Formatting JATS

as easy as 1-2-3

Tony Graham
Mentea
13 Kelly's Bay Beach
Skerries, Co Dublin, Ireland
info@mentea.net
@MenteaXML
http://www.mentea.net

Version 1.0 – 2 April 2014

Formatting JATS

as easy as 1-2-3

Formatting JATS: as easy as 1-2-3 1

- JATS Preview stylesheets
- XSLT 1.0
- XSLT 2.0
- XSLT 3.0

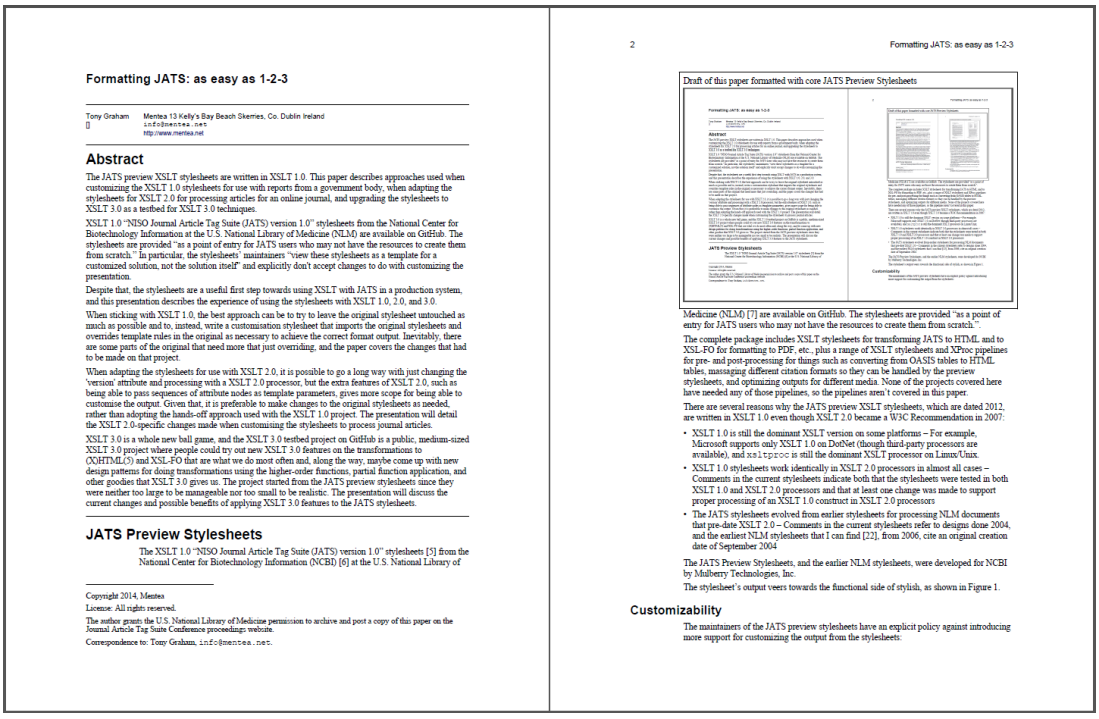
The 1-2-3 comes from using JATS with three versions of XSLT.

JATS Preview stylesheets 2

`https://github.com/NCBITools/JATSPreviewStylesheets`

- XSLT 1.0
- Public domain
- No copyright issues
- Developed for NCBI by Mulberry Technologies

JATS Preview with “selfie” 3



Formatting JATS: as easy as 1-2-3

Tony Graham — Mentea 13 Kelly's Bay Beach Skerries, Co. Dublin Ireland — info@mentea.net — http://www.mentea.net

Abstract

The JATS preview XSLT stylesheets are written in XSLT 1.0. This paper describes approaches used when customizing the XSLT 1.0 stylesheets for use with reports from a government body, when adapting the stylesheets for XSLT 2.0 for processing articles for an online journal, and upgrading the stylesheets to XSLT 3.0 as a testbed for XSLT 3.0 techniques.

XSLT 1.0 "NISO Journal Article Tag Suite (JATS) version 1.0" stylesheets from the National Center for Biotechnology Information at the U.S. National Library of Medicine (NLM) are available on GitHub. The stylesheets are provided "as a point of entry for JATS users who may not have the resources to create them from scratch." In particular, the stylesheets' maintainers "view these stylesheets as a template for a customized solution, not the solution itself" and explicitly don't accept changes to do with customizing the presentation.

Despite that, the stylesheets are a useful first step towards using XSLT with JATS in a production system, and this presentation describes the experience of using the stylesheets with XSLT 1.0, 2.0, and 3.0.

When sticking with XSLT 1.0, the best approach can be to try to leave the original stylesheet untouched as much as possible and to, instead, write a customisation stylesheet that imports the original stylesheets and overrides template rules in the original as necessary to achieve the correct format output. Inevitably, there are some parts of the original that need more that just overriding, and the paper covers the changes that had to be made on that project.

When adapting the stylesheets for use with XSLT 2.0, it is possible to go a long way with just changing the 'version' attribute and processing with a XSLT 2.0 processor, but the extra features of XSLT 2.0, such as being able to pass sequences of attribute nodes as template parameters, gives more scope for being able to customise the output. Given that, it is preferable to make changes to the original stylesheets as needed, rather than adopting the hands-off approach used with the XSLT 1.0 project. The presentation will detail the XSLT 2.0-specific changes made when customising the stylesheets to process journal articles.

XSLT 3.0 is a whole new ball game, and the XSLT 3.0 testbed project on GitHub is a public, medium-sized XSLT 3.0 project where people could try out new XSLT 3.0 features on the transformations to (X)HTML(5) and XSL-FO that are what we do most often and, along the way, maybe come up with new design patterns for doing transformations using the higher-order functions, partial function application, and other goodies that XSLT 3.0 gives us. The project started from the JATS preview stylesheets since they were neither too large to be manageable nor too small to be realistic. The presentation will discuss the current changes and possible benefits of applying XSLT 3.0 features to the JATS stylesheets.

JATS Preview Stylesheets

The XSLT 1.0 "NISO Journal Article Tag Suite (JATS) version 1.0" stylesheets [5] from the National Center for Biotechnology Information (NCBI) [6] at the U.S. National Library of

Correspondence to: Tony Graham, info@mentea.net.

Draft of this paper formatted with core JATS Preview Stylesheets

Medicine (NLM) [7] are available on GitHub. The stylesheets are provided "as a point of entry for JATS users who may not have the resources to create them from scratch.".

The complete package includes XSLT stylesheets for transforming JATS to HTML and to XSL-FO for formatting to PDF, etc., plus a range of XSLT stylesheets and XProc pipelines for pre- and post-processing for things such as converting from OASIS tables to HTML tables, massaging different citation formats so they can be handled by the preview stylesheets, and optimizing outputs for different media. None of the projects covered here have needed any of those pipelines, so the pipelines aren't covered in this paper.

There are several reasons why the JATS preview XSLT stylesheets, which are dated 2012, are written in XSLT 1.0 even though XSLT 2.0 became a W3C Recommendation in 2007:

- XSLT 1.0 is still the dominant XSLT version on some platforms – For example, Microsoft supports only XSLT 1.0 on DotNet (though third-party processors are available), and `xsltproc` is still the dominant XSLT processor on Linux/Unix.
- XSLT 1.0 stylesheets work identically in XSLT 2.0 processors in almost all cases – Comments in the current stylesheets indicate both that the stylesheets were tested in both XSLT 1.0 and XSLT 2.0 processors and that at least one change was made to support proper processing of an XSLT 1.0 construct in XSLT 2.0 processors
- The JATS stylesheets evolved from earlier stylesheets for processing NLM documents that pre-date XSLT 2.0 – Comments in the current stylesheets refer to designs done 2004, and the earliest NLM stylesheets that I can find [22], from 2006, cite an original creation date of September 2004

The JATS Preview Stylesheets, and the earlier NLM stylesheets, were developed for NCBI by Mulberry Technologies, Inc.

