THE POWER TO KNOW.

SAS® Information Retrieval Studio 12.2

Quick Start Guide

SAS® Documentation

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2013. *SAS® Information Retrieval Studio 12.2: Quick Start Guide*. Cary, NC: SAS Institute Inc.

SAS® Information Retrieval Studio 12.2: Quick Start Guide

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st electronic book, March 2013

SAS provides a complete selection of books and electronic products to help customers use SAS software to its fullest potential. For more information visit **support.sas.com/bookstore** or call 1-800-727-3228.

Contents

About This Book

How This Quick Start Guide Can Help You

The *SAS Information Retrieval Studio: Quick Start Guide* provides basic configurations, matcher examples, and sample multi-process configurations. (These instructional examples are subject to change.) Each of the sample pipeline projects is designed to jump-start your development efforts for specific, but common, scenarios.

The samples in this book are designed to be used for the following purposes:

- meet a specific set of requirements
- follow best practices
- see simple, easy-to-follow examples that omit details. These particulars include both obvious and complex operations:
  - *Obvious operations* assumes standard, post-beginner knowledge.
  - *Complex operations* assumes the ability to go beyond the simple example, using both interface exploration and Help provided.

For all of these reasons, the sample configurations that are provided are context specific, although generalizable, and do not include either step-by-step directions or screenshots.

You can use the configuration examples as they are explained. Alternatively, choose to modify these examples to meet the requirements of your organization.

What You Should Know in Order to Use This Quick Start Guide

SAS Information Retrieval Studio: Quick Start Guide assumes that the following statements are true:

- You understand how to use a search interface.
- You can perform basic search operations.

- You are familiar with indexes.
- You have a basic understanding of how to use the various components that are required. These components include crawlers, document processors, indexes, and search interfaces.

In other words, *SAS Information Retrieval Studio: Quick Start Guide* assumes that you are not a beginner in developing search projects. For this reason, use the provided example.

Prerequisites

Here are the prerequisites for using SAS Information Retrieval Studio:

- SAS Information Retrieval Studio installed on your machine
- A supported browser installed on your desktop client
- Access to data sources
- (Optional) Rules such as category rules and concept definitions created in other SAS applications

If you have any questions about whether you are ready to use SAS Information Retrieval Studio, contact your system administrator.

Conventions

This manual uses the following typographical conventions:

Convention	Description
`TGM_ROOT`	The root directory where SAS Information Retrieval Studio is installed, typically the following: **Windows:** `C:/Program Files/SAS Information Retrieval Studio` **UNIX:** `/opt/SAS Information Retrieval Studio`
`.xml`	Code examples are shown in a fixed-width font.

Convention	Description
Start button	The labels for user interface controls are shown in a bold, sans-serif font.
www.sas.com	The hypertext links are shown in a light blue, fixed-width font, and are underlined.

1 About SAS Information Retrieval Studio

1.1 What Is SAS Information Retrieval Studio?

In order to locate much of the information that businesses require today, it is necessary to gather documents. Normalize the text in these input documents and then choose to process, index, and export documents. In an environment where data, and its types, grow exponentially there is also a need to automate these related processes.

SAS Information Retrieval Studio is an application that anyone can use to locate documents on the web, feeds, or in file systems. This application combines several key technologies to provide a comprehensive solution to data collection, indexing, searching, and so on. These technologies are bundled into one customizable product.

Use SAS Information Retrieval Studio to perform tasks such as gathering documents from locations such as the web or your local machine. Normalize and process the input text before indexing for search or outputting the input text in a file format such as XML or CSV.

1.2 The Benefits for You

SAS Information Retrieval Studio provides the following benefits:

Retrieve information easily

The web, feed, and file crawlers gather the documents that you specify according to your parameters. Documents are chunks of text, with or without markup tags, gathered from the Internet, feeds, and databases. These chunks of text can be treated by the document processors that parse, convert, categorize, extract concepts, and so on. The documents can then be sent to the index or to another program. If indexed, the documents can be searched by your end users.

Build a custom information retrieval pipeline

Choose to build an information retrieval system that is customized to meet the needs of your organization. You can choose all, or some, of the following components:

Crawlers

Choose the web, feed, or file crawlers to gather documents from the web, from feeds, and from file systems, respectively.

Pipeline server

Choose your document processors that parse, categorize, extract concepts, convert documents into text, and so on. These processors can also hand the gathered documents to other applications such as SAS Sentiment Analysis Workbench.

Indexing server

Choose how, and whether, input documents are indexed. End users can search indexed documents using a customized search window that runs on the query web server.

Query web server

Specify how the matching documents are returned in the search window, the appearance of this window, and how end users can navigate the returns.

Query statistics server

See the counts for the entered queries according to various time frames.

Use both global and project level operations

Specify settings for the Markup Matcher and Document Conversion servers at the global level. Create an individual project in order to specify pipelines that can use these servers to extract specific types of data from input documents.

Customize components easily

Easy-to-use windows and wizards simplify the process of customizing the information retrieval components that you choose. These panes also provide log files, statistics, information about the processes involved, and data on documents in the pipeline.

Integrate with other SAS products

SAS Information Retrieval Studio includes functionalities that are designed to work with other SAS applications. For example, apply category, concept, and contextual extraction rules through the document processor using rules developed in SAS Enterprise Content Categorization Studio.

Empower business owners by locating data

SAS Information Retrieval Studio enables you to configure processes according to your organization's requirements. Use this program to locate, process, index, and customize a search window for your data. See various types of informational statistics.

Improve the business value of IT and the corporate data that it manages

SAS Information Retrieval Studio provides you with easy, self-service access to the information contained in your documents. Use SAS Information Retrieval Studio to locate, process, index, and search your data.

1.3 How Does SAS Information Retrieval Studio Work?

SAS Information Retrieval Studio contains web, feed, and file crawlers that can gather documents from a specified location. The text in these documents is

normalized and then processed before being indexed or sent to another application. All of these processes are optional. The user specifies the components to use, configures these components, and can enable end users to perform facetted search using labels.

