NLM/NCBI Journal Publishing 3.0 Preview Stylesheets: Technical documentation

Journal Information
Journal ID: N/A
ISSN: N/A

Article/Issue Information
Publication date: 2009

NLM/NCBI Journal Publishing 3.0 Preview Stylesheets: Technical documentation

Mulberry Technologies, Inc.

This document provides technical information regarding the HTML and PDF Preview stylesheets for the NLM/NCBI Journal Publishing 3.0 tag set. (See http://dtd.nlm.nih.gov/publishing/.) It is intended for technical users, that is, personnel responsible for installing, configuring, and maintaining or extending these stylesheets. For information intended for general users regarding the scope and application of these stylesheets, see the User's Guide in this distribution. For instructions on how to run these stylesheets on Journal Publishing content using a basic configuration, see the Quick Start Guide.

General overview

The various tasks performed by stylesheets in this distribution have been distinguished so they may be mixed and matched to meet specific local requirements, including:

- Generating either HTML, PDF or XHTML output.
- Autoformatting of structured citation elements (`element-citation`) according to different sets of guidelines (we include stylesheets supporting the NLM/Pubmed citation style and an APA-like citation style).
- Filtering content based on the values of `specific-use` attributes assigned to elements in the source data.

These various operations are performed in stages as follows:

Preprocesses

Preprocesses are all designed to perform operations needed by some processes but not others, so they can be excluded when not needed. Generally speaking it is possible to run more than one preprocess, as long as they do not directly conflict with one another. When they do, it can be expected that the first process in the chain to handle a particular element or structure in the input will be the one to take effect.

All preprocessing stylesheets accept arbitrary (valid) Journal Publishing 3.0 XML and emit the same format: they are "modified identity transformations", whose operations are limited to a particular subset of elements related to their functional requirements.

Because some preprocesses perform operations that are not practically achieved in modifiable, maintainable XSLT 1.0, they use XSLT 2.0 and require an XSLT 2.0 processor. Others will work in either an XSLT 1.0 or 2.0 processor.

Brief descriptions of each preprocess appear below. Additionally, all stylesheets in this package are documented internally.

Main preview processes

There are two preview stylesheets in this distribution. `jpub3-html.xsl` formats Journal Publishing 3.0 in HTML for display in web browsers.[1] `jpub3-xslfo.xsl` converts Journal Publishing 3.0 XML into XSL

[1] Unfortunately, due to tradeoffs forced by discrepancies in their modeling of block vs inline elements in the Journal Publishing tag set and HTML, not all results of the HTML preview stylesheet can be guaranteed to be valid HTML. But none of the known issues have hindered correct rendering in browsers.

formatting objects, ready for rendering by an XSL formatting engine into PDF output.

Both main preview stylesheets accept a filtered form of Journal Publishing 3.0, in which structured citation elements have been replaced with formatted citations. The assumption here is that all processes will need to include some form of citation processing in order to format citations in an appropriate style. For purposes of this documentation, this filtered form can be called "Simple citation" Journal Publishing 3.0 (it is specified in more detail below). Some projects may have citations that are already formatted using `mixed-citation` elements, which provides for inline punctuation and hence formatting within the source data. Since all the citation preprocessors respect punctuation that is already in place, any of them may be used as the preprocessor in such cases.

If input data should happen already to conform to the "Simple citation" format, a citation preprocessor may be dispensed with altogether. Some projects may wish to do this — "unbundling" the citation formatting to manage it up front or by different means altogether — in order to be able to use a wider range of XSLT processors, including XSLT processors in web browsers such as Internet Explorer, Firefox or Safari, to render their previews directly. The main preview stylesheets are implemented in XSLT 1.0 in order to enable this.

Post-process

Because some Journal Publishing 3.0 data may include MathML, the rendering of MathML may be an issue. The PDF preview stylesheet simply passes MathML through to be rendered by the formatter; many commercial FO engines now support MathML. On the HTML side, however, things are more complex since MathML is not supported by generic web browsers displaying HTML.