The stylesheet's output veers towards the functional side of stylish, as shown in Figure 1.

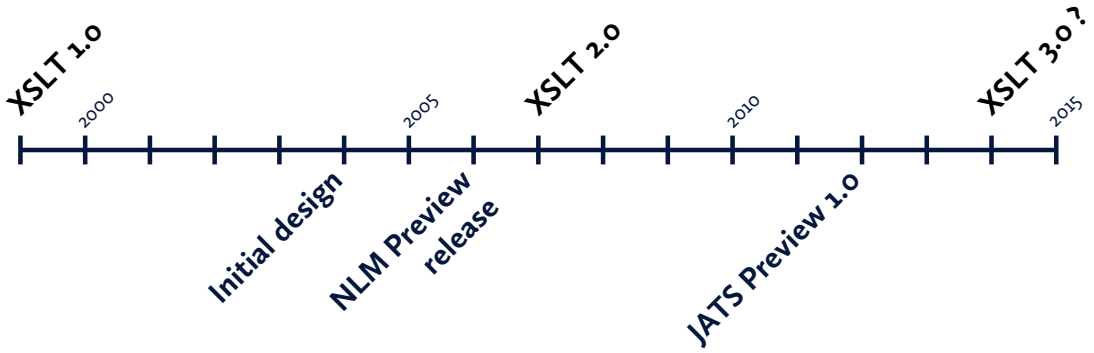
Customizability

The maintainers of the JATS preview stylesheets have an explicit policy against introducing more support for customizing the output from the stylesheets:

Paper for this talk formatted using JATS Preview stylesheet with picture of paper formatted using JATS preview stylesheets.

(‘selfie’ was added to the OED in 2013 so maybe it doesn’t need quotes)

Reconstructed timeline 4



Reconstructed from comments in code, downloads, and emails with Kim Tryka and Tommie Usdin.

Why still XSLT 1.0 in 2012? 5

- XSLT 1.0 still dominant on some platforms
  - .NET
  - Linux/Unix
- Also tested with XSLT 2.0
- NLM stylesheets developed circa 2006/2007
  - One well-known XSLT 2.0 processor
  - Java only

What does it do? 6

- Preprocessing
  - Convert OASIS tables to HTML tables
  - Massage citation format
  - Some require XSLT 2.0
- Formatting
  - XML to HTML
  - **XML to XSL-FO for formatting as PDF**
- Post-processing
  - HTML to XHTML for MathML

The only part that I've needed to use, and the only part being covered, is the transformation to XSL-FO and formatting to PDF.

Customizability 7

"These stylesheets are **provided as a point of entry for JATS users** who may not have the resources to create them from scratch. Because there are many varied implementations of JATS, you should have no expectation that these stylesheets will create production ready files in any arbitrary system. Instead, the stylesheets should be customized for your particular needs."

"Because **we view these stylesheets as a template for a customized solution, not the solution itself**, we will accept changes that fix an actual bug, but we will not merge in changes that we view as "customization". For example, we will accept changes that fix a problem which otherwise leads to failure in creating a final output file, but we will not accept changes that focus on presentational aspects of the final output (such as font changes, margin changes, graphics sizing, etc)."

Statement about customisation from JATSPreviewStylesheets README with added emphasis.

XSLT features supporting customizability 8

- Templates
- Modular stylesheets
- Named attribute sets

Templates 9

- `match` matches a context in source XML
- Content of `xsl:template` instantiated when template is applied

```
<xsl:template match="td">
  <fo:table-cell xsl:use-attribute-sets="td">
    <xsl:call-template name="process-table-cell"/>
  </fo:table-cell>
</xsl:template>
```

Elements in the body of the template not in the XSLT namespace are copied to the result, and elements and attributes in the XSLT namespace are acted on by the XSLT processor.

Modular stylesheets 10

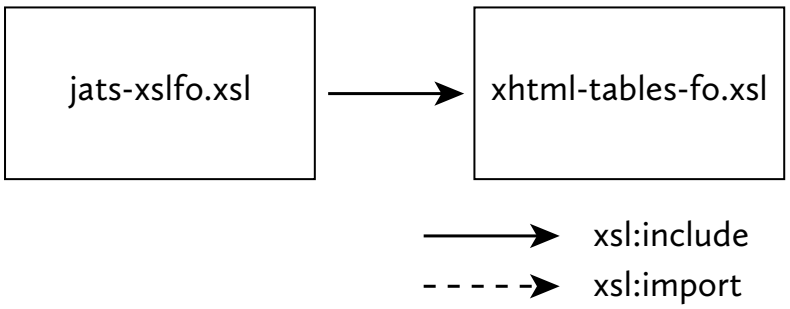
```
<xsl:include
  href = uri-reference />
```

- `href` refers to other stylesheet
- Children of other `xsl:stylesheet` replace `xsl:include`

```
<xsl:import
  href = uri-reference />
```

- `href` refers to other stylesheet
- Imported definitions and template rules *not* part of importing stylesheet
- Have lower *import precedence*

Imports in JATS XSL-FO preview 11



There are more interesting block diagrams later.

Overriding templates 12

- Template in importing stylesheet overrides same context in imported
- Good when overriding complete function of template
- Extra overhead if you just want to change one little thing

Attribute sets 13

- Named set of attribute definitions
- Use in multiple places
- Definitions evaluated in each context where used

```
<xsl:attribute-set name="fig">
  <xsl:attribute name="keep-together.within-page"
      >always</xsl:attribute>
  <xsl:attribute name="id">
    <xsl:value-of select="generate-id()" />
  </xsl:attribute>
</xsl:attribute-set>
```

Since attribute definitions in attribute sets are evaluated each time the attribute set is used, the value of the `id` attribute will be unique to each context.

JATS Preview supporting customizability 14

- Global variables
- Attribute sets
- Named templates

Example customization 15

- Add to attribute set from JATS stylesheets

```
<xsl:attribute-set name="td">
  <xsl:attribute name="line-stacking-strategy"
      >max-height</xsl:attribute>
</xsl:attribute-set>
```

- New attribute set reusing merged `td` attribute set

```
<xsl:attribute-set name="td-small"
    use-attribute-sets="td">
  <xsl:attribute name="line-height">10pt</xsl:attribute>
  <xsl:attribute name="border">none</xsl:attribute>
  <xsl:attribute name="padding-top">0pt</xsl:attribute>
  <xsl:attribute name="padding-bottom">0pt</xsl:attribute>
</xsl:attribute-set>
```

- Override JATS stylesheet in more-specific context

```
<xsl:template
    match="td[ancestor::table[@style = 'small']]">
  <fo:table-cell xsl:use-attribute-sets="td-small">
    <xsl:call-template name="process-table-cell"/>
  </fo:table-cell>
</xsl:template>
```

The `xsl:attribute-set` extends the 'td' defined in the JATS Preview stylesheet.

The new 'td-small' attribute set includes the attribute definitions from all declarations for the 'td' attribute set plus the definitions contained in its definition.

The template matches on a more-specific context than the general-purpose template for `td` in the JATS Preview stylesheets, so in those particular contexts, the XSLT processor uses this template, which adds a different set of attributes to the generated `fo:table-cell` but which still uses the 'process-table-cell" named template from the JATS Preview stylesheets as is used in the original template for `td`.

