for the HTTP publish/subscribe server. (This port is specified by the -http argument of the $DFESP_HOME/bin/dfesp_xml_server command.)

5. In the Tags field, you can attribute one or more tags to the ESP server.

   Tags can be used to group and filter ESP servers. Tags are single-term descriptors for the ESP server. Tags cannot contain spaces. Duplicate tags are not permitted on a single ESP server.

6. If required, change the setting for the Authentication field:
   - None: This is the default option.
   - Authenticate using an OAuth token: This option is relevant only if the ESP server is configured to require authorization. If you select this option, an additional field is displayed where you must enter the OAuth token.
- **Authenticate using a user name and password**: This option is relevant only if the ESP server is configured to require authorization. If you select this option, additional fields are displayed where you must enter the user name and password.

7 If required, select the **Connect using SSL** check box. Selecting this check box is relevant only if the ESP server is configured to require SSL encryption.

8 Click **OK**.

The ESP server is displayed on the **ESP Servers** page, along with your other ESP servers. The **ESP Servers** page does not indicate which ESP servers were detected automatically and which ones you connected to directly.

You can now associate the ESP server with a deployment. For more information, see “Add ESP Servers” on page 22.

---

**Edit an ESP Server**

1 Open the ESP server if it is not already open: on the **ESP Servers** page, select the ESP server that you want to open and click **新征程**.

   A separate page that displays the ESP server opens.

2 Edit the fields in the **Identification** tile as required:

   - **Name**: If the ESP server was detected automatically by SAS Event Stream Manager, you cannot edit the **Name** field.

   - **Tags**: You can attribute one or more tags to the ESP server. Tags can be used to group and filter ESP servers. **Tags** are single-term descriptors for the ESP server. Tags cannot contain spaces. Duplicate tags are not permitted on a single ESP server.

3 Click **新征程**.
Remove ESP Servers from a Deployment

You can remove an ESP server from a deployment if the ESP server is not currently running any projects. You must stop and unload any projects that were previously running on the ESP server. For more information, see “Stop a Running Job” on page 136.

Remove an ESP Server from a Deployment

To remove an ESP server from a deployment:

1. Open the deployment if it is not already open: on the Deployments page, select the deployment that you want to open and click .

   A separate page that displays the deployment opens.

2. Right-click the ESP server that you want to remove and select Remove ESP server.

   The ESP server is removed from the deployment. If required, you can now add it to another deployment.

Remove All ESP Servers from a Deployment

To remove all ESP servers from a deployment:

1. Open the deployment if it is not already open: on the Deployments page, select the deployment that you want to open and click .

   A separate page that displays the deployment opens.

2. Click .

   The Add and Remove ESP Servers window appears.

3. Click between the Available servers table and the Selected servers table.
All the ESP servers are removed from the deployment. If required, you can now add them to another deployment.

---

**Remove an ESP Server from SAS Event Stream Manager**

You can remove an ESP server from SAS Event Stream Manager if the ESP server is not associated with a deployment (for more information, see “Remove ESP Servers from a Deployment” on page 26) or with an agent. Removing an ESP server from SAS Event Stream Manager means that the ESP server will not be detected by SAS Event Stream Manager. The ESP server itself continues to exist.

To remove an ESP server from SAS Event Stream Manager:

1. On the **ESP Servers** page, select the ESP server that you want to remove and click 
   - The Delete ESP Server window appears.

2. Click **OK**.

SAS Event Stream Manager no longer detects the ESP server that you selected.
You can use SAS Event Stream Processing Studio to create projects using a visual interface. If you create the project’s XML code manually, you must upload it to SAS Event Stream Processing Studio before you can use it in SAS Event Stream Manager.
Before SAS Event Stream Manager accepts an uploaded project, it validates the project’s content against an active ESP server. Therefore, an ESP server must be available to SAS Event Stream Manager before you attempt to upload a project.

**Note:** Project XML files uploaded to SAS Event Stream Processing Studio must be encoded in UTF-8 format. Uploading project XML files that are not encoded in UTF-8 format can display invalid characters in SAS Event Stream Processing Studio.

To upload a project:

1. On the **Projects** page, click ![Upload](https://example.com/upload). The Upload Project window appears.
2. Click **Browse** and navigate to the location of your project.
   
   **Note:** Projects must be in XML format.
3. If you want this project to be a production project, select the **Production project** check box. For more information, see “Production Assets” on page 14.
4. In the **Project** field, enter a name for the project.
5. In the **Description** field, enter a description of the project. For example, you can enter the purpose of the project to allow users to differentiate between projects with similar names.
   
   If the project that you are uploading contains a project description, the **Description** field is automatically populated with that text. The project description must be contained in a `description` element that is located directly within the `project` element.
6. In the **Tags** field, you can attribute one or more tags to the project.
   
   Tags can be used to group and filter projects. Tags are single-term descriptors for the project. Tags cannot contain spaces. Duplicate tags are not permitted on a single project.
In the **Version notes** field, you can add notes to further describe the project. Adding notes can be useful if you want to track changes between versions of the same project within SAS Event Stream Manager.

Click **OK** to upload the project. When successfully uploaded, the project appears on the **Projects** page. The following figure shows an example:

![Projects page example](image)

**Note:** If the project fails validation, an error message appears. You can use the information in the message to correct the issue and then upload a revised project.

When you upload a new version of a project that was updated in SAS Event Stream Processing Studio, the project's major version is updated in SAS Event Stream Manager. For example, SAS Event Stream Processing updates the version in the project XML file from 1.0 to 2.0 and this is displayed in SAS Event Stream Manager.

However, if you update the project XML file manually, that is, not in SAS Event Stream Processing Studio, the version is always set to 1.0 in SAS Event Stream Manager. This happens even if the major version and minor version are set manually in the XML file. To ensure that the version increases as expected, update the project in SAS Event Stream Processing Studio.

---

**View a Project**

To view detailed information about a project, select the project on the **Projects** page and click **View a Project**. A page with five tabs appears.
The Details Tab

The **Details** tab displays general information relating to the project entered when the current version of the project was uploaded. The following figure shows an example:

*Figure 4.1  The Details Tab*

---

The XML Tab

The **XML** tab displays a read-only view of the XML used to construct the model. You can collapse XML elements to restrict your view to the elements that you are interested in. The following figure shows an example of the **XML** tab:
The **Diagram Tab**

The **Diagram** tab displays a graphical representation of the project’s windows and edges. Each window can display icons that represent its state. For more information, see “Window Icons” in *SAS Event Stream Processing: Using SAS Event Stream Processing Studio*. The following figure shows an example of the **Diagram** tab:
Figure 4.3 The Diagram Tab

You can use the buttons on the toolbar to zoom in and out of the view, or to zoom to fit the view.

The Versions Tab

The Versions tab displays the current version and any previous versions of the project. The following figure shows an example:
To open a previous version, right-click it and select **Open project version**. A separate page appears where you can review the metadata and XML content relating to that version.

The major version is updated when you upload a new version of the project that was updated in SAS Event Stream Processing Studio. For example, when the project is updated in SAS Event Stream Processing Studio, the version in the project XML file is updated from 1.0 to 2.0. For more information, see “Upload a New Version of a Project” on page 37.

**Note:** If the project XML file is updated manually, that is, not in SAS Event Stream Processing Studio, the version is always set to 1.0 in SAS Event Stream Manager. This happens even if the major version and minor version are set manually in the XML file. To ensure that the version increases as expected, update the project in SAS Event Stream Processing Studio.
The minor version is updated when you update a project that references a model that is stored in the SAS Model Manager common model repository, and a new champion model is declared in SAS Model Manager. For example, SAS Event Stream Manager updates the project from version 1.0 to 1.1. For more information, see “Update SAS Micro Analytic Service Modules” on page 123.

**The Files Tab**

The **Files** tab displays analytics model files that are loaded from the SAS Model Manager common model repository and are executed through SAS Micro Analytic Service Modules and the Calculate window. The following figure shows an example:
Change the Production Status of a Project

Certain assets can be marked as production assets. For more information, see “Production Assets” on page 14.

To change the production status of a project, on the Projects page, right-click the project and select **Toggle production**.

The Production column on the Projects page changes to reflect your choice.

Upload a New Version of a Project

You cannot use SAS Event Stream Manager to edit a project, but you can upload an updated version of the XML file. When you upload a new version of the file, you can also enter new project identification details, such as the project name or tags.

1. On the Projects page, select the project and click ![Edit Project](image).

   The project opens on a new page.

2. Click ![Upload New Version](image).

   The Upload Project window appears.

3. Fill out the fields as required and click **OK**.

   The fields in the window are the same as when you upload a new project. For more information, see “Upload a Project” on page 29.

   Information that you enter in the fields replaces previously recorded information for the project. For example, if you enter a new name in the Project field, the value that you provide becomes the name of the project. Any tags that you enter in the Tags field replace previous tags for this project rather than adding to the existing set of
tags. If you leave the **Tags** field empty, the new version of your project will have no tags.

When you upload a new version of a project that was updated in SAS Event Stream Processing Studio, the project’s major version is updated in SAS Event Stream Manager. For example, SAS Event Stream Processing updates the version in the project XML file from 1.0 to 2.0 and this is displayed in SAS Event Stream Manager.

However, if you update the project XML file manually, that is, not in SAS Event Stream Processing Studio, the version is always set to 1.0 in SAS Event Stream Manager. This happens even if the major version and minor version are set manually in the XML file. To ensure that the version increases as expected, update the project in SAS Event Stream Processing Studio.

---

### Delete a Project

If you have uploaded multiple versions of a project, you can delete the entire project or a specific version of the project. If there is only one version of the project, you can use any of the deletion methods described here.

---

### Delete the Latest Version of a Project

1. On the **Projects** page, select the project whose latest version you want to delete.

2. Click ![delete](delete.png).

   The Delete Latest Version window appears.

3. Click **OK** to confirm the deletion.

   The latest version of the project is deleted. If there are any remaining earlier versions, the project is still displayed on the **Projects** page.
Delete an Earlier Version of a Project

1 On the Projects page, right-click the project and select Open project.

2 Click the Versions tab.

3 Right-click the version that you want to delete and select Delete project version.
   The Delete Project window appears.

4 Click Delete to confirm the deletion.

Delete All Versions of a Project

1 On the Projects page, right-click the project and select Open project.

2 Click the Versions tab.

3 Click .
   The Delete Project window appears.

4 Click Delete to confirm the deletion.
Filters

SAS Event Stream Manager enables you to search for ESP servers that match certain criteria and save those criteria as a filter that you can reference in job templates. For example, you might want to deploy a project to ESP servers that have the tag `primary`. If the number of ESP servers that have this tag changes, a filter that references this tag still finds any matching ESP servers.

A filter contains a filter expression. The filter expression specifies the search criteria for ESP servers.
You can specify that a filter is available only to a specific deployment or that a filter is available for use with all deployments. For example, this means that the filters available to a deployment called Deployment1 include filters specific to that deployment as well as filters available to all deployments. However, filters specific to another deployment called Deployment2 are not available to Deployment1.

In addition, the SAS Event Stream Manager user interface can show the result of applying a filter. This means that for a given deployment, the user interface lists the ESP servers that match the filter criteria. This functionality is different from applying a filter. A filter is applied when a job template that references the filter is deployed.

You can reference filters from the `server-filters` and `server-filter-selector` elements in job templates. You can also specify a filter expression directly in the `server-filters` element. For more information about the `server-filters` element, see “server-filters” on page 95. For more information about the `server-filter-selector` element, see “server-filter-selector” on page 92.

In addition to creating filters yourself, there is one situation where SAS Event Stream Manager constructs a filter expression that you can then save as a filter. If a job fails to execute on some ESP servers, SAS Event Stream Manager enables you to save a filter that references those failed servers. For more information, see “Rerun a Job” on page 134.

---

**Working with Filters for a Specific Deployment**

When you open a deployment, you can work with filters that are available for that specific deployment.

To work with filters that are available for use with all deployments, use the Filters page instead. For more information, see “Managing Filters” on page 44.
Show the Result of Applying a Filter

Showing the result of applying a filter means showing only those ESP servers that match a selected filter. This functionality is different from applying a filter. A filter is applied when a job template that references the filter is deployed.

1. Open the deployment if it is not already open: on the Deployments page, select the deployment that you want to open and click .

   A separate page that displays the deployment opens.

2. In the drop-down list that displays the text Enter a filter expression, either select an existing filter or enter a filter expression. For more information, see “Filter Expression Syntax” on page 46.

3. Click to see what the result of applying this filter would be.

   The table on the page changes to display only those ESP servers that match the filter expression.

You can save your filter for further use. For more information, see “Save a Filter” on page 43.

To display all ESP servers in the table again, click Clear.

Save a Filter

1. Complete the steps in “Show the Result of Applying a Filter” on page 43.

2. Click .

   The Filter Properties window appears.

3. Edit the fields as required:
   - **Name**: The name of the filter. To rename an existing filter, enter a new name.
To overwrite an existing filter, select the filter name from the drop-down list. The contents of all fields, except for **Filter expression**, are replaced with the values from the selected existing filter.

- **Description**: A description of the purpose of the filter.
- **Tags**: One or more tags attributed to the filter.
  Tags can be used to group and sort filters. **Tags** are single-term descriptors for the filter. Tags cannot contain spaces. Duplicate tags are not permitted on a single filter.
- **Filter expression**: A valid expression that defines the filter criteria. For more information, see “Filter Expression Syntax” on page 46.
- **Scope**: Select **One deployment** or **Any deployment**. If you select **One deployment**, the filter is assigned to the current deployment only. However, you cannot assign the filter to a different deployment on this page. This functionality is available only when you edit the filter in the **Filters** page. For more information, see “Edit a Filter” on page 46.

4 Click **OK**.

---

**Managing Filters**

The **Filters** page enables you to manage filters that are available in SAS Event Stream Manager. This includes filters that are available for use only with specific deployments as well as filters that are available for use with all deployments.

