Configuration Files

On this page

Aspire uses an xml file oriented schema. Every time a content source is created from the Content Management UI, a set of files are created; each of which contains a particular piece of configuration for the content source.

Content Source Configuration

The configuration of a content source consists of four xml files that are located in the configuration folder. These files are:

- general.xml
- connector.xml
- content-source.xml
- workflow.xml

```
${aspire.home}/config/content-sources
```

For every new content source configured, a new folder is created in the location specified above. The name of the content source folder will be a *norm alized* version of the content source name specified from the UI.

The content source folder name normalization follows this set of rules:

1. Remove whitespaces
2. If the folder name already exists, add a consecutive number id at the end of the name.

Example

- **Name**: test folder --> **Normalized Name**: testfolder
- **Name**: testfolder --> **Normalized Name**: testfolder1
  - **Assumption**: testfolder already exists.

To correctly load a content source, all four files are required on its configuration folder, otherwise the content source will not be loaded into the UI.

General Configuration File

The *general.xml* file contains the general information associated with the content source: **displayName**, **normalizedName** (used to uniquely identify the content source), **schedule** and the current **state** (*active* or *inactive*) of the content source.

```
<contentSource active="true">
  <schedule type="manually"/>
  <displayName>cifsTest</displayName>
  <normalizedName>cifsTest</normalizedName>
</contentSource>
```

Connector Configuration File

The *connector.xml* file contains the connector application definition and property values to install the connector into Aspire.

```
<application config="com.searchtechnologies.aspire:app-cifs-connector">
  <properties>
    <property name="generalConfiguration">true</property>
    <property name="snapshotDir">${dist.data.dir}/${app.name}/snapshots</property>
    <property name="ExtractTextMaxSize">unlimited</property>
    <property name="disableTextExtract">false</property>
    <property name="workflowReloadPeriod">15s</property>
    <property name="workflowErrorTolerant">false</property>
    <property name="debug">true</property>
  </properties>
</application>
```

Content Source Configuration File

The *content-source.xml* file contains the content source specific configuration to connect and perform a crawl against a given repository. Information like the repository url, the user credentials and any other repository configuration required by Aspire to crawl said repository.

```
<connectorSource>
  <url>smb://servername/AspireTesting/</url>
  <partialScan>true</partialScan>
  <subDirUrl>LSA</subDirUrl>
  <domain>search</domain>
  <username>jdoe</username>
  <password>pass1234</password>
  <indexContainers>true</indexContainers>
  <scanRecursively>true</scanRecursively>
  <fileNamePatterns>
    <include pattern=".*LSA.*"/>
    <exclude pattern=".*tmp.*"/>
  </fileNamePatterns>
</connectorSource>
```

Workflow Configuration File

The *workflow.xml* file contains the workflow configuration of all workflow trees configured for the content source: *afterScan*, *onPublish*, *onAddUpdate*, *onDelete* and/or *onError*. This file contains the configuration of all rules set on each of these workflow trees. For more details see Workflow.

```
<?xml version="1.0" encoding="UTF-8"?>
<workflow version="2">
  <templates>
    <template id="Boolean (Byte array)" type="choice">
      <description>Boolean test (from a byte array)</description>
      <ruleDescription>Tests if the value of '${field}' is '${value}'</ruleDescription>
      <dxf><dxf:template version="1.0" xmlns:dxf="http://www.searchtechnologies.com/DXF/2011"
><properties><field display="Field name" type="string"><dxf:help>The field from the document to test</dxf:
help></field><value display="Value" type="string"><dxf:help>The value to test for</dxf:help></value><
/properties></dxf:template></dxf>
      <script>

        f = doc.${field}?.getContent()
        if (f instanceof byte[])
          f = new String(f)
        f == "${value}"</script>
    </template>
    <template id="JobTerminate" type="script">
      <description>Terminates the job</description>
      <ruleDescription>Terminates the job</ruleDescription>
      <script>job.terminate()</script>
    </template>
    <template id="RaiseException" type="script">
      <description>Raises an exception</description>
      <ruleDescription>Exception: "${msg}"</ruleDescription>
      <dxf><dxf:template version="1.0" xmlns:dxf="http://www.searchtechnologies.com/DXF/2011"
><properties><msg display="Message" type="string"><dxf:help>The exception message</dxf:help></msg><
/properties></dxf:template></dxf>
```

```
      <script>

        import com.searchtechnologies.aspire.services.AspireException
        throw new AspireException("WorkflowException", "${msg}")
      </script>
    </template>
    <template id="SetStringValue" type="script">
      <description>Assigns a string value to a field</description>
      <ruleDescription>Assigns the value "${value}" to the field ${field}</ruleDescription>
      <dxf><dxf:template version="1.0" xmlns:dxf="http://www.searchtechnologies.com/DXF/2011"
><properties><field display="Field name" type="string"><dxf:help>The field from the document to modify</dxf:
help></field><value display="Value" type="string"><dxf:help>The string value to set</dxf:help></value><
/properties></dxf:template></dxf>
      <script>doc.${field} = "${value}"</script>
    </template>
    <template id="Switch (Byte array)" type="choice">
      <description>Switch (from a byte array)</description>
      <ruleDescription>Switches based on the value of '${field}'</ruleDescription>
      <dxf><dxf:template version="1.0" xmlns:dxf="http://www.searchtechnologies.com/DXF/2011"
><properties><field display="Field name" type="string"><dxf:help>The field from the document to test</dxf:
help></field></properties></dxf:template></dxf>
      <script>

        f = doc.${field}?.getContent()
        if (f instanceof byte[])
          f = new String(f)
        f</script>
    </template>
  </templates>
  <rules>
    <rule appName="/PostToCIFS" config="com.searchtechnologies.aspire:app-publish-to-file" id="1" type="
application">
      <description>PublishToFile</description>
      <properties>
        <property name="logFile">log/${app.name}/publishToFile.jobs</property>
        <property name="debug">false</property>
        <property name="numJobs">5</property>
      </properties>
    </rule>
  </rules>
  <plans>
    <plan name="onPublish">
      <reference rid="1"/>
    </plan>
  </plans>
  <applications>
    <application config="com.searchtechnologies.aspire:app-publish-to-file" name="/PostToCIFS">
      <properties>
        <property name="logFile">log/${app.name}/publishToFile.jobs</property>
        <property name="debug">false</property>
        <property name="numJobs">5</property>
      </properties>
    </application>
  </applications>
</workflow>
```

Add Content Source from the File System

Aspire allows users to create and add content sources directly from the configuration files without the need of saving the content sources through the UI and without restarting Aspire. This is possible due to the synchronization functionality introduced with ZooKeeper for failover. See Failover for Aspire using Zookeeper for more information.

To add a content source configuration:

1. Create a new folder, outside of ${aspire.home}/config/content-sources, with the **normalizedName** of the content source.
2. Create the *general.xml* as described above. Make sure the **normalizedName** element has the same value as the folder name.
3. Create the *connector.xml* and *content-source.xml* as described above. For more information on different content sources available, visit the Developer Notes for each Connector.
4. Create the *workflow.xml* as described above. For more information, visit Workflow.

5. Copy the folder **normalizedName** into ${aspire.home}/config/content-sources
6. Refresh the Aspire Management page, if there are no errors, you will see the new content source.