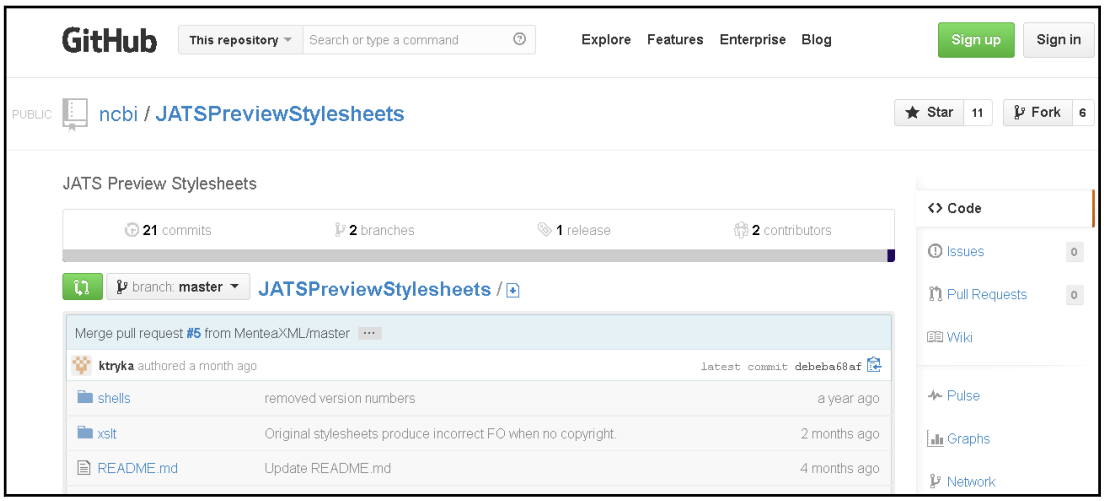
This illustrates in a nutshell how a customisation is able to extend, override, and reuse the constructs in the core JATS Preview stylesheets.

Summary: JATS Preview 16

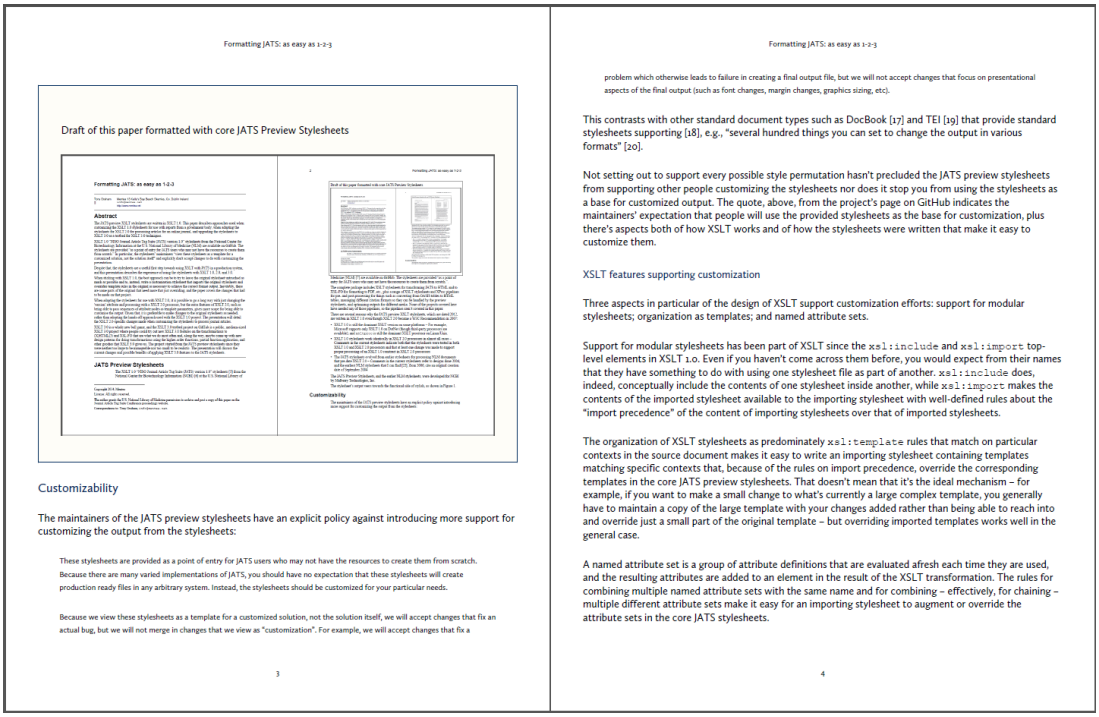
- XSLT 1.0
- Not accepting customisations into core
- Stylesheet structure facilitates customisations

Aside: GitHub 17

- "World's largest open source community"
- Git distributed version control system
- Easy to "fork" – make your own version of projects
- Easy to "pull" merge requests from other projects



XSLT 1.0: Government body 18

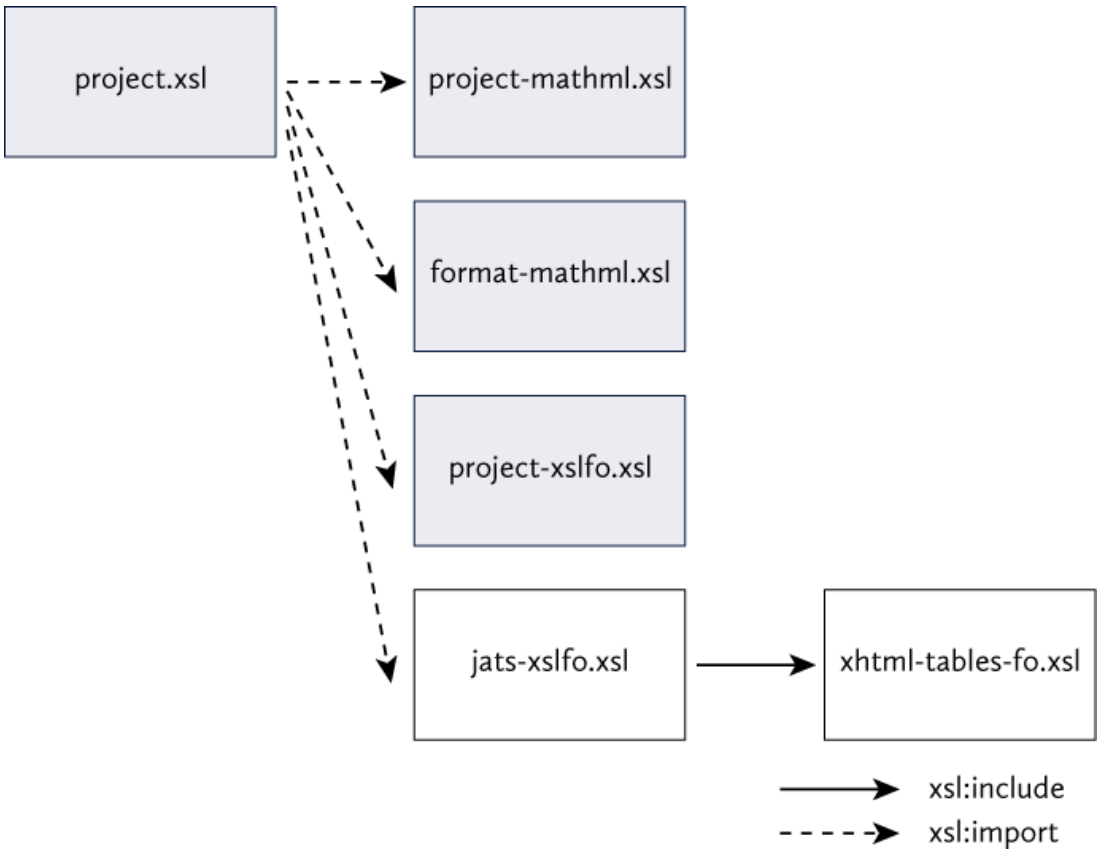


The paper for this talk formatted using XSLT 1.0 stylesheets

Project details 19

- Source: variation on JATS Blue with custom metadata
- Result: similar page design to JATS preview stylesheets
- XSLT 1.0 because...
  - Client preference
  - Body and back content unchanged from JATS
  - Page design similar to JATS preview
- Customisation...
  - Changes in new modules
  - Import JATS Preview stylesheets

