

The Xylia Toolkit: Editing XML with Java

Jonathan Milley, Theodore S. Norvell, and Dennis K. Peters
Faculty of Engineering and Applied Science,
Memorial University of Newfoundland,
St. John's, NFLD, Canada

Abstract:

This paper presents an overview of the Xylia software library, this library is being designed for the development of user friendly visual editors for the eXtensible Markup Language (XML)[1]. XML is a standard for markup languages. Each language conforming to XML is similar in structure to HTML, but has its own set of permissible tags, and rules about how they may relate. XML is designed to store structured data in a standardized, cross-platform, textual format. The Java programming language and the Swing toolkit from Sun Microsystems [2] are being used to build the library, which is suitable for application developers who wish to create simple, easy to use, robust editors for specific XML languages. Since both technologies are intended to promote cross-platform and cross-application interoperability, they are a good match for each other.

1. Overview

The world is embracing the Internet, and the World Wide Web, at an incredible pace. The last 10 years has seen the Internet grow from a small community of mainly academics and governmental institutions, into a world-wide commercial and recreational phenomenon. The possibility of reaching a global audience is more and more a common occurrence. The troubles this brings, while not overpowering the benefits, are diverse and myriad. Simply by the user visiting a web page, software from one Internet source may be used to process data from another Internet source, while running on a browser/operating system/hardware platform from third, fourth, and fifth sources. The universal nature of the world-wide web means that standards for software, platforms, and data are not just desirable, but required. This is where XML (the eXtensible Markup Language) and Java make their claim to fame. Java is a programming language that provides the capability of writing software that will run on almost any device, from those that fit into the palm of your hand, up to the largest supercomputer, and a great many in between. XML sets a standard for creating markup languages, allowing you to create custom document formats that are compatible with commodity software. To facilitate the programming of editors for XML documents, we are creating a software library in Java. The Xylia Toolkit provides a way to add XML editing capabilities to Java programs. It builds upon and complements the capabilities of the standard Java libraries. Xylia isn't an XML editor in and of itself; it is a building block; combine it with Java's user interface objects, and you can quickly build and tailor an XML editor to fit your needs.

2. Java

The designers of the Java language set out to create a pure object-oriented language, without what they saw as the shortcomings and flaws in its predecessors. Java has now evolved into a language used for not just small applications, but also large software systems.

The result of Java being platform independent is that it has become the most popular way for distributing applications on web pages, through the Internet. It is in this arena that Java excels, but it is also used in standalone programs.

For the Xylia project, the greatest benefit of using Java is its powerful standard libraries. These libraries provide the key functions needed for user interfaces, file access, etc. The Xylia project relies on the Swing component library, especially its text package, and the HTML classes in particular. The HTML library supports the creation of HTML editors and limited support for the Cascading Style Sheets standard. This imposes some limitations on any derivative works, but provides much of the low-level functionality needed to build on.

```
<?xml version="1.0" ?>
<message>
  <from>Jonathan Milley</from>
  <to>The Audience</to>
  <re>Introduction to XML</re>
  <text>
    So you see, I used tags similar to
    HTML, but defined my own.
  </text>
</message>
```

Figure 1

3. XML (the eXtensible Markup Language)

XML is the other key technology for Xylia. XML is a standard developed by the World Wide Web Consortium (W3C). The W3C is also responsible for many other standards used on the Internet, such as HTML, and Cascading Style Sheets. XML is not truly a markup language in and of itself; it is a standard that defines how to specify markup languages.

There are many existing XML based markup languages such as the Wireless Markup Language (WML), used for displaying content on handheld devices like cell phones; the Chemical Markup Language (CML), used for storing descriptions of chemical compounds; the Math Markup Language (MathML), for storing mathematical formulae and proofs, etc.

XML also can be used as a way to pass data between different software packages, over networks, etc. Because it is text based, it has great compressibility, and error checking capabilities. Using XML to transfer data has downsides such as exposing data; but the idea behind the push for XML is interoperability, and a move away from proprietary file formats.

Figure 1 shows an example XML document. Like HTML documents, XML documents are trees. The nodes of the tree are called its *Elements* and the leaves are sequences of characters. Each element is associated with a *tag*, chosen from a finite alphabet, and a set of *attribute/value* pairs. Each XML language is specified by providing, among other information, a set of possible tags and, for each tag, a regular expression over the set of tags. A document is considered to be *valid* if, for each of its elements, the sequence of tags on its children agrees with the regular expression associated with its tag. The specification of an XML language is stored in a document known as a Document Type Definition (DTD)[1].