1.4 How Does SAS Information Retrieval Studio Fit into the SAS Product Line?

As an integral part of the SAS product line, SAS Information Retrieval Studio provides crawlers, indexing, and searching capabilities. These functionalities facilitate the processes of information retrieval and management. Use these capabilities with the following SAS products, among others:

Export document collections to SAS Sentiment Analysis Workbench and SAS Enterprise Miner

Export the files that the web crawler gathers in SAS Information Retrieval Studio to SAS Sentiment Analysis Workbench. Here you can see reports about overall sentiment. Analysts can also see and review individual documents in SAS Sentiment Analysis Workbench. You can also export files to SAS Enterprise Miner to locate topics and themes in your input documents.

Categorize and extract concepts

SAS Information Retrieval Studio enables you to apply the rules defined in SAS Content Categorization Studio to your gathered documents.

Extract text from XML and HTML files

Use the SAS Markup Matcher Server to extract text from `.xml` and `.html` input documents in a customizable manner.

Convert files

Use the Document Conversion Server to extract text from input files such as Adobe PDF and Microsoft Office.

1.5 How to Get Help for SAS Information Retrieval Studio

Go to **Start > SAS Information Retrieval Studio > Release Notes**.

1.6 What is a Document?

A document consists of a unit of text. For example, a document can be any of the following:

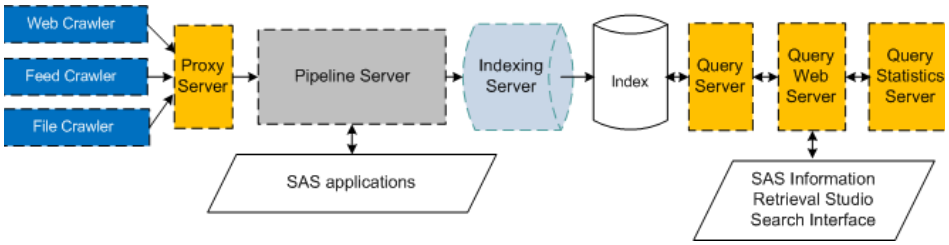
- an HTML page
- an XML page
- a Microsoft Word file
- a PDF file
- one row in a CSV file or a database
- one article or summary in a feed

In SAS Information Retrieval Studio, each document is represented as a configurable set of fields. Each file has a name and a value. Unnecessary fields can be either left empty or omitted from the document.

1.7 Architecture

The architecture diagram that is shown below provides an overview of the application processes that you can choose to use in your customized configuration. Some, but not all, of these components are covered in this book.

Figure 1-1 SAS Information Retrieval Studio



2
Configuration Basics

- *Developing Configurations*
- *Configuring Gathering Using Crawlers*
- *Configuring Normalizing Processors*
- *Configuring Processing Components*
- *Configuring Indexing Operations*
- *Configuring Exporting Processors*

2.1 Developing Configurations

SAS Information Retrieval Studio lets you gather, normalize, process, index, and export documents. After you install and access SAS Information Retrieval Studio, choose and configure the components that run simultaneously to meet your requirements.

In SAS Information Retrieval Studio, a document is defined as a unit of textual data. For example, a document can be an HTML page, a Microsoft Word file, a PDF file, or one row in a CSV file or a database.

When you define how documents flow through SAS Information Retrieval Studio, you specify a series of configurations that run simultaneously on servers. For this reason, follow these best practices:

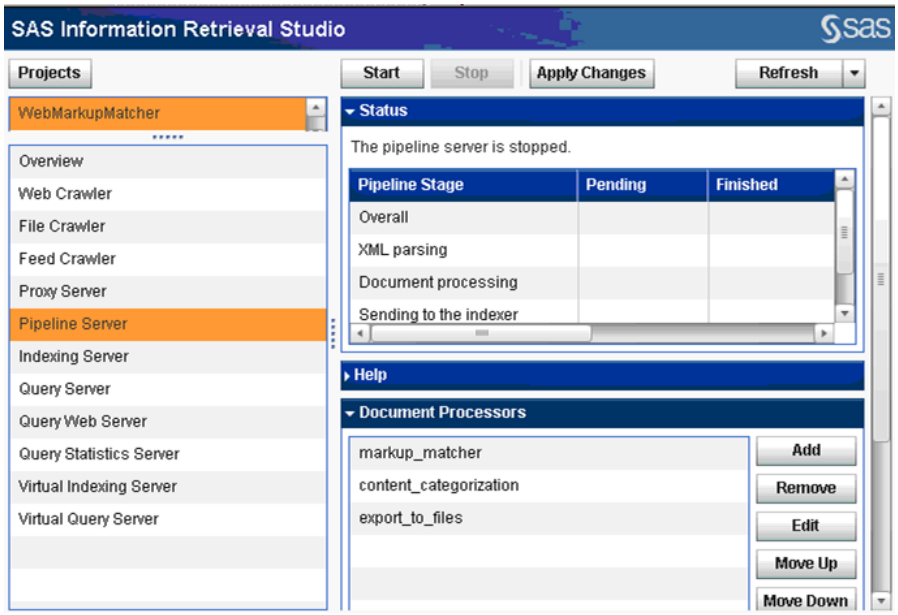
- Always apply your changes and stop and restart your servers after each change.
- Remove any unnecessary pipeline processors.
- Ensure that all of the required servers are up-to-date and running.
- Close the browser window every night and reopen the browser before working the next day.

- Start the Document Inspector operation before you start a crawler to see the fields and their content in one document as it passes through the pipeline server. (Click **Take Snapshot** in the Document Inspector pane of the Pipeline Server to start this operation.)
- Read this chapter and follow all of these directions before attempting your own configurations. Examples of configurations that require multiple components are available in Chapter 4: *Sample Configurations*.

Choose to configure only the gathering, normalizing, processing, indexing, and exporting operations that you require by using the following sections. After you configure the components that you require, you can start a crawler and run all of the configured processes simultaneously on their respective servers. In other words, be sure that all of the components that your operations require are configured and running before starting your crawler.

The examples in this chapter demonstrate the use of a default SAS Information Retrieval Studio project. You can develop multiple projects to handle the requirements of different configurations simultaneously. For this reason, it is unnecessary to install multiple copies of SAS Information Retrieval Studio.