Import structure 20



MathML fix-up modules

- Separate modules that can be dropped when problems solved
- `project-mathml.xsl` – add parentheses around display equation number
- `format-mathml.xsl` – workaround too-high accented characters

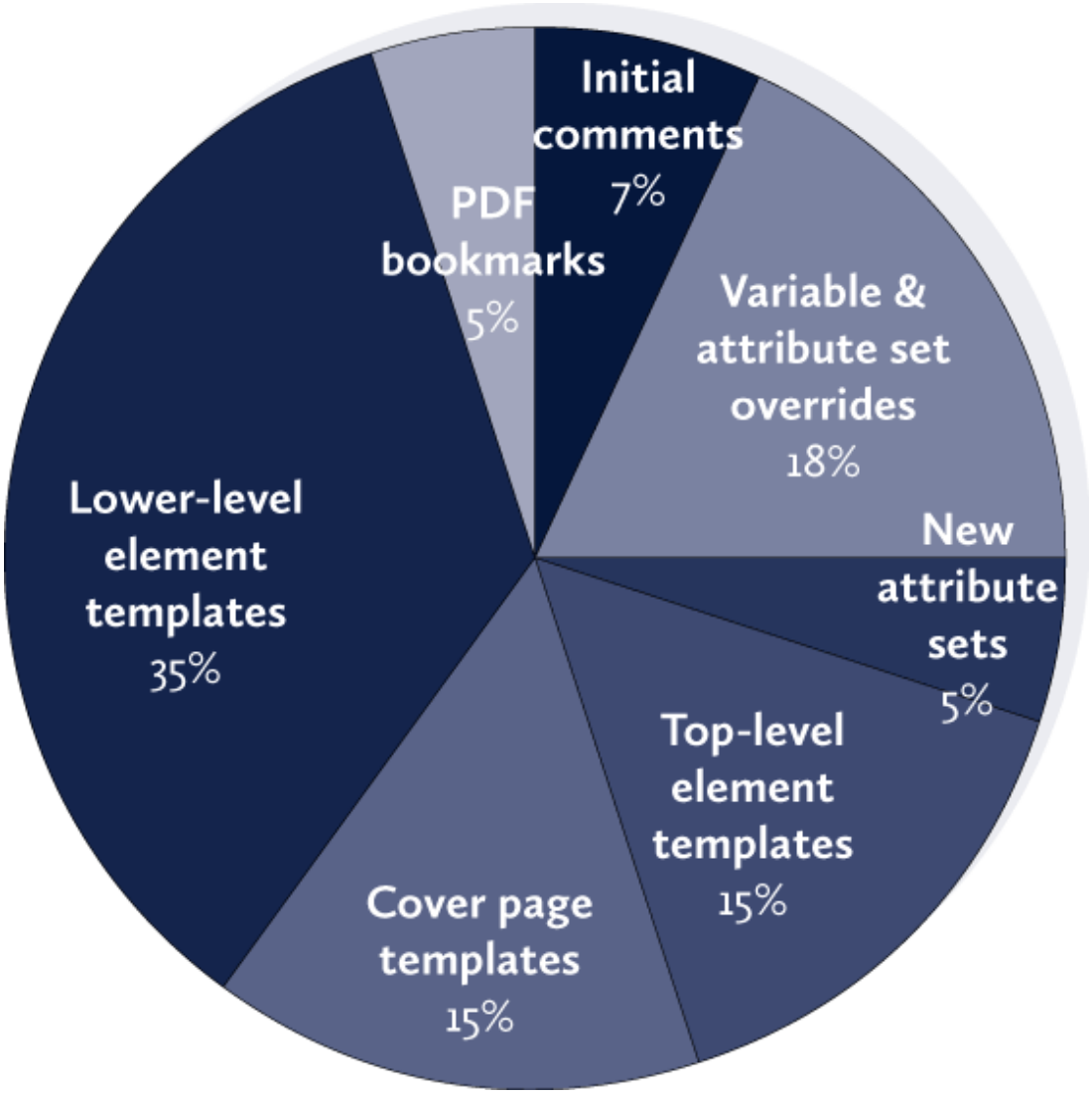
$$\frac{SE(\hat{p})/\hat{p}}{-\ln(\hat{p})} > .175 \text{ when } \hat{p} \leq .5$$

becomes

$$\frac{SE(\hat{p})/\hat{p}}{-\ln(\hat{p})} > .175 \text{ when } \hat{p} \leq .5$$

(Latest formatter has rewritten MathML support)