**Sort Filters**

You can arrange the information in the table on the **Filters** page in several ways. However, you might find it particularly helpful to sort the table using the Usage column, so that filters available to different deployments or to all deployments are listed together. For more information about sorting tables, see “Arrange Information in Tables” on page 12.
Create a Filter

The Filters page enables you to create a filter by entering a filter expression. However, you cannot check the result of applying your filter on this page. This functionality is available only when you open the page for a specific deployment and enter a filter expression there. For more information, see “Working with Filters for a Specific Deployment” on page 42.

1. Click [New].

The Filter Properties window appears.

2. Edit the fields as required:
   - **Name**: The name of the filter.
   - **Description**: A description of the purpose of the filter.
   - **Tags**: One or more tags attributed to the filter.
     Tags can be used to group and sort filters. Tags are single-term descriptors for the filter. Tags cannot contain spaces. Duplicate tags are not permitted on a single filter.
   - **Filter expression**: A valid expression that defines the filter criteria. For more information, see “Filter Expression Syntax” on page 46.
Scope: Select One deployment and then select the desired deployment, or select Any deployment.

3 Click OK.

Edit a Filter

To edit a filter, click . The Filter Properties window appears. For more information about how to use the fields in this window, see “Create a Filter” on page 45.

Delete a Filter

1 Click .

   The Delete Filter window appears.

2 Click Delete.

Filter Expression Syntax

Here are examples of supported filter expressions:

- `hostname='server1234.example.com'`
  Finds ESP servers that are running on the host server1234.example.com.

- `label~'Test*'`
  Finds ESP servers whose names begin with the characters Test and are followed by any other characters. For example, this filter finds an ESP server with the name Test_1 but not ESP_Test_1 or test_1.

  Note: If the name of the ESP server contains an apostrophe, enclose the filter term in double quotation marks, and escape the apostrophe with a backslash or an
apostrophe. For example, the filter `label~"SiteA\'s*"` finds an ESP server with the name `SiteA's_ESP_server`.

- `((tags='tag1')|(tags='tag2'))`
  Finds ESP servers that have the tag `tag1` or `tag2` or both.

- `((tags='tag1')&(tags='tag2'))`
  Finds ESP servers that have the tags `tag1` and `tag2`.

- `version='5.2'`
  Finds ESP servers running on SAS Event Stream Processing 5.2.

- `ssl=true`
  Finds ESP servers with SSL authentication.

- `analyticsLicensed=true`
  Finds ESP servers whose hosts have a SAS Event Stream Processing Analytics license installed.
Working with Job Templates

Job Template .......................................................... 50
Create a Job Template ............................................... 51
Upload a Job Template ................................................. 52
Edit a Job Template .................................................. 53
Change the Production Status of a Job Template ............. 55
View a Job Template .................................................. 56
  The Details Tab ...................................................... 56
  The XML Tab ......................................................... 57
  The Versions Tab .................................................... 57
Deploy a Job Template ............................................... 58
Download a Job Template ............................................ 59
  Download the Latest Version of a Job Template ............. 59
  Download an Earlier Version of a Job Template ............. 60
Delete a Job Template ............................................... 60
  Delete the Latest Version of a Job Template ................. 60
  Delete an Earlier Version of a Job Template ................. 61
  Delete All Versions of a Job Template ......................... 61
Job Template

A job template is an XML document that contains a set of instructions used to create a job. That is, a job template outlines the steps required to execute a project on an ESP server. When you deploy a job template, a job is created.

The job template also specifies what input you must supply when you deploy the job template. For example, when you deploy a job template that contains an instruction to load a project, SAS Event Stream Manager prompts you to select the project that you want to load.

A job template contains the following high-level elements:

- **localization-strings** – a collection of strings that display localized output
- **parameters** – a collection of user parameters that allow a user to enter data into the deployment and customize a job template when it is deployed
- **enumerations** – a collection of definitions that restrict user input when used with user parameters
- **initialization** – a collection of definitions to replace placeholders to alter a project when it is deployed
- **server-filters** – a collection of definitions that specify filters for ESP servers
- **instructions** – a collection of instructions that describe operations that must be performed to create or modify a deployment
- **failure-instructions** – a collection of instructions that attempt to return the system to normal operation after an instruction has failed to execute

If the job template contains a **parameters** element, you are prompted to enter each of the user parameters when you deploy the job template. SAS Event Stream Manager displays a list of projects and ESP servers that are available.

The string values from a job template’s **localization-strings** element are used to construct labels used when specifying user parameters. After you have selected your
user parameters, SAS Event Stream Manager produces and executes a job derived from the job template and the user parameters that you entered. As the job executes, you receive progress updates. The string values from the localization-strings element are used to display labels for each instruction.

For example job templates, see “Overview of Example Job Templates” on page 63.
For more information about job template contents, see “Overview of Job Template Content” on page 78.

Create a Job Template

1. On the Job Templates page, click

   The Create New Job Template window appears.

2. In the Name field, enter a unique ID for the job template.

3. In the Description field, you can enter the purpose of the job template.

4. In the Tags field, you can attribute one or more tags to the job template.

   Tags can be used to group and filter job templates. Tags are single-term descriptors for the job template. Tags cannot contain spaces. Duplicate tags are not permitted on a single job template.

5. If you want to create a production job template, select the Production template check box. For more information, see “Production Assets” on page 14.

6. In the Version notes field, you can add notes to further describe the job template.

   Adding notes is especially useful if you want to track changes between versions of the same job template within SAS Event Stream Manager.

7. Click OK.

   Your new job template appears on the Job Templates page. The following figure shows an example:
You must now add content to your job template to make it functional. For more information, see “Edit a Job Template” on page 53.

Upload a Job Template

To upload a job template:

1  On the Job Templates page, click and select Upload job template.
   The Upload Job Template window appears.

2  In the File field, click Browse and navigate to the location of your job template.
   Job templates must be in XML format.

3  In the Tags field, you can attribute one or more tags to the job template.
   Tags can be used to group and filter job templates. Tags are single-term descriptors for the job template. Tags cannot contain spaces. Duplicate tags are not permitted on a single job template.

4  If you want this job template to be a production job template, select the Production template check box. For more information, see “Production Assets” on page 14.

5  In the Version notes field, you can add notes to further describe the job template.
   Adding notes is especially useful if you want to track changes between versions of the same job template within SAS Event Stream Manager.
6 Click OK.

When successfully uploaded, the job template appears on the Job Templates page. The following figure shows an example:

![Job Template Table]

Note: If the job template fails validation, an error message appears. You can use the information in the message to correct the issue and then upload a revised job template.

---

Edit a Job Template

1 On the Job Templates page, select the job template that you want to edit and click .

A detailed view of the job template appears.

2 Click the XML tab.

The XML that defines the job template appears. The following figure shows an example:
3 Edit the content of the job template as required. For more information, see “Overview of Job Template Content” on page 78.

The following table describes the buttons on the XML editor toolbar:

<table>
<thead>
<tr>
<th>Icon</th>
<th>Action</th>
<th>Keyboard Shortcut</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Revert]</td>
<td>Reverts your previous change.</td>
<td>Ctrl + Z</td>
</tr>
<tr>
<td>![Revert]</td>
<td>Reverts the effects of the undo action.</td>
<td>Ctrl + Y</td>
</tr>
<tr>
<td>![Remove]</td>
<td>Removes the selected code from its original position.</td>
<td>Ctrl + X</td>
</tr>
<tr>
<td>![Copy]</td>
<td>Copies the selected code to the clipboard.</td>
<td>Ctrl + C</td>
</tr>
<tr>
<td>![Paste]</td>
<td>Pastes the code on the clipboard at the cursor’s position.</td>
<td>Ctrl + V</td>
</tr>
<tr>
<td>![Find]</td>
<td>Searches for specific text.</td>
<td>Ctrl + F</td>
</tr>
<tr>
<td>![Format]</td>
<td>Formats manually entered XML code.</td>
<td>not available</td>
</tr>
</tbody>
</table>

4 Click ![Apply].
The New Version window appears.

5 In the Tags field, you can attribute one or more tags to the job template.

Tags can be used to group and filter job templates. Tags are single-term descriptors for the job template. Tags cannot contain spaces. Duplicate tags are not permitted on a single job template.

6 In the Version notes field, enter any additional information to describe the changes that you have made to this version of the job template.

7 Click OK.

Only valid job templates can be saved. If the changes that you made to the job template are valid, the Job Template Saved window appears.

8 Click OK.

The instructions here explain how to edit the latest version of a job template. It is also possible to edit an earlier version of a job template. When you save changes to an earlier version of a job template, that version becomes the latest version of the job template. For more information about accessing earlier versions, see “The Versions Tab” on page 57.

---

**Change the Production Status of a Job Template**

Certain assets can be marked as production assets. For more information, see “Production Assets” on page 14.

To change the production status of a job template, on the Job Templates page, right-click the job template and select Toggle production.

The Production column on the Job Templates page changes to reflect your choice.
View a Job Template

To view detailed information about a job template, right-click the job template on the Job Templates page and select Open. A page with three tabs appears.

The Details Tab

The Details tab displays general information relating to the job template entered when the job template was created, uploaded, or edited. The following figure shows an example:

Figure 6.1 The Details Tab
The XML Tab

The XML tab displays the XML for the job template. You can collapse XML elements to restrict your view to the elements that you are interested in. The following figure shows an example of the XML tab:

Figure 6.2   The XML Tab

The Versions Tab

The Versions tab displays the current version and any previous versions of the job template. The following figure shows an example:
To open a previous version, right-click it and select **Open job template version**. A separate page appears where you can review the metadata and XML content relating to that version.

### Deploy a Job Template

1. On the **Job Templates** page, select a job template and click [Deploy](#).

   The Job Template window appears.

2. In the **Deployment** field, select the deployment to which you want to deploy the job template.
The Job Template window is refreshed to display additional fields, which reflect the parameters element of the job template that you are deploying. Complete these additional fields.

3 Click OK.

SAS Event Stream Manager creates a job from the job template that you deployed. Your new job appears on the Log page. The following figure shows an example:

You can now monitor the running job. For more information, see “Monitor Jobs” on page 131.

You can also monitor the deployment. For more information, see “Monitor Deployments” on page 118.

For information about stopping a running job, see “Stop a Running Job” on page 136.

You can deploy only the latest version of a job template. To deploy an earlier version, edit and save that earlier version so that it becomes the latest version. For more information about accessing earlier versions, see “The Versions Tab” on page 57.

---

Download a Job Template

Download the Latest Version of a Job Template

1 On the Job Templates page, select the job template that you want to download.

2 Click and select Download job template.
The job template is downloaded to your computer. The location of the downloaded job template might vary depending on your browser’s configuration.

**Download an Earlier Version of a Job Template**

1. On the **Job Templates** page, right-click the job template and select **Open**.
   
   A separate page with detailed information about the job template appears.

2. Click the **Versions** tab.

3. Right-click the version that you want to download and select **Download job template**.
   
   The job template is downloaded to your computer. The location of the downloaded job template might vary depending on your browser’s configuration.

---

**Delete a Job Template**

If you have edited a job template several times and saved multiple versions of it, you can delete the entire job template or a specific version of the job template. If there is only one version of the job template, you can use any of the deletion methods described here.

**Delete the Latest Version of a Job Template**

1. On the **Job Templates** page, select the job template whose latest version you want to delete.

2. Click .

   The Delete Latest Version window appears.

3. Click **OK** to confirm the deletion.
The latest version of the job template is deleted. If there are any remaining earlier versions, the job template is still displayed on the **Job Templates** page.

**Delete an Earlier Version of a Job Template**

1. On the **Job Templates** page, right-click the job template and select **Open**.

2. Click the **Versions** tab.

3. Right-click the version that you want to delete and select **Delete job template version**.
   
   The Delete Job Template window appears.

4. Click **Delete** to confirm the deletion.

**Delete All Versions of a Job Template**

1. On the **Job Templates** page, right-click the job template and select **Open**.

2. Click the **Versions** tab.

3. Click [trash can](#).

   The Delete Job Template window appears.

4. Click **Delete** to confirm the deletion.
Example Job Templates

Overview of Example Job Templates

Example Job Template: Stock Trade

Overview

Job Template Outline

Adding an Enumeration

Adding the load-project Instruction

Adding Parameters for the load-project Instruction

Adding String References

Adding the project-transformation Instruction

Adding a Filter

The Complete File

Example Job Template: Stop a Project

Overview of Example Job Templates

The topics in this chapter describe example job templates:

- For a job template that loads and starts a project, see “Example Job Template: Stock Trade” on page 64.

- For a job template that stops and unloads a project, see “Example Job Template: Stop a Project” on page 74.
These job templates are also used as part of an end-to-end example of how SAS Event Stream Manager works. For more information, see “Overview of the Stock Trade Example” on page 143.

---

Example Job Template: Stock Trade

**Overview**

This example describes a job template that loads and starts a SAS Event Stream Manager project, trades_connector_stocksymbol_job_template.xml.

An outline of the job template is created first, and then detail is added to each of the high-level elements in the job template. The complete job template XML is displayed at the end. For more information, see “The Complete File” on page 72.

The complete job template file is included in the SAS Event Stream Manager examples package, which you can download from SAS Event Stream Manager examples.

This job template is also used as part of another example, where it is used together with other resources to create a deployment and deploy a job. For more information, see “Overview of the Stock Trade Example” on page 143.

**Job Template Outline**

Here is the outline for the trades_connector_stocksymbol_job_template.xml file:

```xml
<?xml version="1.0"?>
<job-template id="FilteredTrades"
    localization-id="jobName"
    description-localization-id="jobDescription">

    <localization-strings default-language="en-gb">
        <language id="en-gb">
            <string id="jobName">Filter Trades by Stock Code</string>
            <string id="jobDescription">Deploy filtered trades demo with data connectors</string>
        </language>
    </localization-strings>
</job-template>
```
The top-level element, job-template, has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>A unique identifier given to the job template.</td>
</tr>
<tr>
<td>localization-id</td>
<td>A reference to a string value for the job template name used in SAS Event Stream Manager.</td>
</tr>
<tr>
<td>description-localization-id</td>
<td>A reference to a string value for the job template description used in SAS Event Stream Manager.</td>
</tr>
</tbody>
</table>

The localization-strings element in the job template outline stores the string identifier and values for each supported language. Within the localization-strings element, there is a language element whose id attribute is set to en-gb (United Kingdom English). This language element contains string elements. The jobName and jobDescription strings are referenced by the localization-id and description-localization-id attributes in the job-template element at the beginning of the file. When the job template is uploaded and then displayed in SAS Event Stream Manager, these values are displayed as metadata about the job template.