Display 2-1: SAS Information Retrieval Studio Main User Interface



2.2 Configuring Gathering Using Crawlers

2.2.1 Overview of the Web and File Crawlers

Choose a crawler to locate and return the documents that contain the information that you seek. Crawlers gather documents from the Internet or your corporate fileshares.

This section explains how to configure the following crawlers:

- web crawler: enters the Internet from the specified entry point and returns documents.
- file crawler: crawls fileshares on your organization's network, or your local machine. These file types can include web pages that are stored in XML or HTML formats and desktop documentation that is stored in file types such as PDF and Microsoft Office. Use the appropriate document processor for each file type.

2.2.2 Configuring the Web Crawler

2.2.2.A Implementation

To configure the web crawler:

1. Access the Web Crawler pane.
2. Specify, or autodetect, an HTTP proxy server. The HTTP proxy server is different from the proxy server (not the pipeline proxy server). For more information, see your network administrator.
3. Specify one or more URLs as entry points for the web crawler.

   By default, the prefix of each entered URL is added to the scope of the crawl.
4. (Optional) You can edit the scope of the crawl.

2.2.2.B What You Need to Know

After the web crawler collects the maximum number of pages allowed, this crawler stops running. Restart the web crawler at any time.

2.2.3 Configuring the File Crawler

2.2.3.A Implementation

To configure the file crawler:

1. Access the File Crawler pane.
2. Specify the directory path that this crawler traverses.
3. (If you crawl HTML and XML documents and specify `markup_matcher` in the pipeline server) Go to **Configuration > General Settings**, select `Yes` for **Encapsulate XML Files**.

2.2.3.B What You Need to Know

After the file crawler collects all of the files in the traversed directories, the crawler stops running. You can restart the file crawler at any time.

Be sure to select `Yes` for **Encapsulate XML Files** if you are gathering HTML or XML files using the file crawler and are specifying `markup_matcher` as the text normalization tool. If you leave the default option `No` selected, `markup_matcher` is not applied in the pipeline server.

2.3 Configuring Normalizing Processors

2.3.1 Overview of Configuring Text Normalization Processors

Text normalization takes the input document and parses the text so that the text can be consumed by other operations. The text normalization tools are called from the pipeline server. Within the pipeline server, the input document is represented as a set of named fields that are paired with values (field-value pairs). For this reason, you can specify a logically ordered list of processors that perform tasks on the input fields beginning with text normalization.

The text normalization tools include `parse_html`, `heuristic_parse_html`, `markup_matcher`, and `document_converter`, among others. Before you configure the pipeline server, decide whether to call one of the following normalizers. If so, access the Global pane to ensure that the selected server is running before you try to add one of these text normalizers to the pipeline server.

- SAS Markup Matcher (for use with the `markup_matcher` document processor): upload your custom matcher for input documents that are in XML or HTML format. For more information about how to develop your custom matcher using SAS Markup Matcher, see Chapter 3: *Creating a Matcher Using SAS Markup Matcher*.
- Document Conversion (for use with the `document_converter` document processor): extract plain text content from PDF or Microsoft Office input documents without specifying any configuration for this normalizer.

2.3.2 Implementation Using the Pipeline Server

To configure text normalization using the pipeline server:

1. Access the Pipeline Server pane.
2. Specify a text normalizer using one of the examples in *Overview of Configuring Text Normalization Processors* above. Keep the normalizer first in the Document Processors pane.

2.3.3 What You Need to Know

Text normalization is the required first step in the pipeline server.

Select `Yes` for **Encapsulate XML Files** if you are gathering HTML or XML files using the file crawler and you are also specifying `markup_matcher` as the text normalization tool. If you leave the default option `No` selected, `markup_matcher` is not applied in the pipeline server.

2.4 Configuring Processing Components

2.4.1 Overview of Processing Operations

You can perform document processing on normalized text before it is output to another application or to the index. These processing operations add document tags, labels, or perform operations such as `match_and_copy`.

One example is tagging input field-value pairs with a category or a concept using the `content_categorization` processor.

Write category, concept, and LITI (contextual extraction) rules using SAS Content Categorization Studio. These projects run on the SAS Content Categorization Server.

You can use these tags for facetted search. Facetted search enables end users to intuitively navigate to the documents that match their input query terms using the labels that map to your tags.

2.4.2 Implementation Using the Pipeline Server

To configure document processing using the pipeline server:

1. Access the Pipeline Server pane.
2. List any of the document processors explained in Section 2.4.1 *Overview of Processing Operations* on page 21 below the text normalization processors.

2.4.3 What You Need to Know

In order to be applied, document processors are listed after text normalizers.

If you do not see the categories or concepts that you expect, check the configuration file for SAS Content Categorization Server. The configuration file references the `.desc` files in the `descriptors` directory. The `.desc` files, in turn, reference the `Models` folder for the uploaded SAS Content Categorization Server projects. Also ensure that SAS Content Categorization Server is running.

2.5 Configuring Indexing Operations

2.5.1 Overview of the Indexing Operations

Index your documents as field-value pairs after the text is normalized. If you choose to process your documents, processing operations are an intermediary between normalizing and indexing. Document processors such as `content_categorization` can enable you to add facetted search capabilities to your indexed documents when you specify the categories and concepts fields as labels. For more information about adding document processors, see Section 2.4 *Configuring Processing Components* on page 21.

The indexing server stores field-value pairs in the index, where they can be matched by input query strings. Different types of fields are used to match specific types of indexed information. Specify the following types of fields:

- **`Standard`**: used for keyword searching
- **`Info`**: used for display in the search results list
- **`Boolean`**: used for facetted search labels

If you create new documents and fields using SAS Markup Matcher, add these fields to the indexing server if you plan to index these documents.

You can delete your index, or add to the index, at any time. If you make any changes to the components that feed the index, apply the changes and restart the indexing server. If you change the configuration of the indexing fields, rebuild your index.

2.5.2 Implementation Using the Indexing Server

To configure the indexing server:

1. Access the Indexing Server pane.
2. Specify the field types in your index. See *Overview of the Indexing Operations* above.