What's in `project-xslfo.xsl`? 22

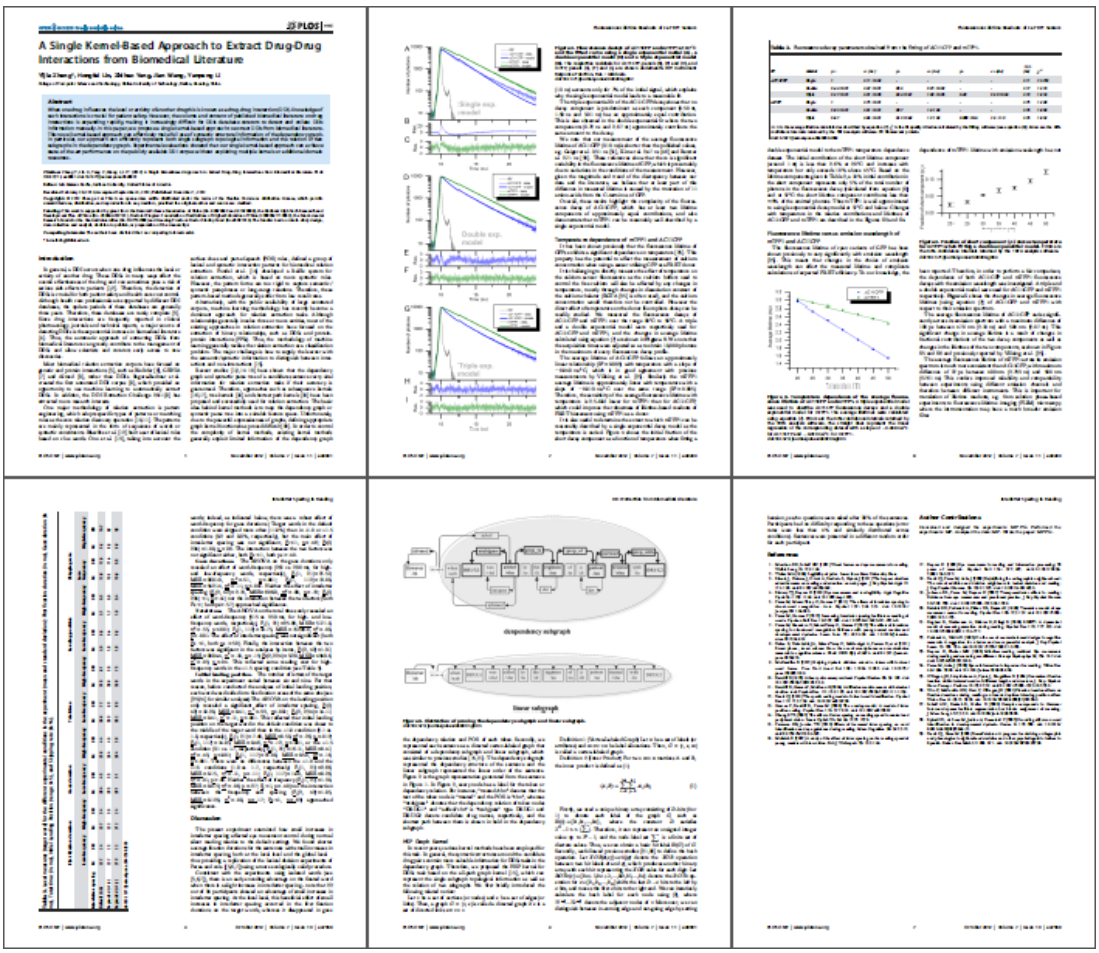


Summary: XSLT 1.0 23

- Customisation on top of JATS Preview stylesheets
- Preview stylesheets provided sufficient hooks

XSLT 2.0: PLOS ONE

24



Sample PLOS ONE pages.

Project details

25

- Peer-reviewed, open-access, online publication
- Public Library of Science
- JATS/NLM markup
- Lights-out batch formatting with XSL-FO
- Previously produced use 3B2 and (presumably) manual fix-up
- XSLT 2.0 because...
  - Big differences in metadata, figure, table handling
  - Needed vendor extensions
- Customisation...
  - Modified version of `jats-xslfo.xsl`
  - Additional XSLT modules

PONE “features” 26

- Figures and tables float to top (or bottom) of page
- Figures column-wide or page-wide
  - No size information in XML
- Figure graphic+caption can’t overflow page
- Tables column-wide, page-wide, or page-high
  - Page-high may be single column
  - May be multiple pages
  - No width indication in XML
  - No row spanning (thank goodness!)
- No figures or tables allowed after start of back matter

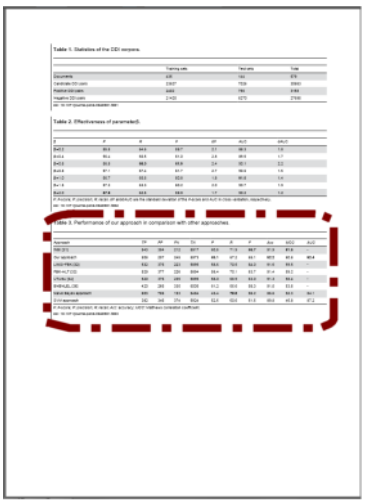
XSLT/XSL-FO “features” 27

- Page-wide floats
  - Vendor extension for column-wide
- Floats don’t break
- Floats only at top of page
  - Bottom-float extension available but unused
- Graphic size not available to XSLT
- *Fire-and-forget* processing

Table handling 28

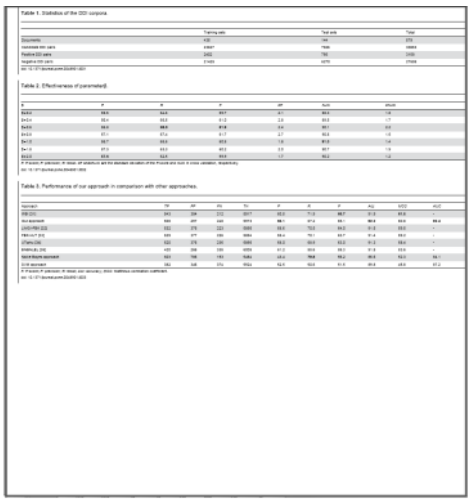
- “Pre-format” tables in three widths on *long* pages
  - Column-wide, page-wide, (width of) page-high
  - Prefix table IDs with string indicating width
- Format to area tree XML
- Compare area trees for each table
- Use width with least area and no overflow
- Recreate as multiple `fo:float` if overflows page
  - Re-use table column widths from area tree to remain consistent

Picking “Best” Tables 29



Page-wide

Column-wide



Page-high

Three tables formatted in each of three widths, with preferred versions highlighted.

Sized and placed tables 30

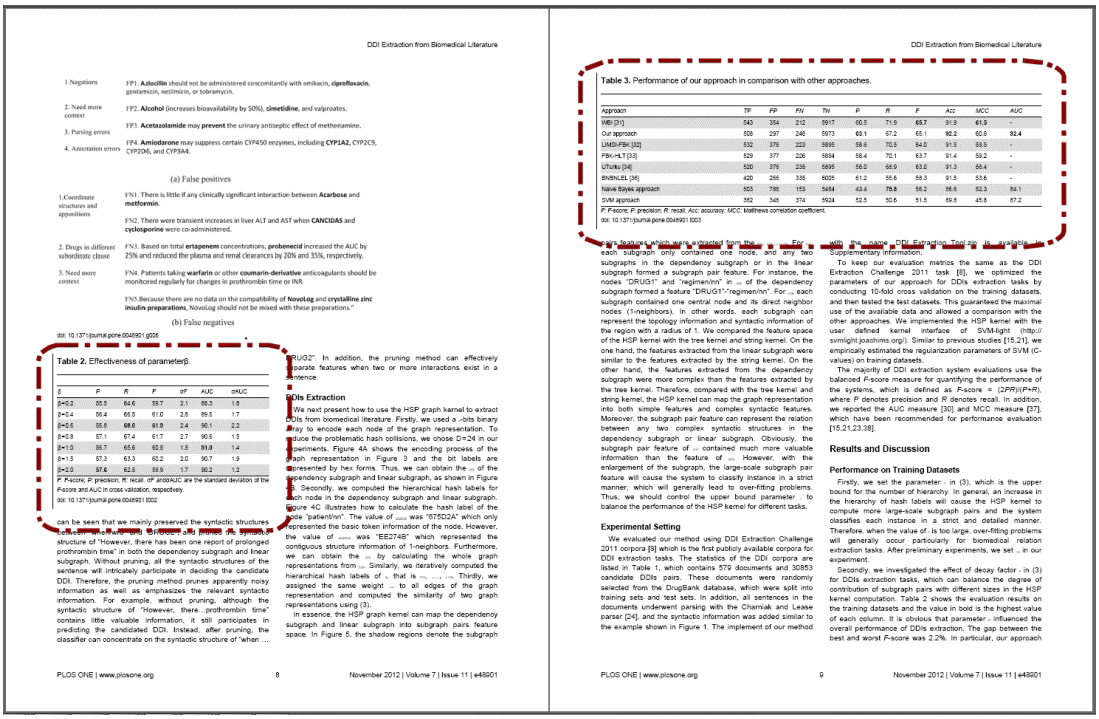


Table 2. Effectiveness of parameterβ.