Some current browsers now support MathML display in XHTML output. For these purposes, an HTML to XHTML converter has been provided as a post-process. More details are provided below.

Stylesheets in this package

While the preview stylesheets in this distribution are intended as fairly comprehensive solutions, the supporting stylesheets may serve simply as demonstrations of how to achieve common tasks. Projects should feel free to copy, modify or extend any of them in order to support local requirements better.

Summary of stylesheets, dependencies, inputs and outputs

Stylesheet	Purpose	Requires	Input format	Output format
Preprocessors				
`citations-prep/jpub3-PMCcit.xsl`	Formats unpunctuated citations according to NLM/Pubmed guidelines	XSLT 2.0	Journal Publishing 3.0	"Simple citation" Journal Publishing 3.0
`citations-prep/jpub3-APAcit.xsl`	Formats unpunctuated citations according to APA guidelines	XSLT 2.0	Journal Publishing 3.0	"Simple citation" Journal Publishing 3.0

`prep/jpub3-webfilter.xsl`	Excludes content with `specific-use` = "print-only"	XSLT 1.0	Journal Publishing 3.0	Journal Publishing 3.0
`prep/jpub3-printfilter.xsl`	Excludes content with `specific-use` = "web-only"	XSLT 1.0	Journal Publishing 3.0	Journal Publishing 3.0
Main Journal Publishing 3.0 Preview stylesheets				
`main/jpub3-html.xsl`	Formats Journal Publishing data in HTML	XSLT 1.0	"Simple citation" Journal Publishing 3.0	HTML (or XML-wf HTML)
`main/jpub3-xslfo.xsl`	Formats Journal Publishing data as PDF	XSLT 1.0, XSL-FO formatter[i]	"Simple citation" Journal Publishing 3.0	PDF
Postprocessor				
`post/xhtml-ns.xsl`	Converts HTML to XHTML (for support of W3C MathML)	XSLT 1.0	HTML	XHTML
Shell ("wrapper") stylesheets				
`jpub3-APAcit-html.xsl`	Formats Journal Publishing data in HTML with APA-like citations	XSLT 2.0/Saxon	Journal Publishing 3.0	HTML
`jpub3-PMCcit-html.xsl`	Formats Journal Publishing data in HTML with NLM/ Pubmed citations	XSLT 2.0/Saxon	Journal Publishing 3.0	HTML
`jpub3-PMCcit-web-html.xsl`	Formats Journal Publishing data in HTML with NLM/ Pubmed citations, excluding "print-only" content	XSLT 2.0/Saxon	Journal Publishing 3.0	HTML
`jpub3-PMCcit-xhtml.xsl`	Formats Journal Publishing data with NLM/Pubmed citations in XHTML (allows for MathML)	XSLT 2.0/Saxon	Journal Publishing 3.0	XHTML
`jpub3-APAcit-xslfo.xsl`	Formats Journal Publishing data as PDF with APA-like citations	XSLT 2.0/Saxon, XSL-FO formatter [i]	Journal Publishing 3.0	PDF
`jpub3-PMCcit-xslfo.xsl`	Formats Journal Publishing data as PDF with NLM/ Pubmed citations	XSLT 2.0/Saxon, XSL-FO formatter [i]	Journal Publishing 3.0	PDF
`jpub3-PMCcit-print-fo.xsl`	Formats Journal Publishing data as PDF with NLM/ Pubmed citations, excluding "web-only" content	XSLT 2.0/Saxon, XSL-FO formatter [i]	Journal Publishing 3.0	PDF

[i] Tested with AntennaHouse 4.3 with MathML support.

Running the stylesheets

Stylesheets included in this distribution make it possible to run processing pipelines easily. For various reasons, users and publishers may wish to modify these methods.

Using the provided shell stylesheets

The easiest way to apply either of the preview stylesheets is to use one of the shell ("wrapper") stylesheets with Saxon, the XSLT 2.0 engine, as described in the Quick Start and User guides. The shell stylesheets use Saxon extensions to apply a set of transformations in sequence quickly, with a minimum of memory or disk overhead. Each one calls one or the other main preview stylesheet, with different pre- or postprocessors depending on its intended application.