Categories and concepts are automatically added as `Label` fields in the Indexing Server > General Settings pane. These labels enable end users to intuitively navigate to the documents that match their input query terms.

2.5.3 What You Need to Know

If your documents are not indexed, they cannot be searched for query operations.

The indexed documents can also be exported as files that can be used with other SAS applications.

If you choose to build an index, other operations can affect the build process. For example, see the following list of operations:

- Starting and stopping a crawler affects the flow of documents to the server. For example, if you stop the crawler and then restart it, the same set of documents might be added to the index twice.
- Using some of the document processing operations in the pipeline server automatically specifies the names of the fields passed to the indexing server.
- Changing the field names, types, and functionalities that you specify in the Configuration pane of the indexing server, affects the index. If you change the configuration of the indexing fields, rebuild your index.

The Delete Index operation removes the existing index. A new index can be built with the specified changes after you restart the crawler.

The Apply Changes operation deletes the existing index. The indexing server is restarted so that a new index can be built.

End users access the Query Interface to search the index.

2.6 Configuring Exporting Processors

2.6.1 Overview of Exporting Output Documents

Export documents, whether these documents are also indexed, in order to use the output with applications such as SAS Content Categorization Studio.

Specify the output types and the location for this output using the pipeline server. Export file types include `.xml`, `.csv`, and, `.txt` and can be sent to a directory or to another application.

2.6.2 Implementation Using the Pipeline Server

To add export processors to the pipeline server:

1. Access the Pipeline Server pane.
2. Specify the document processors. For example, you can specify `export_csv` (installation directory), `export_to_files` (installation directory `work/export-to-files`), or `export_to_sentiment_analysis_workbench` (export directly to SAS Sentiment Analysis Workbench). These processors are ordered to follow normalization and processing operations.

2.6.3 What You Need to Know

You can choose `export_csv` to perform a quick output check (after using the document inspector operation) on the content of the exported documents.

If you select **`CSV export`** in the Document Processor: content_categorization > Output pane, the application adds a column for each output field.

You can specify that multiple CSV files be output when you use New File Creation pane within the Document Processor: export_csv pane.

3
Creating a Matcher Using SAS Markup Matcher

- *SAS Markup Matcher Overview*
- *Overview of How to Create a Custom Matcher*
- *Creating a Document and Its Fields*
- *Using SAS Markup Matcher Extensions*
- *Adding Metadata*
- *Using Regular Expressions*
- *Developing and Applying a Template*
- *Publishing and Uploading Your Matcher*
- *What You Need to Know*

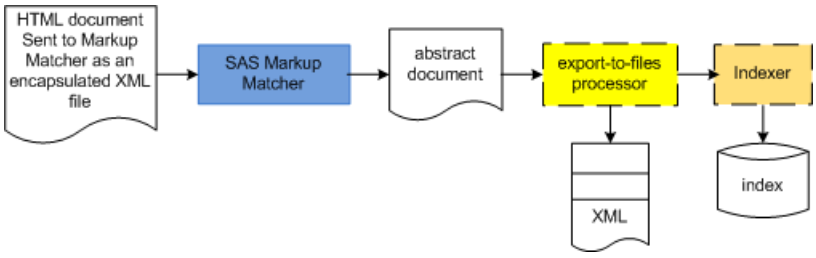
3.1 SAS Markup Matcher Overview

Use this chapter to specify a custom matcher that runs on the Markup Matcher Server. Matchers map the information in the abstract documents sent to them according to the specified document and field creation rules that you write in your matcher. You specify the rules that tell the matcher when to create a new document and what fields to add to that document. An abstract document is then output by SAS Markup Matcher. This output can either be indexed as field-value pairs or new files can be created in formats such as `.csv` and `.xml`.

If you choose to output `.csv` documents, you can use these files with SAS data sets or with Microsoft Excel. If you output to `.xml` documents, you can specify the fields that you want to use with applications such as SAS Content Categorization and SAS Sentiment Analysis. To index the new fields that you define in SAS Markup Matcher, add these fields to the indexing server.

You can also specify how a document is output for direct export to SAS Sentiment Analysis Workbench.

Figure 3.1 Document Processing with SAS Markup Matcherr



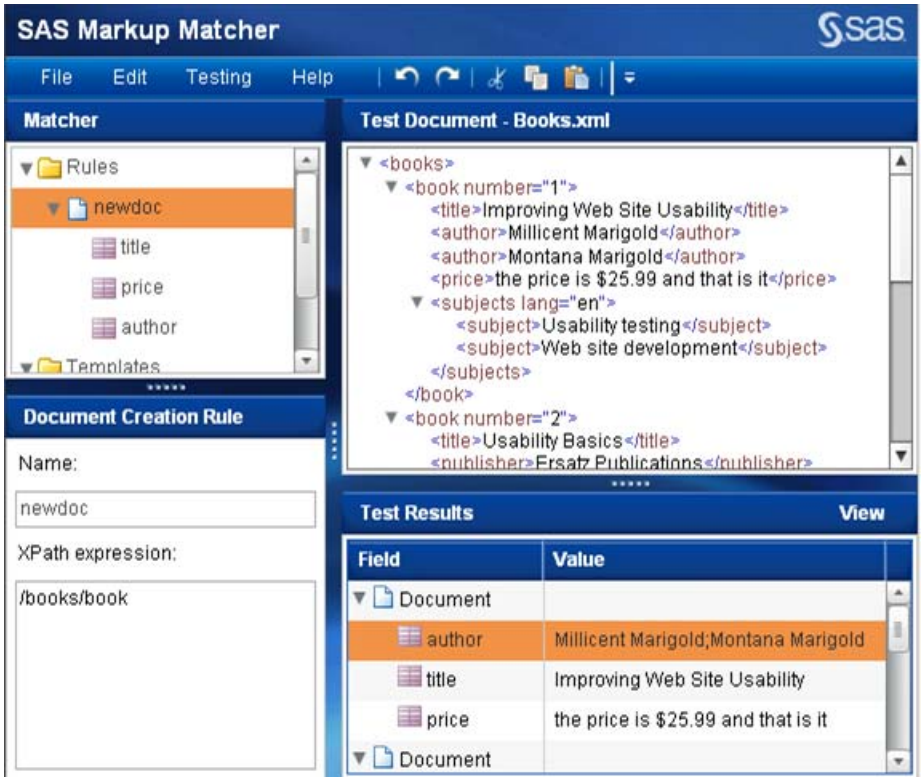
3.2 Overview of How to Create a Custom Matcher

To develop a rule-based matcher for your XML and HTML input files, review the following steps and examples:

1. Access Markup Matcher Server:

   Go to **Global > Markup Matcher Server > Status** and click the URL link (that is available when the service is running) to access the SAS Markup Matcher user interface. SAS also recommends that the HTTP proxy server is running for Markup Matcher Server. (The example shown below is frequently referenced in the rule-writing

examples that are found in this chapter.) To use a customized SAS Markup Matcher user interface layout, drag and drop the panes.