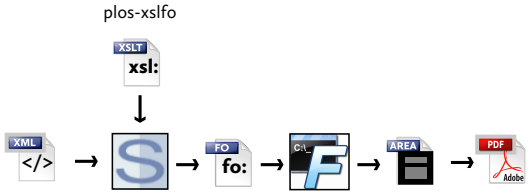
β	P	R	F	dP	AUC	dAUC
β=0.2	55.5	64.6	59.7	2.1	89.3	1.8
β=0.4	56.4	66.5	61.0	2.8	89.8	1.7
β=0.6	56.8	68.0	61.9	2.4	90.1	2.2
β=0.8	57.1	67.4	61.7	2.7	90.6	1.5
β=1.0	56.7	65.6	60.8	1.8	91.0	1.4
β=1.5	57.3	63.3	60.2	2.0	90.7	1.9
β=2.0	57.6	62.5	59.9	1.7	90.2	1.2

Table 3. Performance of our approach in comparison with other approaches.

Approach	TP	FP	FN	TN	P	R	F	Acc	MCC	AUC
MBI [31]	643	354	212	5917	60.8	71.9	65.7	91.9	61.5	-
Our approach	608	297	248	5973	63.1	67.2	65.1	92.2	60.8	92.4
LSWD-FBK [32]	532	376	223	5895	58.6	70.5	64.0	91.5	59.5	-
FBK-HLT [33]	529	377	226	5894	58.4	70.1	63.7	91.4	59.2	-
UTurku [34]	520	376	236	5895	58.0	68.9	63.0	91.3	58.4	-
BNBNLEL [36]	420	266	336	6005	61.2	55.6	58.3	91.5	53.6	-
Naive Bayes approach	603	795	153	5476	43.4	79.8	56.2	86.6	52.3	84.1
SVM approach	582	346	374	5924	62.8	60.6	61.8	89.8	45.6	87.2

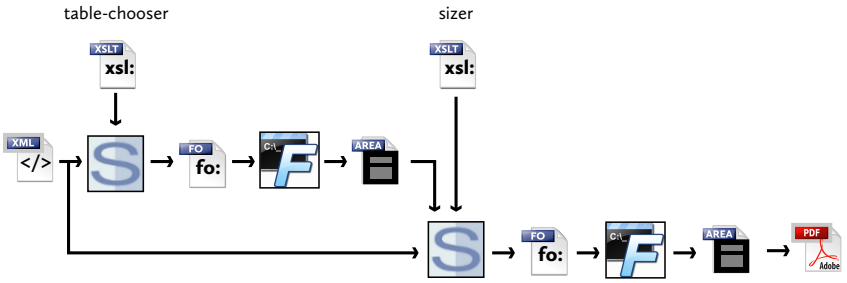
Column-wide and page-wide tables placed on pages.

Usual processing model 31



The conventional XSLT–XSL-FO processing model.

Table-handling processing model 32

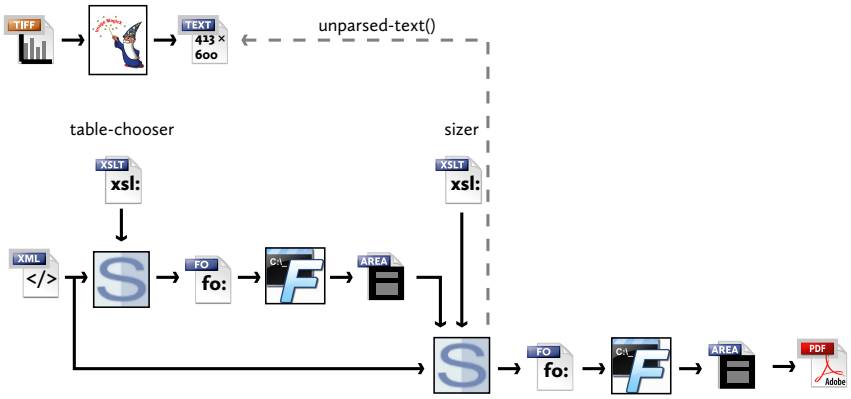


The processing model including preprocessing tables to generate an area tree from which to determine the preferred width for each table.

Graphics handling 33

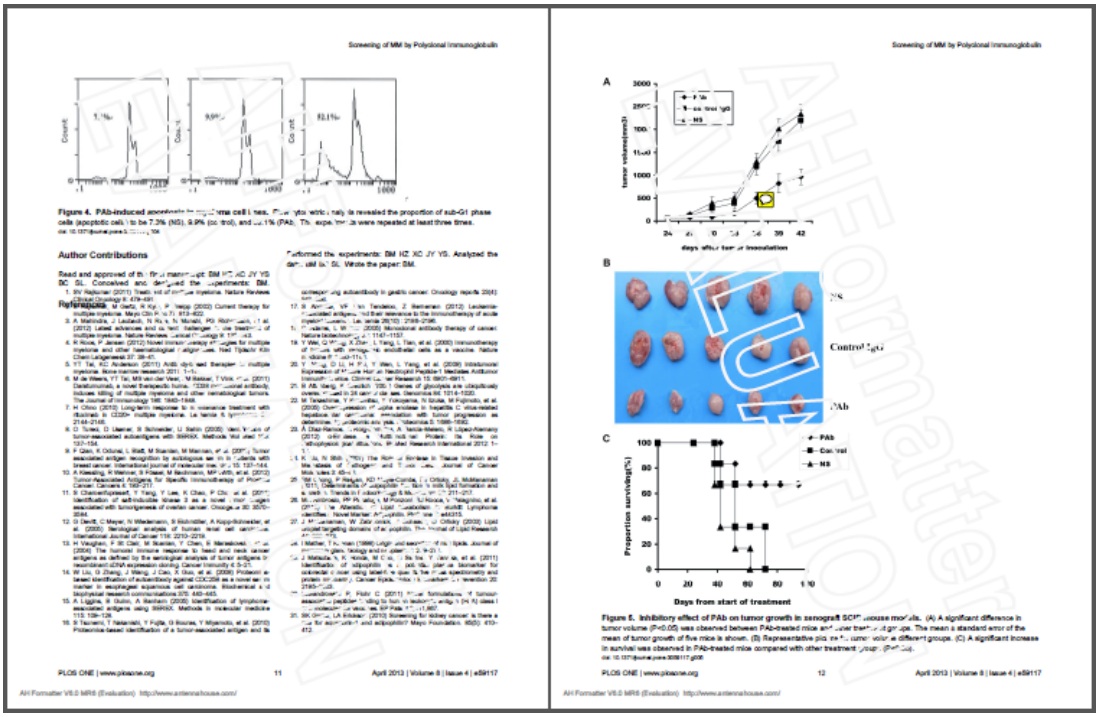
- Get TIFF graphics
- ImageMagick `identify` gives graphic size and resolution
- “Pre-format” caption at both widths to get exact size
- Choose best width
- (Possibly) scale down graphic so caption also fits on page

Figure-handling processing model 34



Processing model when graphics handling added.

Floats after back matter 35

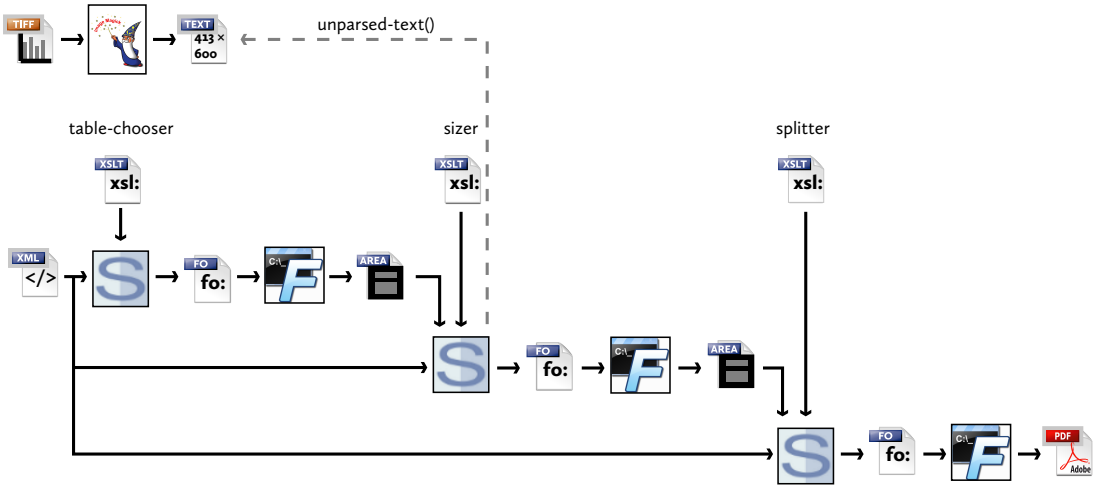


Figures and tables are required to not appear after the start of the back matter.