The enumerations element in the job template outline is populated with elements that specify user input options.
The `parameters` element in the job template outline is populated with elements to gather the user input required for executing instructions.

The `initialization` element in the job template outline is populated with elements to alter SAS Event Stream Processing projects as the projects are deployed.

The `server-filters` element in the job template defines a filter for ESP servers. When the job template is deployed, the filter resolves to a specific ESP server.

The `instructions` element in the job template outline guides SAS Event Stream Manager through the process of creating or modifying a deployment. It is populated with an instruction to start a SAS Event Stream Processing model.

**Adding an Enumeration**

An `enumeration` is a finite list of options that is presented to the user to restrict input. The enumeration is referenced from the `parameters` element of the job template. In this example, the user is limited to selecting from three stocks. The enumeration uses string identifiers for the localized string values that are displayed in SAS Event Stream Manager during execution.

The XML `enumerations` element looks like this:

```xml
<enumerations>
  <enumeration id="symcode">
    <enumeration-value id="MCRO" localization-id="mcro" />
    <enumeration-value id="SPY" localization-id="spy" />
    <enumeration-value id="LQD" localization-id="lqd" />
  </enumeration>
</enumerations>
```

The `enumeration` element has an attribute called `id`, which is a unique identifier given to this enumeration.

Each `enumeration-value` child element has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>A unique identifier given to the enumeration value.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Purpose</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>localization-id</td>
<td>A reference to a string value for the enumeration value when it is displayed in SAS Event Stream Manager.</td>
</tr>
</tbody>
</table>

**Adding the load-project Instruction**

Instructions tell SAS Event Stream Manager what to do with assets in a deployment. In this example, a `load-project` instruction is added to load and then start a SAS Event Stream Processing project.

Like other SAS Event Stream Manager instructions, the `load-project` instruction uses string identifiers for the localized string values that are displayed in SAS Event Stream Manager during execution. It also has XML attributes for parameters to identify required input for the project to load and identify the ESP server, as well as other additional parameters. These parameters can be specified as literal values in the XML or as references to user-defined input parameters. In this example, user-defined input parameters are specified.

The XML `instructions` element looks like this:

```xml
<instructions>
  <load-project
    id="load-project"
    localization-id="load-project"
    server-filter="filter1"
    project-id="{transformed-project.id}"
    project-version="{project.version}"
    running-project-name="{symcode}_{project.version}"
    start="true" />
</instructions>
```

The XML element name, `load-project`, is the instruction name. In this example, the XML line has been split to make it more readable. In a functional job template, it is more likely to be on a single line.
The load-project instruction has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>A unique identifier given to the instruction. All instructions require this attribute, and its value must be unique in the job template.</td>
</tr>
<tr>
<td>localization-id</td>
<td>A reference to a string value for the instruction when it is displayed in SAS Event Stream Manager.</td>
</tr>
<tr>
<td>server-filter</td>
<td>A reference to a filter for ESP servers. A filter resolves to a set of ESP servers to which the project will be deployed. The filter is defined in the server-filters element within the job template.</td>
</tr>
<tr>
<td>project-id</td>
<td>A unique identifier is created internally for each SAS Event Stream Processing project that has been uploaded to SAS Event Stream Manager. This value is specified either as a literal string or with a variable placeholder indicated by the use of the braces, { and }. The most reliable and easiest way to obtain this identifier is through the variable resolution used in this example. The variable in this case is {transformed-project.id}, which is explained in the initialization section.</td>
</tr>
<tr>
<td>project-version</td>
<td>Each project that has been uploaded to SAS Event Stream Manager is versioned. This value is specified either as a literal string or with a variable placeholder indicated by the use of the braces, { and }. The variable in this case is {project.version}, which is referenced and explained in the parameters section.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Purpose</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>running-project-name</td>
<td>A name for the running project that will be created in SAS Event Stream Processing. This value must be unique to the running ESP server that will execute the project. This value is specified either as a literal string or with variable placeholders indicated by the use of the braces, { and }. In this case, two variables are specified with a literal underscore ( _ ) to create a project name, {symcode}_{project.version}. These variables are explained in the parameters section.</td>
</tr>
<tr>
<td>start</td>
<td>The value of this attribute tells SAS Event Stream Manager to start running the SAS Event Stream Processing project after it has been loaded. In this case, the value is true. If it was false, an additional instruction, start-project, would be required to start the project.</td>
</tr>
</tbody>
</table>

### Adding Parameters for the load-project Instruction

When SAS Event Stream Manager reads a job template in preparation for executing instructions, it first looks to resolve the parameters defined in the parameters element.

Given the content of the load-project instruction, parameters that identify the following items must be resolved:

- the project itself
- the ESP server to deploy the project onto
- a value for the name of the running project
Three selectors are added to the parameters element:

```xml
<parameters>
  <project-selector id="project"
                   localization-id="project" />
  <server-selector id="server"
                   localization-id="server" />
  <enum-selector id="symcode"
                 localization-id="symcode"
                 required="true"
                 source="symcode" />
</parameters>
```

In this example, each selector element is split onto several lines to make it more readable. In a functional job template, it is more likely to be on a single line.

The localization-id attributes are discussed later (for more information, see “Adding String References” on page 71).

The attributes for the selectors are similar. Each of the attributes is described as follows:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>A unique identifier given to the user parameter.</td>
</tr>
<tr>
<td>localization-id</td>
<td>A reference to a string value for this parameter when it is displayed in SAS Event Stream Manager.</td>
</tr>
<tr>
<td>required</td>
<td>If set to true, the user must provide an input value.</td>
</tr>
<tr>
<td>source</td>
<td>This attribute is used only for enum-selector. It provides information about the source of the data. The value must be the name of a defined enumeration.</td>
</tr>
</tbody>
</table>

The project-selector element provides the user with a drop-down list that contains the SAS Event Stream Processing projects that have been uploaded to SAS Event Stream Manager. When a value is selected, the id and version properties of the selected project can be referenced throughout the job template. You can reference these properties by using the id attribute, with either a .id or .version suffix.
The server-selector element provides the user with a drop-down list that contains the ESP servers that have been added to SAS Event Stream Manager. When a value is selected, the id and version properties of the selected project can be referenced throughout the job template. You can reference these properties by using the id attribute, with either a .id or .version suffix.

The enum-selector element provides the user with a drop-down list that contains values from the referenced enumeration. That is, this selector provides a list of stock codes.

Adding String References

The load-project instruction and the three selectors within the parameters element all specify localization-id attribute values that need to be added to the localization-strings element of the job template. Each id attribute must be associated with a string value. These string values are used by SAS Event Stream Manager when prompting for input or when showing progress or error messages when executing a job created from the job template.

The string references are added under the language id="en-gb" element from the outline XML defined earlier. The string references are as follows:

```xml
<string id="project">Project</string>
<string id="server">ESP server</string>
<string id="load-project">Load Project</string>
<string id="symcode">Stock symbol code</string>
<string id="mcro">Micro Focus</string>
<string id="spy">SPDR</string>
<string id="lqd">LQD</string>
```

Adding the project-transformation Instruction

An instruction to replace placeholders when a project is deployed needs to be added to the initialization element. Replacing the placeholders is achieved by using the project-transformation instruction.

The XML initialization element looks like this:

```xml
<initialization>
  <project-transformations>
```

Variables used for \{project\} are resolved at execution when the user selects a project. The placeholder \{symcode\} is replaced with the stock code selected by the user.

**Adding a Filter**

A server-filters element must be added to define a filter for ESP servers. In this example, the filter resolves to the ESP server selected by the user when the job template is deployed. However, in a more complicated example, the filter might resolve to a larger set of ESP servers, and the project might be deployed to more than one ESP server.

The XML server-filters element looks like this:

```xml
<server-filters>
  <server-filter id="filter1"
      filter-expression="objectId='{server.id}'" />
</server-filters>
```

The filter expression contains a \{server.id\} variable. This variable references the server-selector element within the parameters element.

In this example, the server-filters element is added between the initialization and instructions elements.

**The Complete File**

Here is the complete trades_connection_stocksymbol_job_template.xml file:

```xml
<?xml version="1.0"?>
<job-template id="FilteredTrades"
    localization-id="jobName"
    description-localization-id="jobDescription">
  <localization-strings default-language="en-gb">
    <language id="en-gb">
```
<string id="jobName">Filter Trades by Stock Code</string>
<string id="jobDescription">Deploy filtered trades demo with data connectors</string>
<string id="project">Project</string>
<string id="server">ESP server</string>
<string id="load-project">Load Project</string>
<string id="symcode">Stock symbol code</string>
<string id="mcr">Micro Focus</string>
<string id="spy">SPDR</string>
<string id="lqd">iBoxx</string>

</localization-strings>
<enumerations>
  <enumeration id="symcode">
    <enumeration-value id="MCRO" localization-id="mcr" />
    <enumeration-value id="SPY" localization-id="spy" />
    <enumeration-value id="LQD" localization-id="lqd" />
  </enumeration>
</enumerations>
<parameters>
  <project-selector id="project" localization-id="project" />
  <server-selector id="server" localization-id="server" />
  <enum-selector id="symcode" localization-id="symcode"
    required="true"
    source="symcode" />
</parameters>
<initialization>
  <project-transformations>
    <project-transformation id="transformed-project"
      project-id="{project.id}"
      project-version="{project.version}">
      <placeholder id="symcode">{symcode}</placeholder>
    </project-transformation>
  </project-transformations>
</initialization>
<server-filters>
  <server-filter id="filter1" filter-expression="objectId='{server.id}'" />
</server-filters>
<instructions>
  <load-project id="load-project"
    localization-id="load-project"
    server-filter="filter1"
    project-id="{transformed-project.id}"
Example Job Template: Stop a Project

This example describes a simple job template, stop_project_job_template.xml, that stops a running project and unloads it from the ESP server.

The complete job template file is included in the SAS Event Stream Manager examples package, which you can download from SAS Event Stream Manager examples.

This job template is also used as part of another example. For more information, see “Overview of the Stock Trade Example” on page 143.

Here is the complete stop_project_job_template.xml file:

```xml
<?xml version="1.0"?>
<job-template id="clean-server" localization-id="jobName"

description-localization-id="jobDescription">

<localization-strings>
  <language id="en-gb">
    <string id="jobName">Remove a Running Project</string>
    <string id="jobDescription">Unload a project from a
      specified ESP server</string>
    <string id="target-server">ESP server</string>
    <string id="project-to-stop">Project to unload</string>
    <string id="stop-project">Stop Project</string>
    <string id="unload-project">Unload Project</string>
  </language>
</localization-strings>

<parameters>
  <server-selector id="server"
    localization-id="target-server"
    required="true"/>
  <running-project-selector id="project-to-stop"
    localization-id="project-to-stop"
    required="true"/>
</parameters>

<server-filters>
</server-filters>
</job-template>
```
The localization-strings element stores the string identifier and values for each supported language. Within the localization-strings element, there is a language element whose id attribute is set to en-gb (United Kingdom English). This language element contains string elements. The jobName and jobDescription strings are referenced by the localization-id and description-localization-id attributes in the job-template element at the beginning of the file. When the job template is uploaded and then displayed in SAS Event Stream Manager, these values are displayed as metadata about the job template.

The parameters element contains selectors that gather the user input required to execute instructions.

The server-filters element defines a filter for ESP servers. The {server.id} variable references the server-selector element within the parameters element. In this example, the filter resolves to the ESP server selected by the user when the job template is deployed. However, in a more complicated example the filter might resolve to a larger set of ESP servers.

The instructions element contains instructions for stopping the running project specified by the user and unloading it from the ESP server. For more information about the stop-project instruction, see “stop-project” on page 108. For more information about the unload-project instruction, see “unload-project” on page 109.
## Job Template Content

**Overview of Job Template Content** ........................................ 78

*job-template* ................................................................. 78

*localization-strings* .......................................................... 81

*parameters* .......................................................................... 83
- Overview of the parameters Element ........................................ 83
- Common Attributes of Selectors ............................................. 85
- project-selector .................................................................... 86
- server-selector .................................................................... 88
- running-project-selector ...................................................... 89
- text-input-selector ............................................................... 90
- query-selector ..................................................................... 91
- window-selector ................................................................... 91
- enum-selector ...................................................................... 92
- server-filter-selector ........................................................... 92

*enumerations* ........................................................................ 93

*server-filters* ...................................................................... 95

*instructions* ......................................................................... 97
- instructions ........................................................................ 97
- Common Attributes of Instructions and the group Element ........ 101
- load-project ....................................................................... 103
- start-project ....................................................................... 106
- start-connectors ................................................................. 106
Overview of Job Template Content

The topics in this chapter contain information about the content that you can add to job templates.

For a high-level introduction to job templates, see “Job Template” on page 50.

For an overview of example job templates, see “Overview of Example Job Templates” on page 63.

job-template

The job-template element is the top-level element in a job template.