2. Access a testing document:

   Before you write your rules, access a web page or choose an `.xml` or an `.html` file in a folder on your machine. Write your rules to reconfigure or modify the data that is output for this document. For example, create documents for book reviews that omit some of the information found in the input web page such as the price, publisher, and overview. You can also choose to reorder some data or to change the case of the output.

3. Plan your output:

   The output documents in this example are `.xml` files. For other output options, see Section 3.1 *SAS Markup Matcher Overview* on page 25.

Before you use this step, review Section 2.3 *Configuring Normalizing Processors* on page 20. SAS Markup Matcher is applied to input documents by the pipeline server. You can choose to add additional processing steps before the documents are output. These processors might affect how you choose to configure the matcher that you define.

For example, apply categories, concepts, or contextual extraction (LITI) concepts using SAS Content Categorization Studio projects running on SAS Content Categorization Server.

If you add additional processing steps, consider the files that you plan to apply as you write your matcher. You could output a new document for a web page that lists a book for sale, information about that book, and its reviews. In this case, the title, author, price, overview, and reviews might be separate fields in each output document.

You might also want to consider the location of the matching text in the document and the rules in the input project. For example, the project settings that you specify in SAS Content Categorization Studio can affect matching in various ways that include case and matching location.

If you plan to use your documents with SAS Sentiment Analysis, you could specify a matcher that limits the fields in the output document. For example, map relevant information only to the title, author, and review tags. Using this example, choose to map all of a specific user's review sections on a web page to separate fields in the output document. Alternatively, map all of the review section tags for each reviewer to a single field.

Upload the matcher that you create to the Markup Matcher Server. You can also choose to create several matchers, specifying one for each set of document flows, or projects. (You can also choose to create more than one matcher to address different document structures.)

4. Write your rules:

   SAS Markup Matcher contains a user interface that lets you enter rules, or to point-and-click on a tag in the testing document (see Step a. on page 29), to create rules.

   Use the Add Document Creation Rule operation to create one or more output documents from a single input document. Use the Add Field Creation Rule operation to specify the fields in this output document.

If this data is available within the input document, the specified information is mapped to each field in the new, output document. This data can include the output document metadata as a field.

When you specify the SAS Markup Matcher metadata extension `mm:metadata`, you can perform several types of operations. For example, identify the output documents for each input document, search on the metadata, and apply a timestamp or a unique number to each output document.

Create a template to define a rule that can be applied multiple times. Call a template from another template or from a field creation rule.

Use any of these operations to develop a rule (the suggested order is shown below):

a. Point-and-click to create a rule. Select a rule and click on either an XML tag in the test document. For more information, see Section 3.3.2 *Field Creation Rule Example* on page 30.

b. Edit the point-and-click rules using SAS Markup Matcher extensions.

c. Add fields by using SAS Markup Matcher extensions alone. These functions begin with the `mm:` suffix. For information about SAS Markup Matcher-specific functions, see Section 3.4 *Using SAS Markup Matcher Extensions* on page 32.

d. When necessary, manually write rules using any of the syntax examples in this chapter.

3.3 Creating a Document and Its Fields

3.3.1 Document Creation Rule Example

`/books`: a new document is output for each instance of a match on the `books` tag found at the top level of tags in input documents.

The document creation rule specifies the current location. For example, if the field creation rule `./author` is specified, the period (`.`) specifies that the path is relative to the `/books` rule in the document creation rule.

3.3.2 Field Creation Rule Example

`book/author`: return text in the selected node relative to the node selected in the document creation field.

The text demarcated by the `author` tag that is nested directly below the `book` tag is returned. For example, see `Millicent Marigold`.

3.3.3 Absolute and Relative Path Examples

`/book/author`: specify an absolute path to bypass the document creation rule. In other words, jump to the top of the input document and return a match on the node specified.

`//author`: specify an absolute path to bypass the document creation rule. Skip over any intermediary nodes in the document to return the matched text in this node.

`./books/author`: specify a relative path, which does not bypass the node matched by the document creation rule to return the matched text for this node. This rule can also be written as: `books/author`

`.//author`: skip over any number of intermediary nodes such that the matched text is located within the current location (as defined by the document creation rule). This rule can also be written as: `/author`

3.3.4 Match a Specific Node Example

`subjects/subject[2]`: return text in a specific node, where more than one node has the same tag name. For example, return text in the second subject node relative to the document creation rule.

3.3.5 Return Text in div and span Tags

These examples are for `div` tags, make the appropriate changes for `span` tags.

`//div[@align="center"]:` returns a match on `Reviews` for:

```
<div align="center">Reviews</div>
```

`//div[mm:has-class(.,"blue")]`: the `class` attribute is the only attribute that can take several values. Use the (SAS Markup Matcher-specific function) `mm:has-class` to locate the text in a `div` tag such that the specified class appears. This statement is true regardless of any other classes that co-occur.

A match is returned on `Reviews` for:

```
<div class="blue small">Reviews</div>
```

`//div[@id="first"]`: use the *required* syntax in order to return the text in a `div` section with the specified `id` (`id`s are unique).