As delivered, each shell stylesheet provided also depends on the module `main/shell-utility.xsl` to function. This module must be in place, and its location in relation to the stylesheets named in the shell must be stable, for things to work.

Because running the XSL-FO previews stylesheet also entails running an XSL-FO formatter, there are slightly different considerations in setting this up for each type of output:

HTML preview pipeline

Either from the command line or using an XML editor or XML/XSLT IDE, invoke Saxon with the shell stylesheet to apply it to a Journal Publishing 3.0 file or to a subdirectory of such files. If a DTD is referenced by the file, make sure it is available. Place the results in the location of your choice. Copy the `jpub-preview.css` file next to the HTML results. You are done.

XSL-FO preview pipeline

Two alternatives are available:

- Configure your XSL-FO engine to run a transformation using a recent version of Saxon (the stylesheets have been tested with version 9), and invoke the transformation from the formatter.
- Run the transformation in the same way as when producing HTML, to generate an XML XSL-FO file that can be processed by the XSL-FO formatting engine.

The tradeoff is that while the first method requires a (one-time) configuration, it is easier for the user. It is a common feature of XSL-FO engines to allow configuring them to use the XSLT transformation engine of your choice.[2]

Extending or writing your own Saxon shell stylesheet

The shell stylesheets included in this package all have the same logic. Stylesheets to be called in and applied to the source data, in sequence, are named in a variable in the stylesheet. Inserting or removing components is as easy as changing this variable.

For example, the stylesheet `jpub3-PMCcit-xhtml.xsl` has the variable `$processes` declared as follows:

[2] The Journal Publishing 3.0 XSL-FO preview stylesheet was tested with the AntennaHouse XSL formatter, version 4.3. Instructions for configuring AntennaHouse to use an alternative XSLT processor can be found in their user documentation; the settings to change are under the **Format / Format Option Setting** menu option. A tip: test your invocation of Saxon from the command line first, so you can paste accurate settings into the dialog box.

```
<xsl:variable name="processes">
  <step>citations-prep/jpub3-PMCcit.xsl</step>
  <step>preview/jpub3-html-preview.xsl</step>
  <step>post/xhtml-ns.xsl</step>
</xsl:variable>
```

Add a step to this declaration:

```
<xsl:variable name="processes">
  <step>prep/jpub3-webfilter.xsl</step>
  <step>citations-prep/jpub3-PMCcit.xsl</step>
  <step>main/jpub3-html-preview.xsl</step>
  <step>post/xhtml-ns.xsl</step>
</xsl:variable>
```

Now, an additional preprocess `prep/jpub3-webfilter.xsl` is run before the citation preprocessing step.

Of course, we recommend copying a shell with a new name and making changes there, in order to preserve the integrity of files in the distribution.

All paths given are relative to the shell stylesheet at the top level of this distribution.

Note that configuration of the output serialization using the `xsl:output` instruction is designated by the shell stylesheet. Accordingly, a shell stylesheet intended to generate plain HTML might have `method="html"` indicated in its `xsl:output`, while a shell stylesheet generating XHTML should have `method="xhtml"`.

Designing and running your own pipeline

The shell stylesheets are provided for convenience, but they do introduce a dependency on the Saxon XSLT processor. You can remove this dependency by implementing a pipeline in the framework of your choice, including shell scripts or batch files, scripting languages, build tools such as Unix `make` or Apache Ant, or full-blown pipeline processors possibly including XProc implementations.[3]

The principle is simple: each step in the pipeline generates results that are taken as the input for the next step in the pipeline. Serialization may follow each step (if each pipeline reads its input from the file system) or may be performed only after the last step (if steps are connected using event streams or if intermediate results are held in memory).

Keep in mind that an XSLT engine will still be required to run the separate steps, and that several of the stylesheets provided do require an XSLT 2.0 processor.