Here is an example:

```xml
<job-template id="test1"
    localization-id="template-name"
    description-localization-id="template-description"/>

<!-- The rest of the job template contents are added here. -->

</job-template>
```

Table 8.1 Properties of the job-template XML Element

<table>
<thead>
<tr>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>modify-project</td>
</tr>
<tr>
<td>stop-project</td>
</tr>
<tr>
<td>unload-project</td>
</tr>
<tr>
<td>group</td>
</tr>
<tr>
<td>failure-instructions</td>
</tr>
<tr>
<td>initialization</td>
</tr>
<tr>
<td>Attribute</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>id</td>
</tr>
<tr>
<td>localization-id</td>
</tr>
<tr>
<td>description-localization-id</td>
</tr>
</tbody>
</table>

**Child Elements**

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
<th>Required</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>localization-strings</td>
<td>This element contains strings that display localized output.</td>
<td>Yes</td>
<td>The localization-strings element must appear as the first child element of the job-template element.</td>
</tr>
<tr>
<td></td>
<td>For more information, see “localization-strings” on page 81.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
<td>Required</td>
<td>Additional Information</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>parameters</td>
<td>This element contains user parameters that allow a user to enter data into the deployment and customize a job template when it is deployed.</td>
<td>No</td>
<td>The <code>parameters</code> element must appear before the initialization, instructions, server-filters, and failure-instructions elements.</td>
</tr>
<tr>
<td></td>
<td>For more information, see “parameters” on page 83.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>enumerations</td>
<td>This element contains definitions that restrict user input when used with user parameters.</td>
<td>No</td>
<td>For more information, see “enumerations” on page 93.</td>
</tr>
<tr>
<td>initialization</td>
<td>This element contains definitions to replace placeholders to alter a project when it is deployed.</td>
<td>No</td>
<td>For more information, see “initialization” on page 114.</td>
</tr>
<tr>
<td>server-filters</td>
<td>This element specifies filters for ESP servers.</td>
<td>No</td>
<td>For more information, see “server-filters” on page 95.</td>
</tr>
<tr>
<td>instructions</td>
<td>This element contains instructions that describe operations that must be performed to create or modify a deployment.</td>
<td>No</td>
<td>The <code>instructions</code> element must appear before the failure-instructions element.</td>
</tr>
<tr>
<td></td>
<td>For more information, see “instructions” on page 97.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This element contains instructions that attempt to return the system to normal operation after an instruction has failed to execute.

- Required: No.
- For more information, see “failure-instructions” on page 111.

### localization-strings

The `localization-strings` element of the job template contains the string values for the labels displayed by SAS Event Stream Manager when executing that job template. Each string value is fully localizable within one or more language groups. The `localization-strings` element requires a default language.

Here is an example:

```xml
<job-template id="test1"
  localization-id="template-name"
  description-localization-id="template-description">
  <localization-strings default-language="en-us">
    <language id="en-us">
      <string id="template-name">test1</string>
      <string id="template-description">Loads a project</string>
    </language>
  </localization-strings>
</job-template>
```

Each string element has an `id` attribute that is referred to by other elements of the job template when a string value is required. In the example here, `template-name` and `template-description` are referred to in the top-level `job-template` element.

In the SAS Event Stream Manager user interface, the string values are displayed in the following locations:

- on the **Job Templates** page
- in the window that appears when you deploy a job template from the **Job Templates** page
on the **Log** page when the job created from the job template is executing

**Table 8.2  Properties of the localization-strings XML Element**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>default-language</code></td>
<td>This attribute specifies the language to be used if the user’s current locale is not supported.</td>
</tr>
<tr>
<td></td>
<td>▪ Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>▪ Valid values: This value must correspond to a defined language identifier. The value must be a valid ISO locale code. The locale code must be lowercase and in the format <code>aa-bb</code> (for example: <code>en-us</code>).</td>
</tr>
</tbody>
</table>

**Child Element**

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>language</code></td>
<td>There must be one <code>language</code> element for each supported language and at least one that corresponds to the <code>default-language</code> attribute of the parent localiztion-strings element.</td>
</tr>
</tbody>
</table>

**Table 8.3  Properties of the language XML Element**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>id</code></td>
<td>This attribute specifies the locale code for this language.</td>
</tr>
<tr>
<td></td>
<td>▪ Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>▪ Valid values: A valid ISO locale code. The locale code must be lowercase and in the format <code>aa-bb</code> (for example: <code>en-us</code>).</td>
</tr>
</tbody>
</table>

**Child Element**

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>string</code></td>
<td>There must be one <code>string</code> element for each localization string.</td>
</tr>
</tbody>
</table>
Overview of the parameters Element

The parameters element of the job template tells SAS Event Stream Manager what user input to capture and assign to internally stored variables that can be referenced throughout the job template. The parameters element includes child elements whose purpose is to guide user input. These selectors restrict the user's text input or enable the user to select an option from a list. In the following example, the project-selector element enables the user to select from a list of projects stored in SAS Event Stream Manager’s repository:

```xml
<parameters>
  <project-selector id="project"
      localization-id="selectProject"
      required="true"/>
</parameters>
```

You can nest some selectors inside other selectors. For example, information might be needed to populate available values for the selector in the user interface. In addition, you might need to show some parameters only upon selection of a valid value for another parameter. In such cases, the selectors should be nested inside each other, as shown in the following example:

```xml
<parameters>
  <project-selector id="project"
      localization-id="selectProject"
      required="true">
    <query-selector id="query"
        localization-id="checkQuery"
        required="true">
      <window-selector id="window"
          localization-id="checkWindow"
          required="true"/>
    </query-selector>
  </project-selector>
</parameters>
```
### Table 8.4  Properties of the parameters XML Element

<table>
<thead>
<tr>
<th>Child Elements</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>project-selector</td>
<td>This element enables the user to select from a list of projects stored in SAS Event Stream Manager’s repository. For more information, see “project-selector” on page 86.</td>
</tr>
<tr>
<td>server-selector</td>
<td>This element enables the user to select from a list of ESP servers that are managed by SAS Event Stream Manager. For more information, see “server-selector” on page 88.</td>
</tr>
<tr>
<td>running-project-selector</td>
<td>This element enables the user to select a project that is either running on a specific SAS Event Stream Processing server or contained in an uploaded project. For more information, see “running-project-selector” on page 89.</td>
</tr>
<tr>
<td>text-input-selector</td>
<td>This element enables the user to enter a text string that is checked against a regular expression. For more information, see “text-input-selector” on page 90.</td>
</tr>
<tr>
<td>query-selector</td>
<td>This element enables the user to select a currently running continuous query inside a project on a specific SAS Event Stream Processing server. For more information, see “query-selector” on page 91.</td>
</tr>
<tr>
<td>window-selector</td>
<td>This element enables the user to select a currently running window within a continuous query inside a project on a specific SAS Event Stream Processing server. For more information, see “window-selector” on page 91.</td>
</tr>
</tbody>
</table>
enum-selector

This element enables the user to select from a list of restricted values of an enumeration defined either globally or locally in the deployment. For more information, see “enum-selector” on page 92.

server-filter-selector

This element enables the user to select a previously created filter. For more information, see “server-filter-selector” on page 92.

Common Attributes of Selectors

Selectors have some common attributes, as shown in the following table:

Table 8.5  Common Attributes of Selectors

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>This attribute specifies the unique identifier of the selector. This identifier is placed into the execution context.</td>
</tr>
<tr>
<td>localization-id</td>
<td>This attribute specifies the localization string ID.</td>
</tr>
</tbody>
</table>

id

- Required: Yes.
- Valid values: This value must match the regular expression `[0-9A-z] [0-9A-Z_] +.`

localization-id

- Required: Yes.
- Valid values: This value must exist in the default language information within the localization-strings element.
This attribute indicates whether it is mandatory for the user to provide input. If you set this attribute to true, the user must provide input, and an asterisk (*) is displayed next to the user interface control to indicate this to the user.

- Required: No.
- Valid values: true and false.

This attribute specifies the localization string ID that is displayed if validation fails.

- Required: Yes.
- Valid values: This value must exist in the default language information within the localization-strings element.

When a user selects the value of a selector, that value is stored internally by SAS Event Stream Manager. You can reference this value in the job template by the selector’s id value. To do this, enclose the id value in braces: { and }.

An id value can have one or more attributes. To access these attributes, use a dot (.) notation. For example, if the selector’s id value is alpha and this id value has the attribute beta, you can reference the selector by writing {alpha.beta}. Additional examples are provided in the following sections, where each selector is discussed in more detail.

**project-selector**

The project-selector element enables the user to select from a list of projects stored in SAS Event Stream Manager’s repository. This action produces a project object that has the following variables:

- id: the unique identifier of the project that has been selected by the user
- version: the version number of the project that has been selected by the user
- friendlyName: the name of the project that has been selected by the user
Note: Project XML files created in SAS Event Stream Processing Studio 5.2 contain separate variables that correspond to a project’s major version number and a project’s minor version number. This attribute specifies a project’s major version number and minor version number as follows: `project-version="{project.majorVersion}. {project.minorVersion}"`. In addition, previous versions of SAS Event Stream Processing contained a `project.version` variable that corresponded to the project's major version number. In SAS Event Stream Processing 5.2, the `project.version` variable is still valid. However, it is recommended that you specify the project's major version using the `project.majorVersion` variable.

The `friendlyName` project object can be particularly useful if you need to identify the project using its name rather than its identifier. For example, some SAS Event Stream Processing adapters need to reference the project name.

The `project-selector` element is a top-level selector and does not need to be nested inside any other selectors to function properly.

The user does not select the project version. Instead, the latest version of the project is used at all times.

The `project-selector` element is useful for choosing a project that is then published to the ESP server. An example from a simplified job template is shown here:

```xml
<parameters>
  <project-selector id="project"
    localization-id="source"
    required="true" />
  <server-selector id="server"
    localization-id="dstServer"
    required="true" />
</parameters>
<server-filters>
  <server-filter id="filter1"
    filter-expression="(objectId='{server.id}')" />
</server-filters>
<instructions>
  <load-project id="load-project-1"
    localization-id="load-project"
    server-filter="filter1"
    project-id="{project.id}"
    project-version="{project.version}"
    running-project-name="{project.friendlyName}"
    start="true" />
```
server-selector

The server-selector element enables the user to select from a list of ESP servers that are associated with the selected deployment. This action produces an ESP server object that has the following variable:

id: the unique identifier of the ESP server that has been selected by the user

The server-selector element is a top-level selector and does not need to be nested inside any other selectors to function properly.

Alongside the project-selector element, the server-selector element is one of the most commonly used parameter types. The server-selector element is typically used to help define a server-filter element, as shown in the simplified example job template here:

```xml
<parameters>
  <project-selector id="project"
    localization-id="source"
    required="true" />
  <server-selector id="server"
    localization-id="dstServer"
    required="true" />
</parameters>
<server-filters>
  <server-filter id="filter1"
    filter-expression="(objectId='{server.id}')" />
</server-filters>
<instructions>
  <load-project id="load-project-1"
    localization-id="load-project"
    server-filter="filter1"
    project-id="{project.id}"
    project-version="{project.version}"
    running-project-name="{project}"
    start="true" />
</instructions>
```
**running-project-selector**

The `running-project-selector` element enables the user to select a project that is either running on a specific SAS Event Stream Processing server or contained in an uploaded project. This action produces a project object that has the following variable:

- **id**: the unique identifier of the project that has been selected by the user

The `running-project-selector` element must be nested inside a `server-selector` element to function properly.

Not all job templates relate to new installations. In some cases, you might need to make changes to existing projects or unload existing projects to complete the deployment successfully. The example here demonstrates how the following tasks are performed:

1. prompts the user to select an ESP server
2. prompts the user to select a project
3. stops the selected project running on that server

```xml
<parameters>
<server-selector id="server"
localization-id="dstServer"
required="true"
<running-project-selector
  id="project"
  localization-id="srcProject"
  required="true"
</parameters>
<server-filters>
<server-filter id="filter1"
  filter-expression="(objectId='{server.id}')" />
<server-filters/>
<instructions>
<stop-project id="stop-this-project"
  localization-id="stop-project"
  server-filter="filter1"
  running-project-name="{project.id}" />
</instructions>
```
Note: The `stop-project` instruction stops the project. To entirely unload a project, use the `unload-project` instruction. For more information, see “unload-project” on page 109.

**text-input-selector**

The `text-input-selector` element enables the user to enter a text string that is checked against a regular expression. This action produces a string object that can be used throughout the job template.

The `text-input-selector` element is a top-level selector and does not need to be nested inside any other selectors to function properly.

In addition to properties common to all selectors (for more information, see “Common Attributes of Selectors” on page 85), the `text-input-selector` element has the XML attribute shown in the following table:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>validation-expression</td>
<td>This attribute defines the regular expression that is used to perform validation.</td>
</tr>
<tr>
<td></td>
<td>Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>Valid values: The valid regular expression.</td>
</tr>
</tbody>
</table>

In the following example, the user’s choice of a project name is restricted so that the name must begin with a letter. The remainder of the string must contain letters, numbers, or underscores. If validation fails, the localized string defined in `project-not-valid` is displayed to the user.

```xml
<parameters>
  <project-selector id="project"
                      localization-id="source"
                      required="true" />
  <server-selector id="server"
                      localization-id="dstServer"
                      required="true" />
  <text-input-selector id="project"
                      localization-id="source"
                      required="true" />
</parameters>
```
The `query-selector` element enables the user to select a currently running continuous query inside a project on a particular SAS Event Stream Processing server. This action produces a query object with the following variable:

**id**: the unique identifier of the query that has been selected by the user

The `query-selector` element must be nested inside a `running-project-selector` element to function properly.

The `window-selector` element enables the user to select a currently running window within a continuous query inside a project on a particular SAS Event Stream Processing server. This action produces a query object with the following variable:

**id**: the unique identifier of the window that has been selected by the user

The `window-selector` element must be nested inside a `query-selector` element to function properly.
enum-selector

The enum-selector element enables the user to select from a list of restricted values of an enumeration defined either globally or locally in the deployment. This action produces a string that can then be used anywhere in the job template.

The enum-selector element is a top-level selector and does not need to be nested inside any other selectors to function properly.

In addition to properties common to all selectors (for more information, see “Common Attributes of Selectors” on page 85), the enum-selector element has the XML attribute shown in the following table:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>This attribute provides additional information about the source of the data.</td>
</tr>
<tr>
<td></td>
<td>- Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>- Valid values: The name of a defined enumeration.</td>
</tr>
</tbody>
</table>

For more information about enumerations, see “enumerations” on page 93.

server-filter-selector

The server-filter-selector element enables the user to select a filter. A separate window is displayed to the user. The window lists filters available only to the specific deployment as well as filters available for use with all deployments. The window also provides additional details (such as the filter expression) for each filter to help the user select an appropriate filter.

For more information, see “Filters” on page 41.

