# Publicon 1.0 Demonstration

*Structured Document Authoring with Mathematical Content*

**André Kuzniarek**
Wolfram Research, 100 Trade Center Drive, Champaign IL 61820
andre@wolfram.com

*Publicon* 1.0 is a structured document authoring system based on the *Mathematica* notebook front end and enhanced with custom document processing features. Technologies developed by Wolfram Research as support features in *Mathematica* are exploited here with the primary emphasis on an easy-to-use interface for creating and editing technical content for publication. This demonstration will cover *Publicon*'s palette- and menu-driven interface, conversion processes for XML/MathML/HTML and TeX, a reference management system, and general customization features. Presentation by Andre Kuzniarek, Document Technology Manager at Wolfram Research and lead *Publicon* developer.

March 14, 2002

## Overview

The emergence of XML has brought the concept of structured documents to the mainstream. Anyone who has slapped together a web page in HTML has had some exposure to document markup, but highly structured and robust document data is usually left to system developers and database programmers to sort out. XML editing applications tend to be generalized tools for analyzing document trees, particularly to assist with debugging DTDs. However, most document authors would prefer to focus on creating content rather than on document formatting semantics and abstractions. *Publicon* offers the means to that end, while providing enough access to what's under the hood to allow for user customization.

*Publicon* enters the field of document composition tools with an emphasis on maintaining rigid document structure for authors of technical content, since it exploits *Mathematica*'s existing notebook specification with support for interactive 2-D mathematical typesetting (minus any of *Mathematica*'s evaluation capabilities). This revolutionary typesetting system helped to serve as the model for MathML and the notebook "container" is extremely similar to today's DOM. What remained to be solved was a GUI interface for providing access to all the technical document features commonly required by the scientific publishing community, and to provide a practical XML export/import channel.

## Interface Goals

Complicated document markup is easiest to control at the level of raw code, like TeX's macro language. It's up to the user to enter everything just right, and there's enough of a learning curve to become intimidated. LaTeX simplifies TeX composition with macros targeted to specific publishing environments, but it still doesn't free the user to focus on a WYSIWYG and dynamic document representation. Microsoft Word provides the WYSIWYG but lacks convenient access to technical publishing features other than style sheets. The goal of *Publicon* is to combine the approach of targeting specific technical publishing environments (APS, AMS, PubMed/BioMed Central, U of I Thesis style) with a convenient GUI interface.

While *Publicon*'s print output is certainly of professional quality, it lacks some of the more detailed or sophisticated page description features common to expensive page layout applications or TeX, hence it's forward-looking emphasis on authoring structured document data that can be exploited in any context. For version 1.0 that emphasis is narrowed more specifically to articles, rather than large-scale projects like books. Book-specific tools for indexing, tables of contents, and general file management will follow in version 2.0.

With the target document type well controlled, we can focus the interface on document-specific palettes which drive the authoring process. Buttons create templates or placeholders for each of the document elements with underlying structures hidden from the author's view and editing restricted to document content. Data that requires controlled entry is managed through forms, but the objective is to limit these forms to as few as possible. Buttons within the document palettes operate via *Mathematica*-defined functions, some of which accept a range of arguments for convenient user customization. Simple buttons can be constructed on the fly with standard *Publicon* menu features. Customized document palettes can be dropped into *Publicon*'s application layout (like plug-ins) to enhance the available selection of composition tools.

While *Publicon*'s interface is palette/menu/mouse intensive, there is almost an equal degree of support for keyboard-only input. This is particularly true for math, one of *Publicon*'s (and *Mathematica*'s) most valuable document composition features.

## Interface Examples

### General Menu and Interface Layout

1. Menus and items particularly relevant to MathML conference goers
2. Typesetting and formatting tools
3. Default document palette

### Custom Document Palettes

1. Physical Review
2. AMS
3. BioMed Central
4. arXiv
5. Thesis

### Notes and Cross-references Tools

1. XRef targeting and insertion
2. Note entry and formatting

### Bibliography Tools

1. Database format
2. Keyboard shortcut for citation insertion
3. Button/dialog for citation insertion
4. Database editing via a dynamic form
5. Style sheet editing for customization of bibliography formats
6. Transparent support for custom database entities via user-defined reference templates
7. Automatic bibliography generation with complete underlying structure and formatting

## Document Entities and Style Sheets

*Publicon* is based on *Mathematica* 4.1 and therefore is working with an older style sheet paradigm that is not as flexible as CSS, particularly with regard to option inheritance mechanisms. This will be improved in future versions, but for now all *Publicon* style sheets have to use the same list of explicit entities. That list will be a union of entities informed by elements supported in:

- Common TEX environments
- Targeted XML environments
- *Mathematica*
- Bibliography formats: APS, APA, Chicago Manual
- *Publicon*-specific features
- Future book features (TOC, index, chapter ornaments)

## Document Channels

The payoff for adhering to strict document structures is in the ease with which a given document can be transformed into anything else. *Publicon* supports WRI's XML formats by default (NotebookML, ExpressionML), but also supports specific other DTDs including PubMed/BioMed Central and of course MathML.

1. XML and MathML export/import
2. TEX export
3. Techexplorer
4. HTML for Microsoft Word

## Customization

*Publicon* 1.0 is intended as a targeted solution for specific publishing contexts. It's customization options are limited to style sheet modifications, user defined palettes and document palette tweaks. A secondary product is in the works that will function as a consulting platform and will support the *Mathematica* programming language features specific to document manipulation and XML translations.

1. Style sheets (TEX and CSS configuration and general notebook style editing)
2. User defined palettes
3. Document palette customization using source notebooks

## Conclusion

We've attempted to address emerging needs for document authoring by building upon *Mathematica*'s 6 year-old notebook technology with some new XML-specific functionality, targeted to a specific audience that should be receptive to our strongest feature: our user-friendly math typesetting system. However, the *Mathematica* programming language offers powerful control over interface elements and particularly document conversion to and from just about any XML (or TEX) environment. *Publicon* exploits that programmability to provide a specifically targeted authoring solution, but it will also become applicable as a general solution for customized authoring tools in any document-intensive environment.