Source Writing Transformer (2.1 legacy document)

Table of contents

Table of contents

Warning:
This document was copied as is from the Cocoon 2.1 documentation, but has not yet been fully reviewed or moved to its new home.

1. Source Writing Transformer

Diverts xml from a pipeline, writing it to a Source (or deleting it).

Thankfully, FileSource is no longer the only Source that currently implements WritableSource; there are implementations of WebDAV and Apache Slide WritableSources in the scratchpad. Hopefully further ModifiableSource implementations (XMLDB, CVS, Email, SQL, etc.) will be appear in the future.

See the transformer in action with the Cocoon Samples for webdav block, and Wiki about it at WebDAVCMS.

- Name : write-source
- Class: org.apache.cocoon.transformation.SourceWritingTransformer
- Cacheable: no.

2. The Tags

<source:write> [<source:path/>] <source:source/> <source:fragment/> </source:write> <source:insert/> <source:path/> <source:source/> <source:fragment/> [<source:replace/>] [<source:reinsert/>] </source:insert> <source:delete/> <source:source/> [<source:path/>] - Ignored [<source:fragment/>] - Ignored [<source:replace/>] - Ignored [<source:reinsert/>] - Ignored </source:insert>

In the namespace xmlns:source="http://apache.org/cocoon/source/1.0".

The contents of the <source:fragment/> tag are written to the specified ModifiableSource when the document containing it is transformed by SourceWritingTransformer (or deleted if you are using the delete instruction).

3. Definition

<map:transformer name="write-source" src="org.apache.cocoon.transformation.SourceWritingTransformer"> <map:parameter name="serializer" value="xml"/> </map:transformer/>

The SourceWritingTransformer is predefined for you in the main SiteMap.

4. Invocation

This invokes the SourceWritingTransformer on your pipeline.

<map:transform type="write-source"/>

Or you can over-ride the default serializer here.

<map:transform type="write-source"> <map:parameter name="serializer" value="my-special-serializer"/> </map:transform>

5. The Tags in detail

5.1. source:write

The source:write tag can take optional attributes, create (defaults to 'true') and serializer (defaults to the serializer set up in the definition or invocation of the transformer).

Replaces the entire content of a Source (specified by the <source:source/> tag) with the contents of the <source:fragment/> tag, if @create is 'true', a new asset will be created if one does not already exist.

The <source:source/> and <source:fragment/> tags are required, a <source:path/> tag is optional, if specified, the value is used as an XPath to generate xml in your Source, in which to wrap your content.

5.1.1. source:source

The System ID of the Source to be written to.

e.g. <source:source>docs/blah.xml</source:source> or
<source:source>context:/blah.xml</source:source> etc.

5.1.2. source:fragment

The XML Fragment to be written.

For example:

<source:fragment><foo> <bar id="dogcow"/> </foo></source:fragment>

or

<source:fragment> <foo/> <bar> <dogcow/> <bar/> </source:fragment>

etc.

The second example type, can only be used when the <source:path/> tag has been specified.

5.1.3. source:path

[Optional] XPath to specify how your content is wrapped

e.g. <source:path>doc</source:path> - your content is placed inside a <doc/> root tag.

If this parameter is omitted, your content MUST have only ONE top-level node.

5.2. source:insert

The source:insert tag can take optional attributes, create (defaults to 'true') and serializer (defaults to the serializer set up in the definition or invocation of the transformer).

Inserts into a Source (specified by the <source:source/> tag) the contents of the tag <source:fragment/> at the XPath location specified in the <source:path/> tag, if @create is 'true', a new Source will be created if one does not already exist.

The <source:source/>, <source:path/> and <source:fragment/> tags are all required, the <source:replace/> and <source:reinsert/> tags are optional.

5.2.1. source:source

The System ID of the Source to be inserted into.

e.g. <source:source>docs/blah.xml</source:source> or
<source:source>context:/blah.xml</source:source> etc.

5.2.2. source:fragment

The XML Fragment to be written.

e.g.

<source:fragment> <foo> <bar id="dogcow"/> </foo> </source:fragment>

or

<source:fragment> <foo/> <bar> <dogcow/> <bar/> </source:fragment>

etc.