The server-filter-selector element does not need to be nested inside any other selectors to function properly.
Here is an example:

```xml
<parameters>
    <server-filter-selector id="failedserverfilter1"
        required="true"
        localization-id="failedserverfilter1" />
</parameters>
<server-filters>
    <server-filter id="filter1"
        filter-expression="{failedserverfilter1.filterExpression}">
    </server-filters>
<instructions>
    <group id="group1" localization-id="group1"
        server-filter="filter1">
        <!-- appropriate instructions are added here -->
    </group>
</instructions>
```

This simplified example shows a `server-filter-selector` element called `failedserverfilter1`. This element is then referenced in a `server-filter` element called `filter1`. This means that the set of ESP servers included in `filter1` is the set of ESP servers returned by whichever filter a user selects for the `failedserverfilter1` selector. Within the `instructions` element, the `group` element called `group1` then refers to `filter1`. In this way, you can specify a group of instructions to execute against a set of ESP servers that a user selects. The strings shown in this example must be referenced in the `localization-strings` element, which is not shown in the example code here.

---

**enumerations**

An *enumeration* is a finite list of options that is presented to the user to restrict input.

Each job template can support the declaration of local localized enumerations. Such declarations are optional. The `enumerations` element of the job template specifies a set of enumerations that can be used in the deployment. Each enumeration is fully localized and produces output to the user in the user’s specified locale (subject to the job template supporting the locale). Here is an example `enumerations` element that defines a single enumeration:
The enumeration with the id value of country can be referenced in the job template from the parameters element when you are limiting the input options the user can select from. In this example, the user is limited to the values USA and UK.

Table 8.8  Properties of the enumerations XML Element

<table>
<thead>
<tr>
<th>Child Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enumeration</td>
<td>There must be one enumeration element for each enumeration.</td>
</tr>
</tbody>
</table>

Table 8.9  Properties of the enumeration XML Element

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>This attribute specifies the unique identifier of the enumeration. If a global enumeration with the same identifier is defined, this one takes precedence.</td>
</tr>
<tr>
<td></td>
<td>- Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>- Valid values: This value must match the regular expression <code>[0-9A-z][0-9A-Z_-]+</code>.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Child Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enumeration-value</td>
<td>Each distinct value in the enumeration must be listed as a child element.</td>
</tr>
</tbody>
</table>
The `server-filters` element specifies filters for ESP servers. A filter resolves to a set of ESP servers to which the project will be deployed or with which the project is associated. For more information about how filters work, see “Filters” on page 41.

**Table 8.10 Properties of the server-filters XML Element**

<table>
<thead>
<tr>
<th>Child Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>server-filter</td>
<td>This element specifies a filter for ESP servers.</td>
</tr>
</tbody>
</table>

**Table 8.11 Properties of the server-filter XML Element**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>This attribute specifies the unique identifier of the server-filter element.</td>
</tr>
<tr>
<td></td>
<td>- Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>- Valid values: This value must match the regular expression <code>[0-9A-z][0-9A-Z_\-]+</code>.</td>
</tr>
</tbody>
</table>
localization-id

This attribute specifies the localization string ID. If a job fails, SAS Event Stream Manager enables the user to create a filter that references the failed ESP servers (for more information, see “Rerun a Job” on page 134). The job template that was used to create the failed job can include multiple server-filter elements. In this case, SAS Event Stream Manager creates a separate filter for each ESP server group on which the job failed. (For more information about the group element, see “group” on page 110.) The id attribute of the relevant server-filter element is added to the filter name that is entered by the user. If the server-filter element includes a localization-id attribute, this attribute's value is used instead of the id attribute to provide a friendly, localizable name. If a translated string is available in the user's language, that translation is used. If no translated string is available, the value of the id attribute is used instead.

- Required: No.
- Valid values: This value must exist in the default language information within the localization-strings element.

filter-expression

This attribute contains a filter expression or references a previously saved filter (which contains a filter expression).

- Required: Yes.
- Valid values: A valid filter expression or the name of a previously saved filter. For more information, see “Filter Expression Syntax” on page 46.

Here is an example where the filter-expression attribute references an existing filter called failedserverfilter1:

```xml
<server-filters>
  <server-filter id="filter1"
    filter-expression="\{failedserverfilter1.filterExpression\}" />
```
Here is a second example, where the filter-expression attribute includes a filter expression:

```xml
<server-filters>
    <server-filter id="filter1"
        filter-expression="{objectId='{server.id}'}" />
</server-filters>
```

In this second example, the filter expression contains a `{server.id}` variable. This code extract is from the Stock Trade example, where the `{server.id}` variable references the server-selector element (within the parameters element). The filter resolves to the ESP server selected by the user when the job template is deployed. For more information, see “Example Job Template: Stock Trade” on page 64.

The server-filters element can also be used together with the server-filter-selector element (within the parameters element). In addition, a server-filter element can be referenced from a group element (within the instructions element). For more information, including additional example code, see “server-filter-selector” on page 92.

---

**instructions**

The instructions element is at the core of the job template. It contains the commands that SAS Event Stream Manager executes. The instructions often contain identifiers for selectors used with the parameters element, as well as string identifiers from the localization-strings element.

---

**instructions**

The instructions element is the parent element that contains specific instructions.

Here is an example that shows the outline of the instructions element. The details of the stop-project and unload-project instructions have been removed.

```xml
<instructions>
    <stop-project <!-- The details for the stop-project instruction are added here. --> />
</instructions>
```
<unload-project <!-- The details for the unload-project instruction are added here. --> />
</instructions>

The instructions element can also contain a group element that groups instructions together.

Here is an example. The details of the elements have been removed.

<instructions>
  <group <!-- The attributes of the first group element are added here. --> >
    <load-project <!-- The details for the load-project instruction are added here. --> />
    <start-project <!-- The details for the start-project instruction are added here. --> />
    <start-connectors <!-- The details for the start-connectors instruction are added here. --> />
  </group>
  <group <!-- The attributes of the second group element are added here. --> >
    <load-project <!-- The details for the load-project instruction are added here. --> />
    <start-project <!-- The details for the start-project instruction are added here. --> />
    <start-connectors <!-- The details for the start-connectors instruction are added here. --> />
  </group>
</instructions>

Table 8.12  Properties of the instructions XML Element

<table>
<thead>
<tr>
<th>Attributes</th>
</tr>
</thead>
</table>
server-filter

This attribute specifies a filter for ESP servers. The filter resolves to a set of ESP servers to which the project will be deployed or with which the project is associated.

- Required: Each instruction must have a server-filter attribute. This attribute can be included in the instruction itself or in the enclosing group element or instructions element.

  A server-filter attribute on a child element overrides a server-filter attribute on a parent element. For example, a server-filter attribute on a specific instruction overrides a server-filter attribute on the parent instructions element.

- Valid values: The ID of a server-filter element. For more information, see “server-filters” on page 95.
on-failure

This attribute specifies how SAS Event Stream Manager handles an instruction that fails to execute. By default, if an instruction fails to execute, any instructions that depend on the failed instruction are canceled. You can use the on-failure attribute to specify different behavior.

- Required: No. This attribute can be included in a specific instruction or in the enclosing group element or instructions element.

An on-failure attribute on a child element overrides an on-failure attribute on a parent element. For example, an on-failure attribute on a specific instruction overrides an on-failure attribute on the parent instructions element.

- Valid values: continue. Setting the on-failure attribute to continue means that dependent instructions will be executed even if the original instruction fails. If you do not want this to happen, remove the on-failure attribute.

Child Elements

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>load-project</td>
<td>This instruction takes a specific project that has previously been uploaded to SAS Event Stream Manager and publishes it to an active ESP server.</td>
</tr>
<tr>
<td>start-project</td>
<td>This instruction starts an already loaded SAS Event Stream Processing project.</td>
</tr>
</tbody>
</table>
start-connectors

This instruction starts any connectors associated with a published SAS Event Stream Processing project.

- Required: No.
- For more information, see “start-connectors” on page 106.

modify-project

This instruction updates an existing running project, subject to the rules imposed by SAS Event Stream Processing.

- Required: No.
- For more information, see “modify-project” on page 107.

stop-project

This instruction stops a published SAS Event Stream Processing project. Stopping a project does not remove the project, but it stops any new connections to it.

- Required: No.
- For more information, see “stop-project” on page 108.

unload-project

This instruction unloads a published SAS Event Stream Processing project.

- Required: No.
- For more information, see “unload-project” on page 109.

group

This element groups instructions together.

- Required: No.
- For more information, see “group” on page 110.

**Common Attributes of Instructions and the group Element**

Instructions and the group element have some common attributes, as shown in the following table:
### Table 8.13  Common Attributes of Instructions and the group Element

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>This attribute specifies the unique identifier of the instruction or the group element.</td>
</tr>
<tr>
<td></td>
<td>- Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>- Valid values: This value must match the regular expression ([0-9A-z][0-9A-Z_‑]+).</td>
</tr>
<tr>
<td>localization-id</td>
<td>This attribute specifies the localization string ID.</td>
</tr>
<tr>
<td></td>
<td>- Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>- Valid values: This value must exist in the default language information within the localization-strings element.</td>
</tr>
<tr>
<td>server-filter</td>
<td>This attribute specifies a filter for ESP servers. The filter resolves to a set of ESP servers to which the project will be deployed or with which the project is associated.</td>
</tr>
<tr>
<td></td>
<td>- Required: Each instruction must have a server-filter attribute. This attribute can be included in the instruction itself or in the enclosing group element or instructions element.</td>
</tr>
<tr>
<td></td>
<td>- A server-filter attribute on a child element overrides a server-filter attribute on a parent element. For example, a server-filter attribute on a specific instruction overrides a server-filter attribute on the parent instructions element.</td>
</tr>
<tr>
<td></td>
<td>- Valid values: The ID of a server-filter element. For more information, see “server-filters” on page 95.</td>
</tr>
</tbody>
</table>
on-failure

This attribute specifies how SAS Event Stream Manager handles an instruction that fails to execute. By default, if an instruction fails to execute, any instructions that depend on the failed instruction are canceled. You can use the on-failure attribute to specify different behavior.

- Required: No. This attribute can be included in a specific instruction or in the enclosing group element or instructions element.

An on-failure attribute on a child element overrides an on-failure attribute on a parent element. For example, an on-failure attribute on a specific instruction overrides an on-failure attribute on the parent instructions element.

- Valid values: continue. Setting the on-failure attribute to continue means that dependent instructions will be executed even if the original instruction fails. If you do not want this to happen, remove the on-failure attribute.

depends-on

This attribute lists the instructions that this instruction or group depends on.

- Required: No.

- Valid values: The values of id attributes of other instructions in the same job template.

The common attributes have been omitted from subsequent sections, which describe additional XML properties for each instruction and the group element.

load-project

This instruction takes a specific project that has previously been uploaded to SAS Event Stream Manager and publishes it to an active ESP server.
In addition to properties common to all instructions (for more information, see “Common Attributes of Instructions and the group Element” on page 101), the load-project instruction has the XML attributes shown in the following table:

**Table 8.14  Properties of the load-project XML Element**

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
</table>
| project-id | This attribute specifies the ID of the project to be published to the SAS Event Stream Processing factory server. The project ID is generated by SAS Event Stream Manager when a project is uploaded. The ID is available using the project-selector element, or the literal value can be obtained from the application when viewing the project metadata.  
  - Required: Yes.  
  - Valid values: A valid project ID stored internally by SAS Event Stream Manager. |
project-version

This attribute specifies the version of the project to be published to the SAS Event Stream Processing factory server.

- Required: Yes.
- Valid values: A valid project version. This value must be a number.

Project XML files created in SAS Event Stream Processing Studio 5.2 contain separate variables that correspond to a project's major version number and a project's minor version number. This attribute specifies a project's major version number and minor version number as follows:

```xml
project-version="{project.majorVersion}. {project.minorVersion}"
```

In addition, previous versions of SAS Event Stream Processing contained a project.version variable that corresponded to the project's major version number. In SAS Event Stream Processing 5.2, the project.version variable is still valid. However, it is recommended that you specify the project's major version using the project.majorVersion variable.

running-project-name

This attribute specifies the name of the SAS Event Stream Processing project that you want to use to publish the model.

- Required: Yes.
- Valid values: This value must match the regular expression `[A-z0-9_]+`.

start

This attribute specifies whether the project starts automatically.

If set to `true`, the project automatically starts upon publication. Otherwise, a manual start-project instruction is required. The default value is `true`.

- Required: No.
- Valid values: `true` or `false`. 
**start-project**

This instruction starts an already loaded SAS Event Stream Processing project.

In addition to properties common to all instructions (for more information, see “Common Attributes of Instructions and the group Element” on page 101), the start-project instruction has the XML attributes shown in the following table:

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>running-project-name</td>
<td>This attribute specifies the name of the SAS Event Stream Processing project that you want to use to publish the model.</td>
</tr>
<tr>
<td></td>
<td>Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>Valid values: This value must match the regular expression <code>[A-z0-9_]</code>+</td>
</tr>
</tbody>
</table>

**start-connectors**

This instruction starts any connectors associated with a published SAS Event Stream Processing project.

In addition to properties common to all instructions (for more information, see “Common Attributes of Instructions and the group Element” on page 101), the start-connectors instruction has the XML attributes shown in the following table:

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
</table>
**modify-project**

This instruction updates an existing running project, subject to the rules imposed by SAS Event Stream Processing.

The `modify-project` instruction uses the following SAS Event Stream Processing command to set the state of the project to modified:

```
PUT http://<server>:<port>/SASESP/projects/<projectId>/state?value=modified
```

In addition to properties common to all instructions (for more information, see “Common Attributes of Instructions and the group Element” on page 101), the `modify-project` instruction has the XML attributes shown in the following table:

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>project-id</code></td>
<td>This attribute specifies the ID of the project to be published to the SAS Event Stream Processing factory server. The project ID is generated by SAS Event Stream Manager when a project is uploaded. The ID is available using the <code>project-selector</code> element, or the literal value can be obtained from the application when viewing the project metadata. Required: Yes. Valid values: A valid project ID stored internally by SAS Event Stream Manager.</td>
</tr>
</tbody>
</table>

**running-project-name**

This attribute specifies the name of the SAS Event Stream Processing project that you want to use to publish the model.