Splitting at back matter 36

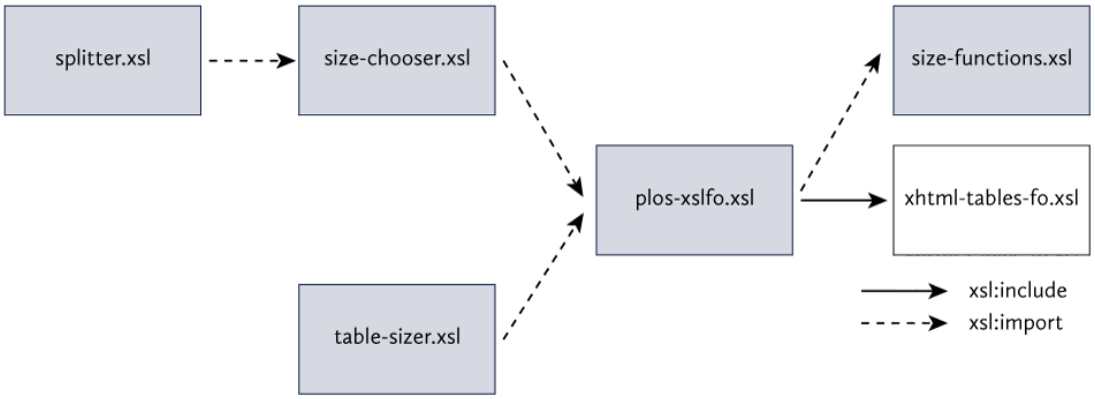
- Format “final” FO with right-width tables and figures to area tree
- Compare positions of first “back” content and last float
  - `back` plus bits from `front`, `body`
- Generate new FO with either one or two `fo:page-seqence`
- If second `fo:sequence`, it contains only back matter so floats in first appear before back matter

Putting It All Together 37



The full processing model.

Import structure 38



All the top-level stylesheets use `plos-xslfo.xsl` for basic formatting.

`splitter.xsl` does everything `size-chooser.xsl` does, and more, so it imports that file rather than importing `plos-xslfo.xsl` directly.

Summary: XSLT 2.0 39

- It shouldn't be this hard
- Column-wide floats require vendor extension
- Navigating area tree isn't easy
- No standard for area tree XML made it harder and even less portable
- Creating new FO and reprocessing easier than rewriting area tree
- EXPath Binary Module (and a TIFF-handling library!) could avoid using ImageMagick
  - Or use vendor extension

XSLT 3.0: xslt3testbed 40

`https://github.com/MenteaXML/xslt3testbed`

- Trying out new XSLT 3.0 features
- Converting existing JATS stylesheets to XSLT 3.0

Why? 41

"...the design process does not include enough feedback; by the time people start reporting their usability experiences, the decisions are difficult to change."

- Early start on patterns and idioms to help adoption
- Find infelicities in spec (and implementations)
- The time is right
  - Project started November 2013
  - XSLT 3.0 Last Call WD – 12 December 2013

> Quote from Micheal Kay, editor of XSLT 3.0 spec: `http://www.biglist.com/lists/lists.mulberrytech.com/xsl-list/archives/201403/msg00332.html`
>
> Motivation comes from looking for a better way to get people using the new version:
>
> - 1997: Wanted to discuss DSSSL so started DSSSList
> - 1998: XSL-List started – people tried every new XSL feature as it came out
> - 2004–2007++: People had working XSLT 1.0 systems and there weren't many XSLT 2.0 processors, so adoption slow
> - 2013–2014: Looking for a quicker win than mailing lists, and people now used to working with GitHub projects

W3C Process 42

- End game for a W3C spec:
  - Last Call
  - Candidate Recommendation
  - Proposed Recommendation
  - Recommendation
- Changes after "Last Call" require more documentation and substantiation

Why JATS? 43

- Simpler than, e.g., DocBook or TEI
- Not a toy
- Potentially useful to authors and archives
- Existing XSLT stylesheets available

Why JATSPreviewStylesheets? 44

`https://github.com/NCBITools/JATSPreviewStylesheets`

- XSLT 1.0
  - Easy for new contributors to add XSLT 2.0-isms
- Public domain
  - No copyright issues
  - XSLT 3.0 stylesheets also public domain
- Explicitly not supporting gazillion customisation parameters, PIs, etc.
  - Simpler processing
  - Fewer user expectations

xslt3testbed goals 45

- Trial different techniques
- Open for dipping into to try random ideas
- Develop patterns and idioms
- Develop XSLT 3.0 package for XHTML tables
  - `xsl:package` new in XSLT 3.0
  - XHTML tables used in many document types

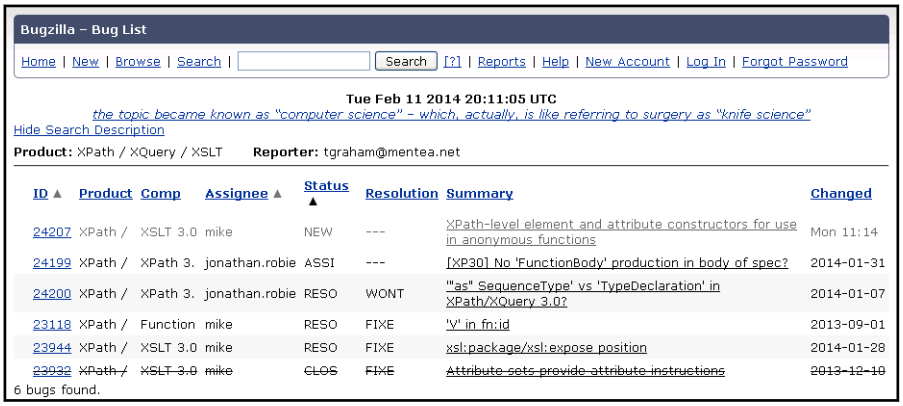
xslt3testbed non-goals 46

- Single best way of doing anything
  - Multiple ways to solve the same problem are okay
- Definitive XSLT 3.0 testbed
  - It's easy to fork and make your own version
- Complete stylesheet for all of JATS
  - Existing stylesheets don't cover everything yet either

Results so far 47

- Trying out maps, anonymous functions, and `xsl:iterate`
- Small advances in multiple areas
- Both XSL-FO and XHTML stylesheets
- More details in XML Prague 2014 talk
  `http://www.mentea.net/resources/xslt30testbed-slides.pdf`

6 W3C Bugzilla bu^H^Htickets so far 48



Bugzilla – Bug List

Home | New | Browse | Search | Search [?] | Reports | Help | New Account | Log In | Forgot Password

Tue Feb 11 2014 20:11:05 UTC

the topic became known as "computer science" – which, actually, is like referring to surgery as "knife science"