A match is returned on the text `Reviews` for the following `div`:

```
<div id="first">Reviews</div>
```

`//div[mm:has-class(.,"pr-review-text")]`: return text within one, specific `div` tag within a table where the document creation rule is:

```
/html/body/table
```

A match is returned on the text `Positive comments about....Burning Bush....I sent this shrub to my sister when she bought her new home. She loved it.` for the following input document snippet:

```
<!toecap html>
<html xmlns="http://www.w3.org/1999/xhtml"
<table width="100%">
<tr>
```

```
<td class="vd7gy">
<div style="float:left;" >
<a href="http://www.myshrubpageexample.com"
class="vd7gy" title="Shrubs-Home"> Home </a>>
</div></div> 
<h1 class="breadcrumb-name">My Shrub Place Page
Example</h1>>
<div class="pr-review-text">
<p class="pr-comments-header">Positive comments about
<em><span class="pr-product-name">Burning Bush</span>
</em>:</p>
<p class="pr-comments">I sent this shrub to my sister
when she bought her new home. She loved it.</p>
</div>
</td></tr>
</table>
```

3.4 Using SAS Markup Matcher Extensions

3.4.1 Overview

SAS Markup Matcher supports all of the standard constructs of XPath and adds the additional functionalities of SAS Markup Matcher specific extension functions. Access the document covering SAS Markup Matcher-specific functions using **Help > Contents**.

3.4.2 mm:string-join

Return a list from the specified nodes and delineate this list using a separator character such as a semicolon (`;`). Use the `concat` operation to precisely specify the returns including reshuffling tags, to add text, or to perform operations such as editing or replacing.

`mm:string-join(author, ";")`: return a string that joins all of the matched text in multiple instances of the same tag that are located within one section. Each returned instance is separated with a semicolon (`;`).

In this example, the `node set` in the `mm:string-join(node set, string)` extension is replaced by the node `author` that is selected in the point-and-click operation.

`mm:string-join(//div[@align = “left”],”;”)`: return and join all of the text in the matching `div` tags. The match: `Reviews; Titles` is returned for the following:

```
<div align=”left”>Reviews</div>
<div align=”left”>Titles</div>
```

3.4.3 Convert Case

The examples shown below are for uppercase. Modify these examples as required for `mm:lower-case`.

`mm:upper-case(//table[@id="productSummary"]/tbody/tr/td[3]/div/div)`: return the match represented only in uppercase text when you want to match text in a case-sensitive manner. You could specify the `mm:upper-case` operation to use the output documents with concepts in SAS Content Categorization. Consider using this operation when you apply the **Match Terms in All Uppercase** setting in the Project Settings > Concepts pane of SAS Content Categorization Studio.

In this example, `string` in the `mm:upper-case(string)` extension is replaced by the node `//table[@id="productSummary"]/tbody/tr/td[3]/div/div` that is selected in the point-and-click operation. All of the mapped text is returned and appears only in uppercase letters.

3.5 Adding Metadata

Metadata adds information from outside of the input document to the output document.

`mm:uuid()`: add a unique identification number to each output document, which is the most common usage case for metadata

`mm:format-date(“yyyy-MM-dd HH:mm:ss”)`: add the current timestamp to each output document

`concat(mm:metadata(“id”), “#”, mm:document-instance())`: specify a unique name for each document that is created whether several documents are created from a single, or multiple, input documents.

In this example, the first output document name could be `books2.xml#0` if the `id` is `books2.xml`. The appropriate crawler, in this example, passes the `id` with the input document to the document processor and the `mm:metadata` function identifies this `id` as a string. (Use **Testing > Metadata** to test metadata by entering sample values into the Metadata <$startrange>metadata examples for a matcher.)

Use the `concat` operation to add the `#` (which references parts of a document in HTML). The `mm:document-instance` function defines the number assigned to the output document. For each input document, the number assigned to each instance of a document, as defined by the document creation rule, begins with 0. (When you specify this operation, you can also refer back to the original document from each output document.) The metadata can be used for operations such as search.

3.6 Using Regular Expressions

Use regular expressions found in the Java library in a field creation rule.

`mm:replace(book/title/text(), “i”, “y”)`: substitute one character with another character. In this example, the character `i` is replaced by the character `y`.

`mm:replace(book/title/text(), "S[a-z]*", "Overall")`: substitute one string with another string. In this example, the word `Site` is replaced by the term `Overall`. (Any term that begins with an uppercase *S* within the specified location would be replaced by the term *Overall*.)

`mm:replace(book/title/text(), "Site *", "")`: replace any number of occurrences of the preceding character when you specify the asterisk (`*`). In this example, the term Site is returned. Any amount of whitespace characters that follow this term are eliminated when the matched string is returned.

`//div[mm:matches(@id, “heading[1-5]$”)]/text()`: matches all heading tags between 1 and 5 that also meet the rule specifications. For this reason, a match is returned for the first line below, but not for the second line:

```
<div id=”heading5”>good</div>
<div id=”heading15”>bad</div>
```

3.7 Developing and Applying a Template

Develop a template rule within a template node and apply that template using a field creation rule.

`mm:string-join(author, “,”)`: specify this text in the `author-template` template to return all of the authors for each book in the input document. Separate these names using commas (`,`) while delimiting the authors for each book using semicolons (`;`).

In the `auth` field creation rule, write:

```
mm:apply-template(//book, “author-template”, “;”)
```

`concat(title, “by”, author[1])`: specify this text in the `title-template` template to return a matched order that differs from the order in the input document. (Use a `concat` operation because the matching order cannot be reshuffled using an `mm:string-join` operation.)

Alternatively, write this rule (When used in the context of a template, the period (`.`) references the current node and returns the next member of the list.):

```
concat(.//title, “by”, .//author[1])
```

In the `books` field creation rule, write:

```
mm:apply-template(//book, "title-template", ";")
```

`concat(@number, “:”, title, “by”, mm:string-join(author, “and”))`: use the `concat` operation with the `mm:string-join` operation within the `number-template` template rule. This example also returns the book number followed by a colon (`:`), the book title, and the word `by`.

In the `books` field creation rule, write:

```
mm:apply-template(book, “number-template”, “;”)
concat(@number, ": ", title, " by ", mm:apply-
template(author, "author-template", " and "))
```

: apply a template (`author-template`) within a template (`book-template2`) in order to return a longer list specifying multiple fields with strings. (This example references the `author-template` template shown above and expands the preceding `mm:string-join` example.)