- **Required**: Yes.
- **Valid values**: This value must match the regular expression `[A-z0-9_]`.+
**project-version**

This attribute specifies the version of the project to be published to the SAS Event Stream Processing factory server.

- **Required:** Yes.
- **Valid values:** A valid project version. This value must be a number.

Project XML files created in SAS Event Stream Processing Studio 5.2 contain separate variables that correspond to a project's major version number and a project's minor version number. This attribute specifies a project's major version number and minor version number as follows:

```xml
project-version="{project.majorVersion}.
{project.minorVersion}"
```

In addition, previous versions of SAS Event Stream Processing contained a `project.version` variable that corresponded to the project's major version number. In SAS Event Stream Processing 5.2, the `project.version` variable is still valid. However, it is recommended that you specify the project's major version using the `project.majorVersion` variable.

**running-project-name**

This attribute specifies the name of the SAS Event Stream Processing project that you want to use to publish the model.

- **Required:** Yes.
- **Valid values:** This value must match the regular expression `[A-zA-Z0-9_]`.+

**stop-project**

This instruction stops a published SAS Event Stream Processing project. Stopping a project does not remove the project, but it stops any new connections to it.
In addition to properties common to all instructions (for more information, see “Common Attributes of Instructions and the group Element” on page 101), the stop-project instruction has the XML attributes shown in the following table:

**Table 8.18  Properties of the stop-project XML Element**

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>running-project-name</td>
<td>This attribute specifies the name of the SAS Event Stream Processing project to stop.</td>
</tr>
<tr>
<td></td>
<td>■ Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>■ Valid values: This value must match the regular expression <code>[A-z0-9_] +</code>.</td>
</tr>
</tbody>
</table>

For an example of how to use the stop-project instruction, see “Example Job Template: Stop a Project” on page 74.

**unload-project**

This instruction unloads a published SAS Event Stream Processing project.

In addition to properties common to all instructions (for more information, see “Common Attributes of Instructions and the group Element” on page 101), the unload-project instruction has the XML attributes shown in the following table:

**Table 8.19  Properties of the unload-project XML Element**

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>running-project-name</td>
<td>This attribute specifies the name of the SAS Event Stream Processing project to unload.</td>
</tr>
<tr>
<td></td>
<td>■ Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>■ Valid values: This value must match the regular expression <code>[A-z0-9_] +</code>.</td>
</tr>
</tbody>
</table>

For an example of how to use the unload-project instruction, see “Example Job Template: Stop a Project” on page 74.
If a project is reported as missing (🚫 ) in the Running Projects tile, you might not be able to unload it using the unload-project instruction. For more information, see “Stop a Running Job” on page 136.

**group**

The `group` element enables you to group instructions together. Grouping instructions can make it easier to manage common dependencies. Grouping instructions also enables you to execute a job on several ESP servers.

In addition to properties common to instructions (for more information, see “Common Attributes of Instructions and the group Element” on page 101), the `group` element has the XML properties shown in the following table:

**Table 8.20** Properties of the `group` XML Element

<table>
<thead>
<tr>
<th>Child Elements</th>
<th>Any type of instruction is permitted inside a group element.</th>
</tr>
</thead>
</table>

Here is an example in which two projects are deployed to two different sets of ESP servers:

```xml
<instructions>
  <group id="startgroup1" localization-id="startgroup1"
        server-filter="filter1">
    <load-project id="load-project1"
                  localization-id="load-project"
                  project-id='{project1.id}'
                  project-version='{project1.version}'
                  running-project-name='{project1display}'
                  start=false />
    <start-project id="start-project1"
                   localization-id="start-project"
                   running-project-name='{project1}'
                   depends-on="load-project1" />
    <start-connectors id="start-connectors1"
                      localization-id="start-connectors"
                      running-project-name='{project1}'
  </group>
</instructions>
```
The first group element provides instructions for loading and starting a project and for starting connectors. When a user deploys the template, the user selects the project to deploy (project1) and enters a display name for the project (project1display). These strings must be referenced in the localization-strings element (these are not shown in the example code here). The project is deployed to a set of ESP servers that match a filter (filter1). This filter must be referenced in the server-filters element (this is not shown in the example code here).

Similarly, the second group element enables the user to select a second project to deploy to another set of ESP servers.

An element called failure-instructions is activated if any of the commands in the instructions element of the job template fail.

The failure-instructions element has the same child elements as the instructions element. That is, the failure-instructions element can contain
any instructions as well as the `group` element. For more information, see “instructions” on page 97.

**Table 8.21  Properties of the failure-instructions XML Element**

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>localization-id</td>
<td>This attribute specifies the localization string ID.</td>
</tr>
<tr>
<td></td>
<td>- Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>- Valid values: This value must exist in the default language information within the localization-strings element.</td>
</tr>
<tr>
<td>server-filter</td>
<td>This attribute specifies a filter for ESP servers. The filter resolves to a set of ESP servers to which the project will be deployed or with which the project is associated.</td>
</tr>
<tr>
<td></td>
<td>- Required: Each instruction must have a server-filter attribute. This attribute can be included in the instruction itself or in the enclosing group element or failure-instructions element.</td>
</tr>
<tr>
<td></td>
<td>A server-filter attribute on a child element overrides a server-filter attribute on a parent element. For example, a server-filter attribute on a specific instruction overrides a server-filter attribute on the parent failure-instructions element.</td>
</tr>
<tr>
<td></td>
<td>- Valid values: The ID of a server-filter element. For more information, see “server-filters” on page 95.</td>
</tr>
</tbody>
</table>
on-failure

This attribute specifies how SAS Event Stream Manager handles a failure instruction that fails to execute. By default, if an instruction fails to execute, any instructions that depend on the failed instruction are canceled. You can use the on-failure attribute to specify different behavior.

- Required: No. This attribute can be included in a specific instruction or in the enclosing group element or failure-instructions element.

An on-failure attribute on a child element overrides an on-failure attribute on a parent element. For example, an on-failure attribute on a specific instruction overrides an on-failure attribute on the parent failure-instructions element.

- Valid values: continue. Setting the on-failure attribute to continue means that dependent instructions will be executed even if the original instruction fails. If you do not want this to happen, remove the on-failure attribute.

Child Elements

The failure-instructions element has the same child elements as the instructions element. That is, the failure-instructions element can contain any instructions as well as the group element. For more information, see “instructions” on page 97.

Here is an example of a failure-instructions element:

```
<failure-instructions localization-id="recovery-steps">
  <unload-project id="unload" localization-id="unload" server-filter="filter1" project="example_proj" /></failure-instructions>
```
You can use the initialization element of the job template to alter SAS Event Stream Processing projects as the projects are deployed by using the project-transformation instruction. The transform looks for a placeholder variable in the project and replaces it with the contents of the XML element. Here is an example:

```xml
<initialization>
  <project>
    <project-transformation id="transformed-project"
      project-id="{project.id}"
      project-version="{project.version}"
    >
      <placeholder id="lang">{language}</placeholder>
    </project-transformation>
  </projects>
</initialization>
```

Variables used for `{project}` are resolved at execution when the user selects a project. This is defined in the parameters element of the job template. The placeholder element within the initialization element has an attribute `id` that is set to the value of the placeholder in the project. In the example here, the project for deployment uses the SAS Event Stream Processing Twitter adapter. The placeholder allows the user to set the language to filter on when deploying:

```xml
<expression>tw_Lang == '{lang}'</expression>
```

### Table 8.22 Properties of the project-transformation XML Element

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>id</strong></td>
<td>This attribute specifies the unique identifier of this instruction.</td>
</tr>
<tr>
<td></td>
<td>■ Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>■ Valid values: This value must match the regular expression <code>[0-9A-z][0-9A-z_]</code> +.</td>
</tr>
</tbody>
</table>
**project-id**  
This attribute specifies the ID of the project to be transformed. The ID is available using the `project-selector` element (for more information, see “project-selector” on page 86), or the literal value can be obtained from the application when viewing the project metadata.
- Required: Yes.
- Valid values: A valid project ID stored internally by SAS Event Stream Manager.

**project-version**  
This attribute specifies the version of the project to be transformed.
- Required: Yes.
- Valid values: A valid project version.

Project XML files created in SAS Event Stream Processing Studio 5.2 contain separate variables that correspond to a project's major version number and a project's minor version number. This attribute specifies a project’s major version number and minor version number as follows:

```
project-version="{project.majorVersion}.
{project.minorVersion}" 
```

In addition, previous versions of SAS Event Stream Processing contained a `project.version` variable that corresponded to the project's major version number. In SAS Event Stream Processing 5.2, the `project.version` variable is still valid. However, it is recommended that you specify the project's major version using the `project.majorVersion` variable.

### Child Elements

**placeholder**  
One or more `placeholder` elements can be defined to replace placeholders in the project.
<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>This attribute specifies the placeholder as it appears in the project.</td>
</tr>
<tr>
<td></td>
<td>- Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>- Valid values: This value must match the value of the placeholder in the project.</td>
</tr>
<tr>
<td>Content</td>
<td>This content specifies the value that the placeholder resolves to.</td>
</tr>
<tr>
<td>placeholder</td>
<td>The content can be a job template variable enclosed in braces: { and }</td>
</tr>
</tbody>
</table>
Monitor Deployments

You can use SAS Event Stream Manager to review details of the active deployments in a SAS Event Stream Processing environment. The **Deployments** page displays any active deployments in your SAS Event Stream Processing environment.

**Figure 9.1** The Deployments Page with Two Deployments

The **Deployments** page displays the following information about the deployment:

- The deployment’s health.
- The deployment’s name.
- The tags attributed to the deployment. Tags can be used to group and filter deployments. You can attribute tags to a deployment by editing the deployment. For more information, see “Edit a Deployment” on page 18.
Monitor Deployments

- Whether the deployment is a production deployment. For more information, see “Production Assets” on page 14.
- The number of ESP servers associated with the deployment.
- The date on which the deployment was created.
- The date on which the deployment was last updated.

The Health column provides a summary of the deployment’s condition. The condition of the deployment is determined by the state of its assets. This information helps you focus on deployments that have problems. The following icons can appear in the Health column:

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Idle" /></td>
<td>Idle — The deployment exists and has ESP servers associated with it, but the deployment has no running projects.</td>
</tr>
<tr>
<td><img src="image" alt="Healthy" /></td>
<td>Healthy — The running projects do not have any errors, and all ESP servers are operating normally. For more information about the health of ESP servers, see “Monitor ESP Servers” on page 128. This status is also displayed for a deployment that has no ESP servers associated with it (for example, a newly created deployment).</td>
</tr>
<tr>
<td><img src="image" alt="Warning" /></td>
<td>Warning — Some aspect of the deployment requires attention, but this situation has not resulted in an error.</td>
</tr>
<tr>
<td><img src="image" alt="Errors Reported" /></td>
<td>Errors Reported — One or more ESP servers are not healthy or one or more running projects have errors.</td>
</tr>
</tbody>
</table>

You can arrange information in the table in several ways. For more information, see “Arrange Information in Tables” on page 12.

Clicking a deployment populates the Running Projects and ESP Servers tiles. For more information about using the Running Projects tile, see “Monitor a Running Project” on page 120. For more information about the ESP Servers tile, see “Monitor ESP Servers” on page 128.
Monitor a Running Project

The **Running Projects** tile lists the running projects for a deployment or an ESP server. The following figure shows an example:

![Running Projects Table](image)

This tile is included in the following locations:

- the **Deployments** page
- the page for any specific ESP server

The following information is displayed for each project:

- the project’s status
- the name of the running project (this value is specified in the job template)
- the name of the project
- the ESP server on which the project is running
- whether updated SAS Micro Analytic Service modules are available for this project

You can arrange information in the table in several ways. For more information, see “Arrange Information in Tables” on page 12.
To view sample data for the windows contained in a running project, right-click the project and select **Open running project**. For more information, see “View Sample Data” on page 121.

To view the profile of the project, right-click the project and select **Open project**. For more information, see “View a Project” on page 31.

To accept updates to SAS Micro Analytic Service modules, right-click the project and select **Update SAS Micro Analytic Service modules**. For more information, see “Update SAS Micro Analytic Service Modules” on page 123.

For information about stopping a running project, see “Stop a Running Job” on page 136.

---

**View Sample Data**

To view sample data for a project, right-click the project in the **Running Projects** tile of the **Deployment** page, and select **Open running project** (for more information, see “Monitor a Running Project” on page 120). A new page opens, where sample data is displayed using tables, with a tab for each window. The following figure shows an example:

![Sample Data Example](image)

**Note:** SAS Event Stream Manager displays a real-time view of the data. If your data source contains a limited number of events (for example, the data source is a CSV file...
rather than a stream of data), the table will be empty after all the events have been displayed.

You can arrange information in the table in several ways. For more information, see “Arrange Information in Tables” on page 12.

You can use the **Show formatted dates** check box to control whether information appears as Coordinated Universal Time (UTC) in ISO 8601 format, for example, 2018-11-30T13:33:47.000Z. The check box is selected by default. If you clear the check box, dates appear in the format received from the ESP server instead (UNIX Epoch time).

You can use the buttons on the navigation toolbar below the table to move between pages of sample data, or to jump to a specific page of sample data. The following figure shows an example:

The following table describes the buttons on the navigation toolbar:

<table>
<thead>
<tr>
<th>Icon</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Displays the first page of sample data.</td>
</tr>
<tr>
<td></td>
<td>Displays the previous page of sample data.</td>
</tr>
<tr>
<td></td>
<td>Displays the next page of sample data.</td>
</tr>
<tr>
<td></td>
<td>Displays the last page of sample data.</td>
</tr>
</tbody>
</table>
Update SAS Micro Analytic Service Modules

SAS Event Stream Processing projects can reference models that are stored in the SAS Model Manager common model repository. When a project is deployed, the model is retrieved from the SAS Model Manager common model repository and written to the ESP server. SAS Micro Analytic Service modules are used to accommodate such models. These models can be subsequently updated if a new champion model is declared. If your running project in SAS Event Stream Manager references a model stored in the SAS Model Manager common model repository, you can update the running project to reflect updates to the model.