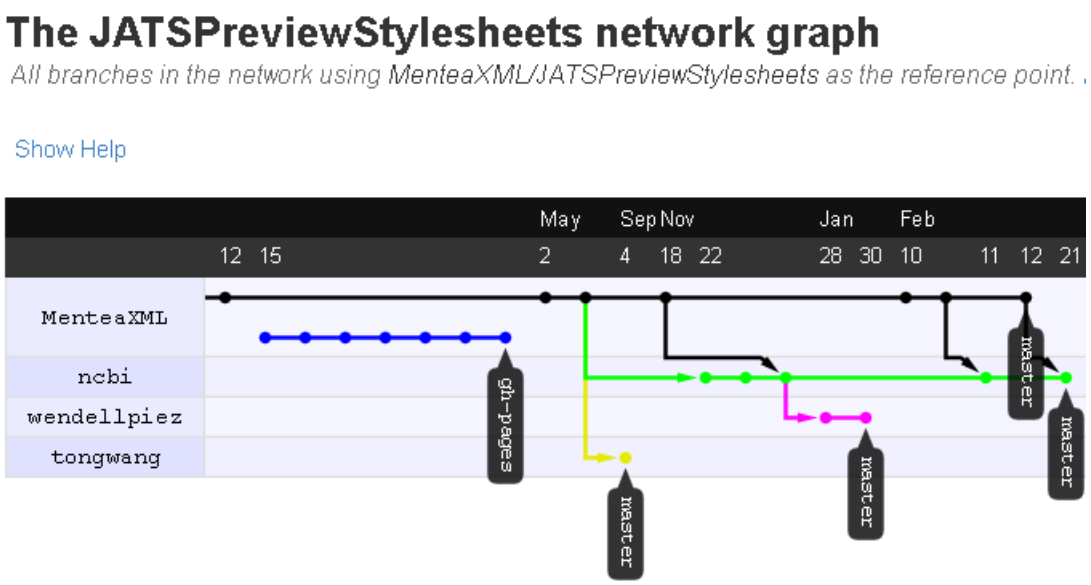
Hide Search Description

Product: XPath / XQuery / XSLT **Reporter:** tgraham@mentea.net

ID ▲	Product	Comp	Assignee ▲	Status ▲	Resolution	Summary	Changed
24207	XPath /	XSLT 3.0	mike	NEW	---	XPath-level element and attribute constructors for use in anonymous functions	Mon 11:14
24199	XPath /	XPath 3.	jonathan.robie	ASSI	---	[XP30] No 'FunctionBody' production in body of spec?	2014-01-31
24200	XPath /	XPath 3.	jonathan.robie	RESO	WONT	"as" SequenceType' vs 'TypeDeclaration' in XPath/XQuery 3.0?	2014-01-07
23118	XPath /	Function	mike	RESO	FIXE	'V' in fn:id	2013-09-01
23944	XPath /	XSLT 3.0	mike	RESO	FIXE	xsl:package/xsl:expose position	2014-01-28
~~23932~~	~~XPath /~~	~~XSLT 3.0~~	~~mike~~	~~CLOS~~	~~FIXE~~	~~Attribute sets provide attribute instructions~~	~~2013-12-10~~

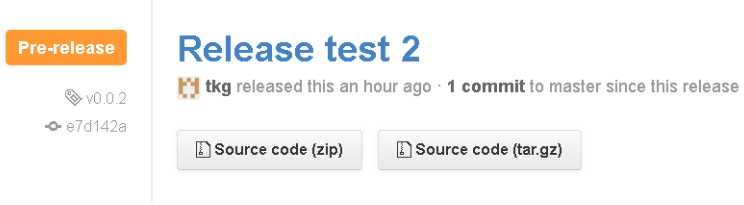
6 bugs found.

5 JATSPreviewStylesheets patches so far 49



Other results 50

- One XSLT processor bug
- One change to Wendell Piez's JATS Oxygen plug-in
- Technique for hosting Oxygen plugins on GitHub



Summary: XSLT 3.0 51

https://github.com/MenteaXML/xslt3testbed

- The time is right
- Useful in multple arenas
- Results summarised on project wiki and http://inasmuch.as/
- Well suited for trying things out
- Go fork and multiply

Conclusion 52

- JATS Preview stylesheets:
  - Explicitly don't support customisation
  - Good basis for your own customization
- Customise by:
  - Layer on top of existing styleheets
  - Modify your copy of the stylesheets
- Usable with XSLT 1.0, 2.0, or 3.0

References 53

- slide 41 – Micheal Kay
  http://www.biglist.com/lists/lists.mulberrytech.com/xsl-list/archives/201403/msg00332.html
- slide 42 – W3C Process Document
  http://www.w3.org/2005/10/Process-20051014/tr.html
- slide 48 – Bugs so far
  https://www.w3.org/Bugs/Public/buglist.cgi?email1=tgraham%40mentea.net&emailreporter1=1&emailtype1=substring&product=XPath%20%2F%20XQuery%20%2F%20XSLT&query_format=advanced

Appendix A
About

Tony Graham

Tony Graham has been working with markup since 1991, with XML since 1996, and with XSLT/XSL-FO since 1998. He is Chair of the Print and Page Layout Community Group at the W3C and previously an invited expert on the W3C XML Print and Page Layout Working Group (XPPL) defining the XSL-FO specification, as well as an acknowledged expert in XSLT, developer of the open source xmlroff XSL formatter, a committer to both the XSpec and Juxy XSLT testing frameworks, the author of "Unicode: A Primer", a member of the XML Guild, and a qualified trainer.

Tony's career in XML and SGML spans Japan, USA, UK, and Ireland, working with data in English, Chinese, Japanese, and Korean, and with academic, automotive, publishing, software, and telecommunications applications. He has also spoken about XML, XSLT, XSL-FO, EPUB, and related technologies to clients and conferences in North America, Europe, and Australia.

Mentea

Mentea specialises in consulting and training in XML, XSL-FO, & XSLT. We are available for on-site meetings and classes, worldwide, but as well as on-site meetings and classes, we routinely keep in touch with clients though email, Skype, instant messaging, and telephone and through a secure, per-client or per-project wiki, revision-control, and issue-tracking system.

Our staff have been working with markup since 1991, with XML since 1996, and with XSLT/XSL-FO since 1998. Based in Dublin, Ireland, Mentea has a global reach: in recent projects, we have helped companies and organisations in the USA, Ireland, England, and France with their XSLT, XSL, and XML, including:

- Writing Schematron for a professional body
- Augmenting a XSLT-based automated schema documentation system that produces both HTML and PDF
- Extending FOP for a software company
- Training in XML, oXygen, DocBook, XSLT 2.0, and XSL-FO
- Formatting JATS to PDF for a scientific journal
- Writing XSLT stylesheets to convert non-XML into XML then into EPUB
- Writing XSLT to convert Excel into XML for a commercial bank

Mentea presents a unique range of skills extending beyond XML and XSL-FO/XSLT into Unicode, SGML, DSSSL, and programming in C, Java, Perl, Lisp, and other languages.

We understand how markup works. Our staff has worked with markup in Japan, USA, UK, and Ireland as user, consultant, and developer, with data in English, French, Chinese, Japanese, and Korean, with academic, automotive, publishing, software, and telecommunications applications, and in the Web Services and document processing arenas.

We are also interested in applying the tools for ensuring software quality – unit testing, code coverage, profiling, and other tools – to XML and XSLT/XSL-FO processing.

Through our associations and affiliations with other consultants around the world, we can call on extra help for large or specialised projects.

MENTEA