Running without a pipeline

Running a single preview stylesheet standalone is not recommended since no support for formatting of citation elements is provided outside the preprocessors designed for that purpose. Nevertheless, if bibliographic citations are not an issue, either the HTML or the XSL-FO preview transformation can be run in an XSLT 1.0 processor, including client-side (in a web browser or other XSLT-capable display engine).

Tweaks and extensions to the main Preview stylesheets

These stylesheets provide for basic formatting of Journal Publishing data for preview; they are not designed to serve as production stylesheets. Their functionality can be extended or modified, however. It is recommended that modifications make use of the XSLT import

[3] XProc is the W3C pipelining language standard, not finalized at the time of writing.

mechanism: write your modified templates in a separate stylesheet module that imports the main stylesheet module. The modifications can then be maintained seperately in one place.

CSS stylesheet

HTML results are generated with a link to `jpub-preview.css`, given as a relative path that expects the named file to be in the same subdirectory as the HTML result file (or the XML source file if it is processed directly in a browser).

This can be altered at runtime by overriding the value of the `css` parameter supplied to the main HTML Preview stylesheet. This allows a CSS stylesheet with another name to be used or a stylesheet to be located elsewhere. If an empty string is provided, the HTML results will have no link to an external CSS.

Of course, altering `jpub-preview.css` or providing your own CSS stylesheet is an easy way to alter the look and feel of output without editing any XSLT. Many of the basic settings in the HTML presentation, including fonts, rules, colors (including the colors of warnings or generated text) may be customized in this way.

Enabling autonumbering in the XSLT

As described in the User's Guide, the Preview stylesheet does not generate labels with automatic numbering for elements in presentation, except the `fn` and `ref` elements as described there. However, the stylesheets are provided with templates that can be modified in order to provide automated numbering. See the templates in mode "label-text" in the stylesheets.

Specification of "simple citation" Journal Publishing 3.0

Projects that wish to dispense with automated formatting of citations, to extend the automated processing provided, or to develop their own citation formatting logic, will need to know the subset of Journal Publishing 3.0 that can be reliably converted by the preview stylesheets. This profile introduces additional constraints on elements related to bibliographic citations: `element-citation`, `nlm-citation`, `mixed-citation` and their children, along with the related elements `product`, `related-article` and `related-object`.

- `element-citation` and `nlm-citation` are not recognized as such. A preprocessor should convert them into `mixed-citation`, providing their contents with punctuation in the process. (This is what the packaged citation preprocessors do.) If these elements appear, preview stylesheets will present them without formatting.
- Along with literal text (including punctuation), these elements are recognized inside `mixed-citation`, `product`, `related-article` and `related-object`:

  `bold`, `italic`, `monospace`, `overline`, `roman`, `sans-serif`, `sc`, `strike`, `underline`, `inline-graphic`, `label`, `email`, `ext-link`, `uri`, `sub`, `sup`, `styled-content`
- These elements are permitted in citation elements in Journal Publishing 3.0, but are *excluded* from the simple citation profile. A preprocessor should strip them from content or convert them into presentation-oriented elements (listed above), with appropriate punctuation, according to the rules of the chosen citation format. If they appear, these element can be expected to be processed by a preview stylesheet according to default rules, which generally means their content will appear with no formatting consequences:

  `abbrev`, `alternatives`, `annotation`, `article-title`, `chapter-title`, `chem-struct`, `collab`, `comment`, `conf-date`, `conf-loc`, `conf-name`, `conf-sponsor`, `date`, `date-in-citation`, `day`, `edition`, `elocation-id`, `etal`, `fpage`, `gov`, `inline-formula`, `institution`, `isbn`, `issn`, `issue`, `issue-id`, `issue-part`, `issue-title`, `lpage`, `milestone-end`, `milestone-start`, `month`, `name`, `named-content`, `object-id`, `page-range`, `part-title`, `patent`, `person-group`, `private-char`, `pub-id`, `publisher-loc`, `publisher-name`, `role`, `season`, `series`, `size`, `source`, `std`, `string-name`, `supplement`, `trans-source`, `trans-title`, `volume`, `volume-id`, `volume-series`, `year`