If an update is available, this is indicated in the Running Projects tile and in a notification bar on the Deployments page. If you want to update just one project, the quickest method is to apply the update from the Running Projects tile. If you want to update multiple projects, you can apply the update from the notification bar on the Deployments page. The notification bar can also be used to update all projects that are running the specified version of the model.

Update a Project

If an update is available, the Update Available column in the Running Projects tile displays ✔. The following figure shows an example:
To accept the update, right-click the project and select **Update SAS Micro Analytic Service modules**. The Log page opens and SAS Event Stream Manager runs a job to fetch and deploy all new champion models for the project.

SAS Event Stream Manager updates the SAS Micro Analytic Service store with the model content and then replaces the SAS Micro Analytic Service module that the project is using. The schemas for the existing windows in the project are not altered during this operation. The new champion model must contain function input and output variables that are compatible with the existing schema.

The minor version of the project is updated, for example, the project version is updated from 1.2 to 1.3.

**Update Multiple Projects**

If an update is available, a notification bar is displayed on the **Deployments** page. The following figure shows an example:
To open the currently deployed version of the project, click **Open Project**.

To hide the notification bar temporarily, click **Ignore**. The notification bar is hidden until you return to the **Deployments** page.

To accept the update for one or more projects that are running the specified version of the model:

1. Click **Deploy** on the notification bar.
   
   The Update SAS Micro Analytic Service modules window appears.

2. Click **Select projects**.
   
   A drop-down list is displayed. Click the drop-down list to display all the projects that reference the updated model. For each project, the ESP server that the project is currently deployed to is shown in brackets. The following figure shows an example:
3 In the drop-down list, select a project to update.
   The project is added to the list of projects to be updated.

4 If required, select an additional project from the drop-down list. Repeat until you have selected all projects that you want to update. The following figure shows an example:
5 Click **OK**.

SAS Event Stream Manager runs a job to fetch and deploy the new champion model for the selected projects. To view the status of this job, open the **Log** page.

SAS Event Stream Manager updates the SAS Micro Analytic Service store with the model content and then replaces the SAS Micro Analytic Service module that each selected project is using. The schemas for the existing windows in the project are not altered during this operation. The new champion model must contain function input and output variables that are compatible with the existing schema.

The minor version of the project is updated, for example, the project version is updated from 1.2 to 1.3.

### Update All Projects

If an update is available, a notification bar is displayed on the **Deployments** page. The following figure shows an example:

![Deployment Example](image)

To open the currently deployed version of the project, click **Open Project**.
To hide the notification bar temporarily, click Ignore. The notification bar is hidden until you return to the Deployments page.

To accept the update for all projects that are running the specified version of the model:

1. Click Deploy on the notification bar.
   The Update SAS Micro Analytic Service modules window appears.

2. Select All projects running this version.

3. Click OK.
   SAS Event Stream Manager runs a job to fetch and deploy the new champion model for all projects that are currently running the specified version of the model. To view the status of this job, open the Log page.

   SAS Event Stream Manager updates the SAS Micro Analytic Service store with the model content and then replaces the SAS Micro Analytic Service module that the project is using. The schemas for the existing windows in the project are not altered during this operation. The new champion model must contain function input and output variables that are compatible with the existing schema.

   The minor version of the project is updated, for example, the project version is updated from 1.2 to 1.3.

Monitor ESP Servers

You can use the ESP Servers page to view all ESP servers that SAS Event Stream Manager has detected, and which are available to be assigned to a deployment. ESP servers that already belong to a deployment are not displayed. The following figure shows an example:
The **ESP Servers** page displays ESP servers that SAS Event Stream Manager has detected automatically. SAS Event Stream Manager communicates with these ESP servers using agents. The **ESP Servers** page also displays any ESP servers to which you have connected directly (without using an agent). For more information, see “Connecting Directly to an ESP Server” on page 24. The **ESP Servers** page does not indicate which ESP servers were detected automatically and which ones you connected to directly.

The **ESP Servers** page displays the following information for each server or device defined as an ESP server:

- The ESP server’s health.
- The ESP server’s name.
- The tags attributed to the ESP server. Tags can be used to group and filter ESP servers. You can attribute tags to an ESP server by editing the ESP server. For more information, see “Edit an ESP Server” on page 25.
- The host on which the ESP server is running.
- The port for HTTP administration requests and for the HTTP publish/subscribe server.
- The SAS Event Stream Processing version installed on the host on which the ESP server is running.
- Whether a SAS Event Stream Processing Analytics license is installed on the host. To deploy a project that contains SAS Event Stream Processing Analytics windows, an appropriate license must be available.
Whether SAS Event Stream Processing has been enabled to meter the number of events that are processed on the ESP server.

The deployments that the ESP server belongs to.

The Health column provides a summary of the ESP server’s condition. The condition of the ESP server is determined by CPU, disk, and RAM usage. This information helps you focus on those ESP servers that have problems. The following icons can appear in the Health column:

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Healthy" /></td>
<td>Healthy — CPU, disk, and RAM usage are acceptable.</td>
</tr>
<tr>
<td><img src="image" alt="Warning" /></td>
<td>Warning — CPU, disk, or RAM usage is elevated.</td>
</tr>
<tr>
<td><img src="image" alt="Errors Reported" /></td>
<td>Errors Reported — CPU, disk, or RAM usage is very high or the ESP server is not available.</td>
</tr>
</tbody>
</table>

**Note:** If SAS Event Stream Manager connects directly to an ESP server rather than through an agent, the status of the ESP server is reported as healthy if the ESP server is running.

You can arrange information in the table in several ways. For more information, see “Arrange Information in Tables” on page 12.

Clicking an ESP Server populates the Running Projects tile with information relating to the ESP server’s running projects. For more information about using the Running Projects tile, see “Monitor a Running Project” on page 120.

To view the ESP server’s profile details, right-click the ESP server and select Open ESP server.
Monitor Jobs

View All Jobs

The Log page displays the status of active and historical jobs, except for any jobs that you have deleted from the page. You can use this information to help resolve problems with your deployments. The following information is displayed for each job:

- the name of the job template from which the job was created
- the name of the deployment that the job template belongs to
- the user who deployed the job template
- the time at which the job template was deployed
- the time at which the job was completed
- the time it took for all the tasks in this job to be completed (whether they were completed successfully or not)
- the status of the job
- the job’s progress (specified as a percentage)
The table is refreshed automatically. You can also refresh the table manually: click .

If the table contains too much information, you can arrange the information in several ways to find the jobs that you are interested in. For more information, see “Arrange Information in Tables” on page 12. You can also delete jobs from the table. For more information, see “Delete a Job” on page 134.

To view more information for a specific job, select the job in the table at the top of the page. The bottom pane displays the following information:

- The Progress tab shows how many tasks were completed and whether tasks were completed on all relevant ESP servers.
- The Failed Tasks tab lists tasks that were not completed.
- The Parameters tab lists parameters in the job template along with the value entered by the user for each parameter.
View Job Details

To view further job details, select a job in the main table on the Log page and click . The job appears on a separate page. The following figure shows an example:

**Figure 9.3  An Open Job**

The top of the page shows a summary, with information about how many tasks were completed and whether tasks were completed on all relevant ESP servers.

The table on this page shows all tasks for the job, in the order in which the instructions are executed. You can expand items in the Task column to view the required level of detail. In the example here, the Loading instructions and Unloading instructions rows relate to instruction groups in this specific example. Load Project, Start Project, and similar rows relate to individual instructions in this job template. The job template
was deployed to three ESP servers, and the **Load Project** row has been expanded to show the result for each ESP server.

You can also use the following check boxes to filter the contents of the table: **Running**, **Not started**, **Completed**, and **Completed with failures**.

---

**Delete a Job**

Deleting a job removes it from the main table on the **Log** page. Deleting jobs can be useful if the table contains jobs that are no longer relevant.

Deleting a job affects only the information that appears in the table; it does not affect job execution. Nevertheless, you cannot delete a job that is still running.

1. Do one of the following:
   - To delete a specific job from the **Log** page, select the job and click .
   - To delete several jobs, click and select **Delete all canceled jobs**, **Delete all failed jobs**, or **Delete all successful jobs**.

   The Delete Job window appears.

2. Click **OK**.

---

**Rerun a Job**

If a job fails to complete successfully, you can run it again. For example, a job might not complete successfully because an ESP server was unavailable or a data file was not present. After you have resolved the problem, you can rerun the job quickly on the **Log** page rather than deploying the job template again. You can also rerun jobs that completed successfully.

Rerunning a job creates a new job, as opposed to starting the original job again.
To rerun a job:

1. On the Log page, select a job and click [ ].

   The Rerun Job window appears.

2. In the Strategy field, select your preferred option:
   - **Rerun all instructions**
   - **Rerun all instructions on servers that failed to complete successfully**
   - **Save failed servers to a filter**

   Select this option to create a filter that references the failed ESP servers. If you select this option, you must enter a name in the Filter name field. You can then reference this filter in another job template that addresses errors or cleans up ESP servers. For example, consider a situation where a project was loaded but a connector was not started because files were not available. A job template that addresses this error might include the `start-connectors` instruction. Consider another situation where a project could not be loaded because it had already been loaded. A cleanup job template might contain the `unload-project` and `stop-project` instructions.

   If the original job template needs to be run instead, use the **Rerun all instructions on servers that failed to complete successfully**.

   For more information about how you can use filters, see “Filters” on page 41.

3. If required, select **Skip failed instructions**.

4. Click **OK**.

   SAS Event Stream Manager reruns the job. The result appears on a new row in the table on the Log page.
Stop a Running Job

Stopping a running job can be helpful if, for example, you want to stop a job that is taking a long time to complete. Stopping a running job means canceling the execution of instructions that have not yet been executed. That is, when you stop a running job, the instruction that is currently being processed will be executed, but any subsequent instructions will not be executed.

Stopping a running job does not stop a running project. For more information, see “Stop a Running Project” on page 136.

To stop a running job:

1. Select the job in the main table of the Log page and click Stop. The Cancel Job window appears.

2. Click OK.

Stop a Running Project

Stopping and unloading a project is a SAS Event Stream Processing operation. To use SAS Event Stream Manager to stop and unload a project, you must deploy a job template that contains instructions for stopping a project and unloading it from the associated ESP server. For an example of such a job template, see “Example Job Template: Stop a Project” on page 74.

When a project has stopped, the Running Projects tile on the Deployments page displays the project’s status as stopped. When a project has been unloaded, it no longer appears in the Running Projects tile.
If a project is reported as missing (🚫) in the **Running Projects** tile, you might not be able to unload it using an instruction. A project might be reported as missing if it was uploaded to SAS Event Stream Manager but later deleted by some other means than by using the SAS Event Stream Manager user interface. To clean up a missing project:

1. On the **ESP Servers** page, right-click the ESP server that contains the affected project.
   
   A page displaying the ESP server’s details opens.

2. In the **Running Projects** tile, right-click the affected project and select **Clean up missing project**.
   
   The affected project no longer appears in the **Running Projects** tile.
SAS Event Stream Manager enables you to monitor your metering servers. This ensures that your production ESP servers are in compliance with the terms of your software license. You must run at least one metering server to track usage data (that is, event counts) on your production ESP servers.

You do not need to track events on development servers because they do not contribute to the event volume assigned to your software license.

To monitor an ESP metering server, you must add it to the list of defined metering servers in SAS Event Stream Manager.

To add a metering server:
1 On the **Metering** page, click ![New](new.png). The Metering Server Properties window appears.

2 In the **Host** field, enter the host name of the server containing the metering server.

3 In the **Metering port** field, either accept the default value (31001) or replace it with the network port defined on the metering server.

4 Click **OK**.

The metering server is displayed on the **Metering** page, along with any other metering servers that were added previously.

---

**Monitor a Metering Server**

**View Metering Server Details**

To view details about a metering server, select the required metering server on the **Metering** page.

**Note:** If the metering server that you want to monitor is not listed, you must add it to SAS Event Stream Manager. For more information, see “Add a Metering Server” on page 139.

The Metering Server window displays each license associated with the selected metering server, and the total number of events recorded for that license for each year. The following figure shows an example:
Group Metering Servers

You can arrange the information in the table on the **Metering** page in several ways. However, you might find it particularly helpful to group the table by the **Host** column, so that all ESP metering servers on a particular host are grouped together.

For more information, see “Arrange Information in Tables” on page 12.
Overview of the Stock Trade Example

This chapter provides an end-to-end example of how SAS Event Stream Manager works. By following this example, you learn the following tasks:

- create a deployment
- associate an ESP server with the deployment
- upload a project and a job template
- deploy the job template
monitor the deployment

stop a running job

The example uses the five files listed below. Four of these files are provided for you by SAS (for more information, see “Prepare the Example Files for Use” on page 148), and one file is created when you deploy the job template.

- trades_connector_stocksymbol_project.xml is the project used in the example. For more information about the content of this project, see “The Stock Trade Project” on page 144.
- trades_connector_stocksymbol_job_template.xml is a job template that loads and starts a project. This job template is discussed in detail earlier in this guide. For more information about the content of this job template, see “Example Job Template: Stock Trade” on page 64.
- trades1M.csv contains stock trade data used as input events in this example.
- placeholder_filtered_trades.csv is an output file. When you deploy the job template, the project processes stock trades and filters out trades that match a specified stock code. The project then writes these trades into the placeholder_filtered_trades.csv file.
- stop_project_job_template.xml is a job template that you can deploy to stop the running project when you have finished exploring this example. This job template is discussed in detail earlier in this guide. For more information about the content of this job template, see “Example Job Template: Stop a Project” on page 74.

The Stock Trade Project

The trades_connector_stocksymbol_project.xml file contains three windows:

- The **source_win** window is a source window. This window is where trades data from the trades1M.csv file enters the model.
- The **symcode_filter** window is a filter window. It contains a filter expression that identifies events with a specified stock symbol code.
The aggFromFilt window is an aggregate window. This window places events into aggregate groups and calculates the weighted average price, maximum price, minimum price, and standard deviation. This window also writes the results in the placeholder_filtered_trades.csv file.