In the `bookstwo` field creation rule, write:

```
mm:apply-template(book, "book-template2", ";")
```

3.8 Publishing and Uploading Your Matcher

Upload the matcher without specifying an `.xml` extension. Apply your changes and restart Markup Matcher Server. SAS also recommends that the HTTP proxy server is running for Markup Matcher Server.

3.9 What You Need to Know

Remember the name of the uploaded matcher that you want to use. You manually enter this name (without an `.xml` extension) when you add `markup_matcher` to the pipeline server.

Download a matcher to make changes or to add rules to the matcher.

Unless you specify absolute paths, field creation rules are based on the document creation rule.

Use the *required* rule syntax in order to return the text in a `div` section with an `id`.

Specify `concat` to precisely specify the returns including reshuffling tags, to add text, or to perform operations such as editing or replacing.with rules. Specify `mm:string-join` to return a list from the specified nodes. You can delineate this list using a separator character such as a semicolon (`;`).

Use `mm:metadata` to specify a unique name for each document that is created whether several documents are created from a single, or multiple, input documents. When you use this operation, you can also reference the original, input document.

Templates are applied by field creation rules.

When writing rules, follow these conventions:

- .: relative to the current location
- /: go back to the top of the document (jump over the document creation rule)
- //: skip over any number of intermediary nodes

Make sure that the Markup Matcher Server is running whenever you specify a `markup_matcher` in the pipeline server.

4
Sample Configurations

- *Overview of Sample Configurations*
- *Indexing Fileshares*
- *Adding Categories and Exporting Matcher Output*
- *Indexing and Exporting Web Documents*
- *Exporting to SAS Sentiment Analysis Workbench*

4.1 Overview of Sample Configurations

This chapter references the single-operation configurations in Chapter 2 and provides multi-configuration examples that address specific tasks. These tasks do not preclude you from adding additional operations to your configuration even when the operations are omitted from the examples.

If you plan to apply a custom matcher as a text normalizer, configure this matcher before using this chapter. For more information, see Chapter 3: *Creating a Matcher Using SAS Markup Matcher*.

When you specify any of the sample configurations in this chapter, make sure that you follow all of the best practices explained in both Chapter 2 and below:

- Follow this ordering when configuring the pipeline server:
  a. Normalize input texts.
  b. (Optional) Process input documents.
  c. (Optional) Export files.
- Check your results at every stage:
  a. (If you develop a matcher) Check your custom matcher using testing documents within SAS Markup Matcher.

b. Use the Document Inspector operation to check the output for each processor as you configure the pipeline server.

c. (If you build an index) Use the query interface to search your index for expected matches.

d. (If you export files) Check a small number of the exported files before exporting a larger number.

If you choose to export your output to files, such as XML, consider using the `export_csv` operation first. A single CSV file can return the field-value data from many files in a format such as XML. Spot check your exported files later if you choose this operation. For more information, see Section 4.3.1 *Discussion* on page 42.

4.2 Indexing Fileshares

4.2.1 Discussion

For this configuration, use the file crawler to index documents that are located in a directory. The documents in this example have XML, HTML, PDF, and Microsoft Office file extensions. `markup_matcher` is listed above `document_converter` to prevent `document_converter` from processing the HTML and XML documents.

4.2.2 Implementation

To index the documents passed to SAS Information Retrieval Studio by the file crawler:

1. Confirm that both Markup Matcher Server and Document Conversion Server are running.
2. Configure the file crawler using Section 2.2.3 *Configuring the File Crawler* on page 19.

3. Configure the pipeline server to normalize the input text using the following processors in the specified order. For more information, see Section 2.3 *Configuring Normalizing Processors* on page 20.

   Select `markup_matcher` and enter the name of the uploaded matcher running on the Markup Matcher Server. For more information, see Chapter 3: *Creating a Matcher Using SAS Markup Matcher*.

   Select `document_converter`.

4. Configure the indexing server using Section 2.5 *Configuring Indexing Operations* on page 22.

4.2.3 What You Need to Know

To ensure proper resolution whenever end users view search results, use an absolute, instead of a relative, path. For Windows fileshares, use UNC names instead of local paths. For example, specify:

```
\\hostname\share
```

The ordering of normalization documents processors is significant.

No other processors are necessary in the pipeline server to enable indexing.

4.3 Adding Categories and Exporting Matcher Output

4.3.1 Discussion

Use the file crawler to gather XML and HTML documents from a repository. Choose to export your custom matcher output as a single CSV file that displays the matching categories for each row, or file. For more information about CSV files, see Section 4.4.1 *Discussion* on page 43.

Use this example in ways that are similar to Section 4.5 *Exporting to SAS Sentiment Analysis Workbench* on page 44. This example enables you to see all of the data that appears in one, or more, spreadsheets instead of separate files.

4.3.2 Implementation

To export the custom matcher output and category matches to one CSV file, use the steps in Section 4.4.2 *Implementation* on page 43 while making the following changes:

1. Confirm only that the Markup Matcher Server is running.
2. Do not select `document_converter`.
3. Select `content_categorization`. Specify only categories and CSV export. For more information, see Section 2.6.2 *Implementation Using the Pipeline Server* on page 24.
4. Edit the `export_csv` processor settings.
5. Omit Step 4. on page 41.

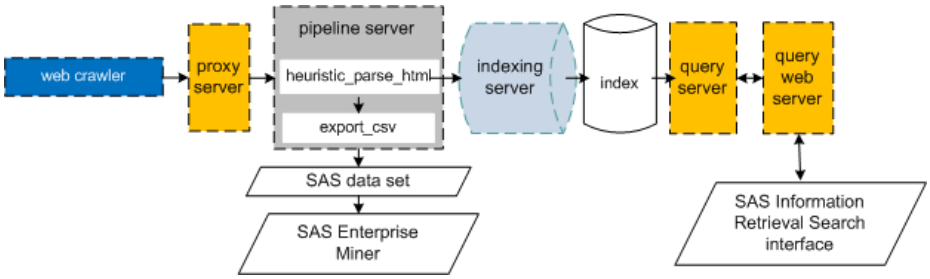
4.4 Indexing and Exporting Web Documents

4.4.1 Discussion

Use the web crawler to gather documents to add to the index. When you add facetted search capabilities, you enable end users to intuitively search your index using the categories or concepts that you specify as labels.