5.2.3. source:path

5.2.4. source:replace

[Optional] XPath (from <source:path/>) to select the node that is replaced by your new content

e.g. <source:replace>foo/bar/dogcow/@status='cut'</source:replace> (is equivalent to this in XSLT: select="foo[bar/dogcow/@status='cut']"), what gets replaced is the <foo/> which has a <bar/> with a <dogcow status="cut"/> in it.

The overwrite attribute of the parent <source:insert/> is used to check if replacing is allowed. If overwrite is 'true' (the default) the node is replaced. If overwrite is 'false' the node is only inserted if the replace node is found.

5.2.5. source:reinsert

[Optional] The XPath (relative to <source:replace/>) to backup the contents of the overwritten node to.

e.g. <source:reinsert>foo/versions</source:reinsert> or <source:reinsert>/doc/versions/foo</source:reinsert>.

If specified and a node is replaced, all children of this replaced node will be reinserted at the given path.

5.3. Notes

- if 'replace' is not specified, your 'fragment' is appended as a child of 'path'.
- if 'replace' is specified and it exists and 'overwrite' is true, your 'fragment' is inserted in 'path', before 'replace' and then 'replace' is deleted.
- if 'replace' is specified and it exists and 'overwrite' is false, no action occurs.
- if 'replace' is specified and it does not exist and 'overwrite' is true, your 'fragment' is appended as a child of 'path'.
- if 'replace' is specified and it does not exist and 'overwrite' is false, your 'fragment' is appended as a child of 'path'.
- if 'reinsert' is specified and it does not exist, no action occurs.

5.4. source:delete

This instruction takes only a <source:source/> parameter (as a child tag) and deletes the corresponding source. All other source:* tags are ignored, if present: this allows for easy source-write instructions to be generated on the fly (e.g. by a stylesheet). Just change source:write to source:delete and you're set.

6. Examples

6.1. Simple Write

<page> ... <source:write xmlns:source="http://apache.org/cocoon/source/1.0"> <source:source>context://doc/editable/my.xml</source:source> <source:fragment><page> <title>Hello World</title> <content> This is my first paragraph. </content> </page></source:fragment> </source:write> ... </page>

6.2. Insert at end

<page> ... <source:insert xmlns:source="http://apache.org/cocoon/source/1.0"> <source:source>context://doc/editable/my.xml</source:source> <source:path>page/content</source:path> <source:fragment> This paragraph gets <emp>inserted</emp>. With this one, at the end of the content. </source:fragment> </source:insert> ... </page>

6.3. Replace

<page> ... <source:insert xmlns:source="http://apache.org/cocoon/source/1.0"> <source:source>context://doc/editable/my.xml"</source:source> <source:path>page/content</source:path> <source:replace>p[1]</source:replace> <source:fragment> This paragraph <emp>replaces</emp> the first paragraph. </source:fragment> </source:insert> ... </page>

6.4. Insert at the beginning

<page> ... <source:insert> <source:source>context://doc/editable/my.xml</source:source> <source:path>page</source:path> <source:replace>content</source:replace> <source:reinsert>content</source:reinsert> <source:fragment> <content> This new paragraph gets inserted <emp>before</emp> the other ones. </content> </source:fragment> <source:insert> ... </page>

This sample does not currently work, see the tests in the scratchpad at http://localhost:8080/cocoon/mount/editor/tests.

You must have built Cocoon with the scratchpad included for this link to work.

6.5. Delete a source

<page> ... <source:delete> <source:source>context://doc/editable/my.xml</source:source> <source:delete> ... </page>

6.6. Sample of the output of these tags

This is the kind of information that the SourceWritingTransformer outputs to the pipeline, replacing the original source:write and source:insert tags

<page> ... <sourceResult> <action>new|overwritten|none</action> <behaviour>write|insert<behaviour> <execution>success|failure</execution> <serializer>xml</serializer> <source>source:specific/path/to/context/doc/editable/my.xml</source> <message>a message about what happened</message> </sourceResult> ... </page>

7. Known Problems

I cannot get the 'insert before' example working, which uses the <source:reinsert/> tag.

8. Warning

It is not known how robust this transformer is under even moderate load, especially when it comes to more than one person modifying the same file at the same time.

1. Comments

add your comments