**Figure 11.1** Diagram of the Stock Trade Project

Here is the project XML:

```xml
<project name="filteredtrades" pubsub="auto" threads="8">
    <description/>
    This project finds trades that match a specified stock code.
    </description>
    <contqueries>
        <contquery name="cq">
            <windows>
                <window-source name="source_win" index="pi_EMPTY" insert-only="true" collapse-updates="true">
                    <schema>
                        <fields>
                            <field name="ID" type="int32" key="true"/>
                            <field name="symbol" type="string"/>
                            <field name="currency" type="int32"/>
                            <field name="udate" type="int64"/>
                        </fields>
                    </schema>
                </window-source>
            </windows>
        </contquery>
    </contqueries>
</project>
```
<field name="msecs" type="int32"/>
<field name="price" type="double"/>
<field name="quantity" type="int32"/>
<field name="venue" type="int32"/>
<field name="broker" type="int32"/>
<field name="buyer" type="int32"/>
<field name="seller" type="int32"/>
<field name="buysellflg" type="int32"/>
<field name="time" type="stamp"/>
</fields>
</schema>
<connectors>
<connector name="csv_data_provider" class="fs">
<properties>
<property name="type">
<![CDATA[pub]]>
</property>
<property name="fstype">
<![CDATA[csv]]>
</property>
<property name="fsname">
<![CDATA[/data/input/trades1M.csv]]>
</property>
<property name="transactional">
<![CDATA[true]]>
</property>
<property name="blocksize">
<![CDATA[1]]>
</property>
<property name="dateformat">
<![CDATA[%Y-%m-%d %H:%M:%S]]>
</property>
</properties>
</connector>
</connectors>
<window-source>
<window-filter name="symcode_filter">
<description>
This is a Filter window with a filter expression to get events that have a specified stock symbol code. The Filter window has the same schema as the input window. As a result, there is no schema defined for the Filter window.
</description>
<expression>symbol == '{symcode}'</expression>
</window-filter>
<window-aggregate name="aggFromFilt" collapse-updates="true">
<schema>
  <fields>
    <field name="symbol" type="string" key="true"/>
    <field name="awap" type="double"/>
    <field name="minPrice" type="double"/>
    <field name="maxPrice" type="double"/>
    <field name="std" type="double"/>
  </fields>
</schema>

<output>
  <field-expr>
    <![CDATA[ESP_aWAve(quantity,price)]]>
  </field-expr>
  <field-expr>
    <![CDATA[ESP_aMin(price)]]>
  </field-expr>
  <field-expr>
    <![CDATA[ESP_aMax(price)]]>
  </field-expr>
  <field-expr>
    <![CDATA[ESP_aStd(price)]]>
  </field-expr>
</output>

<connectors>
  <connector name="Output_CSV" class="fs">
    <properties>
      <property name="type">
        <![CDATA[sub]]>
      </property>
      <property name="fstype">
        <![CDATA[csv]]>
      </property>
      <property name="fsname">
        <![CDATA[/data/placeholder_filtered_trades.csv]]>
      </property>
      <property name="snapshot">
        <![CDATA[true]]>
      </property>
    </properties>
  </connector>
</connectors>

<edges>
  <edge source="source_win" target="symcode_filter"/>
  <edge source="symcode_filter" target="aggFromFilt"/>
Stock Trade Example Steps

Prepare the Example Files for Use

1. Download the SAS Event Stream Manager examples package from SAS Event Stream Manager examples.

2. Save the following files, contained in the examples ZIP file, to a temporary location on your computer:
   - trades_connector_stocksymbol_project.xml
   - trades_connector_stocksymbol_job_template.xml
   - stop_project_job_template.xml

3. Locate the trades1M.csv file, which is provided within your SAS Event Stream Processing installation. This file is typically in the following location: /opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/vwap_xml. Replace <release> with the release number in your SAS Event Stream Processing installation directory path.

4. Update the trades_connector_stocksymbol_project.xml file to point to the location of the trades1M.csv file:
   a. Open the trades_connector_stocksymbol_project.xml file in a text editor.
   b. Locate the following line (for example, by searching):
      ```xml
      <![CDATA[/data/input/trades1M.csv]]>
      ```
   c. Update the directory path to point to the location of the trades1M.csv file on your SAS Event Stream Processing server.
d  Locate the following line (for example, by searching):

<![[CDATA[/data/placeholder_filtered_trades.csv]]]>

e  Update the directory path to point to a location on your SAS Event Stream Processing server where the output file can be written.

f  Save the trades_connector_stocksymbol_project.xml file.

Create the Stock Trade Deployment

1  On the **Deployments** page, click ![New](https://example.com/new).

   The Deployment Properties window appears.

2  In the **Name** field, enter *Stock Trade*.

3  In the **Description** field, enter *A deployment for processing stock trades*.

4  In the **Tags** field, enter *stock*.

5  Click **OK**.

The **Stock Trade** deployment appears on a new page:
Add an ESP Server

You must associate an ESP server with the Stock Trade deployment.

**Note:** Ensure that you have an ESP server that you can use for this example. You can view available ESP servers that SAS Event Stream Manager has detected on the ESP Servers page. SAS Event Stream Manager detects some ESP servers automatically and communicates with these ESP servers using agents. You can connect directly to an ESP server (without using an agent). In this case, SAS Event Stream Manager becomes aware of the ESP server only after you have connected. For more information, see “Connecting Directly to an ESP Server” on page 24.
To add an ESP server to the Stock Trade deployment:

1. Click 🔄. The Add and Remove ESP Servers window appears.

2. Move the desired ESP server from the Available servers table to the Selected servers table.

3. Click OK.

The ESP server appears in the table on the Stock Trade deployment page:

```
<table>
<thead>
<tr>
<th>Status</th>
<th>Name</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ESP_Server_1</td>
<td>server3.example.com</td>
</tr>
</tbody>
</table>
```

**Identification**

ID: 60cdd7c1-83ba-4596-84da-b693f843266e

Name: *Stock Trade*

Production deployment: False

Description: A deployment for processing stock trades

Tags:

- stock
- New tag
Upload the Stock Trade Project

1. On the Projects page, click [Upload].

   The Upload Project window appears.

2. Fill out the fields as follows:
   - **File**: Navigate to the trades_connector_stocksymbol_project.xml file.
   - **Name**: This field is automatically set to filteredtrades when you set the File field.
   - **Description**: This field is automatically set to a description that is extracted from the trades_connector_stocksymbol_project.xml file.
   - **Tags**: Enter stock.
   - **Production project**: Leave this check box deselected.
   - **Version notes**: Enter First version.

3. Click OK.

   The filteredtrades project appears on the Projects page:

   ![Projects Page](image)

<table>
<thead>
<tr>
<th>Name</th>
<th>Tags</th>
<th>Version</th>
<th>Production</th>
<th>Created</th>
<th>Updated</th>
</tr>
</thead>
<tbody>
<tr>
<td>filteredtrades</td>
<td>stock</td>
<td>1.0</td>
<td>×</td>
<td>15/05/2018, 17:00:54</td>
<td>15/05/2018, 17:00:54</td>
</tr>
</tbody>
</table>

   **Note:** To explore the project's contents, right-click the filteredtrades project and select Open project. The project opens and you can click the Details, XML, Diagram, and Versions tabs to explore their contents.
Upload the Stock Trade Job Template

1. On the **Job Templates** page, click ![Upload job template](image) and select **Upload job template**.

   The Upload Job Template window appears.

2. Fill out the fields as follows:
   - **File**: Navigate to the trades_connector_stocksymbol_job_template.xml file.
   - **Tags**: Enter **stock**.
   - **Production template**: Leave this check box deselected.
   - **Version notes**: Enter **First version**.

3. Click **OK**.

   The Filter Trades by Stock Code job template appears on the **Job Templates** page:

   ![Stock Trade Example Steps](image)

   **Note**: To explore the job template’s contents, right-click the **Filter Trades by Stock Code** job template and select **Open**. The job template opens and you can click the **Details**, **XML**, and **Versions** tabs to explore their contents.

Deploy the Stock Trade Job Template

1. On the **Job Templates** page, select the **Filter Trades by Stock Code** job template and click ![Deploy](image).
The Job Template ‘Filter Trades by Stock Code’ v.1 window appears.

2 Fill out the fields as follows:

- **Deployment**: Select **Stock Trade**.
- **Project**: Select **filteredtrades**.
- **ESP server**: Select your ESP server.
- **Stock symbol code**: Select **Micro Focus**.

When the job template is deployed, trades for Micro Focus stock are identified with the **MCRO** stock code.

**Note**: The **Deployment** field always appears in this window. The other fields appear because they are specified in the job template parameters.

3 Click **OK**.

The **Log** page appears. The job created from the **Filter Trades by Stock Code** job template appears on the page.

4 Select the newly created job.

The bottom tile is populated. When all tasks have completed successfully, the overall job status is displayed as ✔️ in the main table.
Note: If you want to view the parameters in the job template along with the values that you entered for each parameter, click the Parameters tab.

Monitor the Deployment

1. On the Deployments page, select the Stock Trade deployment.

2. In the Running Projects tile, right-click the filteredtrades project and select Open running project.

   The MCRO_1 page appears.

3. Select each tab in turn to view the results for each window:
   - The source_win tab displays a snapshot of input events, as shown in the following figure:
**Note:** SAS Event Stream Manager displays a real-time view of the data. When all the events in the CSV file have been displayed, the table will be empty.

- The **symcode_filter** tab lists trades for the selected stock code, **MCRO**. When you view this tab, the table in it is likely to be empty. This is because SAS Event Stream Manager displays a real-time view of the data and there is a limited number of events with the **MCRO** stock code. That is, the table is not intended to collate all events with the **MCRO** stock code.

- The **aggFromFilt** tab displays aggregated results. When you view this tab, the table in it is likely to be empty. This is because SAS Event Stream Manager displays a real-time view of the data, as explained previously. However, you can see the results appear in a CSV file on your server. Open the **placeholder_filtered_trades.csv** file on your SAS Event Stream Processing server:
If you want to see results for the other two stocks, deploy the project again and select a different stock.

**Stop the Stock Trade Job**

When you have finished exploring this example, you can stop the running job by deploying another job template file, stop_project_job_template.xml. Deploying this job template stops the selected project and unloads it from the ESP server.

1. On the **Job Templates** page, click 🔄.
   
   The Upload Job Template window appears.

2. In the **File** field, select the stop_project_job_template.xml file.

3. Click **OK**.
   
   The **Details** tab of the **Remove a Running Project v.1 job template** appears.

4. Click 🔄.
   
   The Job Template ‘Remove a Running Project’ v.1 window is displayed.

5. Fill out the fields as follows:
   
   - **Deployment**: Select **Stock Trade**.
   
   - **ESP server**: Select your ESP server.
Project to unload: Select MCRO_1.

Note: If you selected a different stock code when exploring the example, select the code for that stock instead.

6 Click OK.

The Log page appears. The job created from the Remove a Running Project job template appears on the page.

7 Select the newly created job.

The Tasks tile is populated. This job template has two tasks: Stop Project and Unload Project. When all the tasks in the Tasks tile have completed successfully, the overall job status is displayed as .
Recommended Reading

SAS Event Stream Processing documentation

For a complete list of SAS publications, go to sas.com/store/books. If you have questions about which titles you need, please contact a SAS Representative:

SAS Books
SAS Campus Drive
Cary, NC 27513-2414
Phone: 1-800-727-0025
Fax: 1-919-677-4444
Email: sasbook@sas.com
Web address: sas.com/store/books
adapter
software that publishes event streams into a SAS Event Stream Processing engine and subscribes to event streams from engine windows.

agent (event stream agent)
a small executable program that passes information between SAS Event Stream Manager and an ESP server. Agents relay operational metrics from ESP servers to SAS Event Stream Manager, and perform actions on ESP servers in response to the commands received from SAS Event Stream Manager.

asset (event stream asset)
an identifiable part of a deployment that can be modified and monitored by SAS Event Stream Manager. Examples include projects and ESP servers.

continuous query
a container that holds a directed graph of windows. This graph enables you to specify the connectivity between windows.

data flow (flow)
a data processing model that can be executed on a SAS Event Stream Processing engine.

deployment
a logical grouping of assets as a single unit for lifecycle management and monitoring in SAS Event Stream Manager.
derived window
a window that displays events that have been fed through other windows and that performs computations or transformations on these incoming events.

directed graph
a set of nodes connected by edges, where the edges have a direction associated with them.

engine (events engine)
the top level container in an event stream processing model. See also agent.

ESP
See event stream processing.

ESP server
See event-stream processing server.

event block
a grouping or package of events with a unique ID for use in a continuous query.

event stream
a continuous flow of event blocks.

event stream agent
See agent.

event stream asset
See asset.

event stream processing (ESP)
a process that enables real-time decision making by continuously analyzing large volumes of data as it is received.

event-stream processing server (ESP server)
an engine-executable program that instantiates and executes projects.
**events engine**

See engine.

**events project**

See project.

**factory server**

A server for factory objects that control the creation of other objects, access to other objects, or both.

**flow**

See data flow.

**job**

A collection of tasks that can create output.

**job template**

An XML file that contains instructions for creating a job.

**memory depot**

A repository for indexes and event data that is used by a project.

**modeling API**

An application programming interface that enables developers to write event stream processing models.

**opcode**

An instruction that specifies an action to be performed.

**operation code**

An instruction that specifies an action to be performed.

**project (events project)**

A container that holds one or more continuous queries and is backed by a thread pool of user-defined size. See also continuous query.
publish/subscribe API
a library that enables you to publish event streams into an event stream processor, or to subscribe to event streams, within the event stream processing model. The publish/subscribe API also includes a C and JAVA event stream processing object support library.

Source window
a window that has no windows feeding into it and is the entry point for publishing events into the continuous query.

stream
a sequence of data elements [that are] made available over time.

thread pool
a set of threads that can be used to execute tasks, post work items, process asynchronous I/O, wait on behalf of other threads, and process timers.

window
a processing node in an event stream processing model. Source and derived windows can perform aggregations, computations, pattern matching, and other operations.
Gain Greater Insight into Your SAS® Software with SAS Books.

Discover all that you need on your journey to knowledge and empowerment.

support.sas.com/bookstore for additional books and resources.