In addition to building an index, you can export a single CSV file that can be converted into SAS data sets or accessed using Microsoft Excel. You can use Microsoft Excel to see the information that might otherwise be tagged to multiple XML files in spreadsheet format. If you convert the CSV file into a SAS data set, this output can be consumed by SAS applications such as SAS Enterprise Miner.

Display 4-1 Indexing and Exporting from Input Web Files



4.4.2 Implementation

To index and export a CSV file using documents gathered by the web crawler:

1. Configure the web crawler to gather documents using Section 2.2.2 *Configuring the Web Crawler* on page 18.
2. Add the following normalizing, processing, and exporting tools to the pipeline server:
   - Select `heuristic_parse_html` in order to extract text from HTML documents. For more information, see Section 2.3 *Configuring Normalizing Processors* on page 20.

- Specify the categories and concepts that you map to labels to enable facetted search and to output in your CSV file. Modify Step 3. on page 42 for CSV output for your categories and concepts. See Step 3. below for more information about facetted labels.
- Use Step 4. on page 42.

3. Configure the indexing server using Section 2.5 *Configuring Indexing Operations* on page 22.

4.5 Exporting to SAS Sentiment Analysis Workbench

4.5.1 Discussion

Use the file crawler to gather XML and HTML documents from a repository. Normalize the text in these documents using a matcher that you develop to remap information and to eliminate unnecessary data. Export the newly structured XML files directly to other SAS applications such as SAS Sentiment Analysis Workbench.

4.5.2 Implementation

To export the SAS Markup Matcher output to SAS Sentiment Analysis Workbench:

1. Confirm that the Markup Matcher Server is running.
2. Configure the file crawler using Section 2.2.3 *Configuring the File Crawler* on page 19.
3. Configure the pipeline server:
   - Select `markup_matcher` and enter the name of an uploaded matcher that is running on the Markup Matcher Server. For more information, see Section 2.3 *Configuring Normalizing Processors* on page 20. Also see Chapter 3: *Creating a Matcher Using SAS Markup Matcher*.

- Specify `export_to_sas_sentiment_analysis_workbench` as the export operation. Add only the names of the fields defined in the specified matcher. For more information, see Section 2.6.2 *Implementation Using the Pipeline Server* on page 24.

Appendixes

- Appendix A: *Recommended Reading*
- Appendix B: *Glossary*

Appendix: A
Recommended Reading

The following books are recommended as companion guides:

- *SAS Sentiment Analysis Studio: User's Guide*: Create a SAS Sentiment Analysis Studio project, test, and upload the rules to SAS Sentiment Analysis Server.
- *SAS Sentiment Analysis Server: Administrator's Guide*: Automate the process of applying the rules that you define in SAS Sentiment Analysis Studio to your input documents.
- *SAS Sentiment Analysis Workbench: Installation Guide*: Install SAS Sentiment Analysis Workbench and the prerequisite software.
- *SAS Sentiment Analysis Workbench: Administrator's Guide*: Set up SAS Sentiment Analysis Studio projects, add users, and specify the files to be used. These files include SAS Sentiment Analysis Studio and SAS Content Categorization files.
- *SAS Sentiment Analysis Workbench: User's Guide*: Review and edit the automated analyses and create reports illustrated with graphs.
- *SAS Enterprise Content Categorization: User's Guide*: Create a SAS Content Categorization project, test, and upload the rules to SAS Content Categorization Server.
- *SAS Content Categorization: Installation Guide*: Install SAS Content Categorization.
- *SAS Enterprise Content Categorization: Administrator's Guide*: Understand how to use the collaborative features with SAS Content Categorization Server. Also learn how to install this server.

SAS offers instructor-led training and self-paced e-learning courses to help you get started with the SAS add-in, learn how the SAS add-in works with the other products in the SAS Enterprise Intelligence Platform, and learn how to run stored processes in the SAS add-in.
For more information about the courses available, see `support.sas.com/training`.

For a complete list of SAS publications, see the current SAS Publishing Catalog. To order the most current publications or to receive a free copy of the catalog, contact a SAS representative at

SAS Publishing Sales
SAS Campus Drive
Cary, NC 27513
Telephone: (800) 727-3228*
Fax: (919) 677-8166
E-mail: sasbook@sas.com
Web address:support.sas.com/pubs
* For other SAS Institute business, call (919) 677-8000.

Customers outside the United States should contact their local SAS office.

Appendix: B
Glossary

categorization

concisely defines the subject matter of a document, in other words, the main idea or subject of the document.

concept

specifies any of the following—a string, token, or an argument—to locate in an input document.

contextual extraction

specifies complex definitions, which might be comprised of multiple rules, for concepts. See *LITI*.

crawl

an entire run of a crawler, instead of a single page download.

definition

defines a concept. There can be many rules for each concept definition. This term is also used interchangeably with *rule*. See *rule*.

document

is a unit of textual data. For example, a document can be an HTML page, a Microsoft Word file, a PDF file, or one row in a CSV file or a database. A document can also be an article or summary in a feed.

In SAS Information Retrieval Studio, each document is represented as a configurable set of fields. Each file has a name and a value. Unnecessary fields can be either left empty or omitted from the document.

facetted search

applies identifying labels to matched documents. These labels enable your end users to intuitively navigate to the documents that match their input query terms.

field-value pair

within the pipeline server, the input document is represented as a set of named fields that are paired with values.

label

specifies the value of the field that is passed to the query web server for each match that appears within the search window. Also see *caption*.

LITI

specifies complex definitions, which might be comprised of multiple rules, for concepts. See *contextual extraction*.

metadata

identifies information about information.

rule

defines an output document, field within that document or a template. Unless you use SAS Contextual Extraction Studio, only one rule defines each category. This term is also used interchangeably with *definition*. See *definition*.

string

refers to a group of words or characters.

Index

A

B

C

D

E

F

H

I

M

N

O

P

Q

R

S

T

U

X