To define how an XML document is to be rendered by a program such as a web browser, or an editor created with the Xylia Toolkit, one may define a Cascading Style Sheet [3], which defines the display attributes (text size, colour, margins, etc.) for each element, based on its tag.

There are many useful tools for reading, writing, and checking the validity of XML. There are also many editor programs for editing XML, but most use a tree view of the document, providing a low level editing view.

4. The Xylia Toolkit

The Xylia project is a toolkit (library) for Java that adds the capability of WYSIWYG (what you see is what you get) editing of XML documents. Xylia's interface is taken from one of Java's core libraries, an HTML editor toolkit. The benefits of this are that anyone who has used the HTML toolkit should find no problem using Xylia. The goal is to make it easy for anyone to edit XML documents, and to embed this functionality into other applications. One immediate use of the project is the SIMPLE system [4]. SIMPLE defines a way to describe a mathematical proof (in an XML document) in such a manner that it

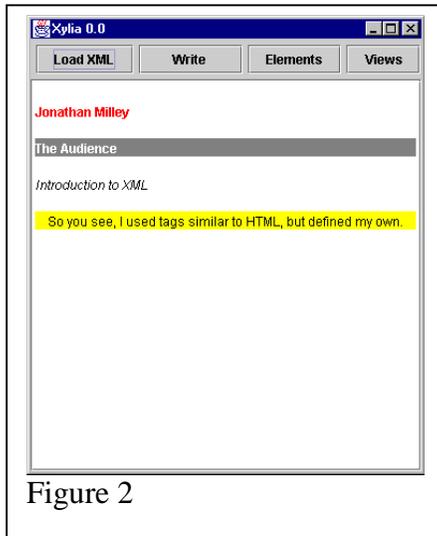


Figure 2

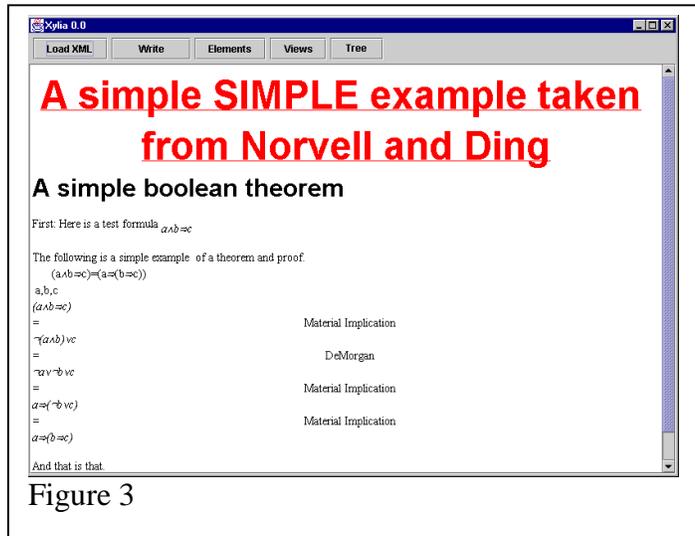


Figure 3

may be verified by a proof-checker. Xylia will make it easy to create and edit proofs that will be verified by another program, and avoid the troubles of tediously editing the XML document in a text editor.

Xylia is really an extension of Sun Microsystems Swing Library for Java. The Swing library provides a great many user interface classes, but also the underlying functionality for representing and editing documents, whether plain text, rich text, or HTML. These libraries were designed for extensibility, and are very modular. Since Xylia implements the same interfaces as the HTML library, an HTML editor program can easily be converted to XML and vice versa. The added benefit of this is that programmers can take advantage of previous skills, and make use of existing documentation and examples meant for HTML, but which also can apply to Xylia.

The library is made up of a collection of classes that work together to achieve the WYSIWYG editing capability. The XMLDocument class is responsible for reading in an XML document, storing it as a tree of elements, and low level manipulation of those elements (inserting at specific locations, removing elements, modifying elements, etc.) The XMLEditorKit class is responsible for providing higher level editing capabilities such as inserting at the cursor position, inserting specific patterns, and providing an XMLViewFactory object. The XMLViewFactory class is responsible for creating the appropriate View objects for each XML element. The View objects are responsible for painting the XML elements on the screen.

The XMLDocument class uses the XMLParser class to actually accomplish reading, validating and verifying an XML document. The XMLParser class, in turn, uses a SAX (simple API for XML) Parser which is part of Sun Microsystems's JAXP [5] library (Java API for XML Processing). These Xylia classes work together with other Swing classes. The JEditorPane is the visual component of the editor; it handles all the user input (cursor position, character inserts, etc.) and organizes the views for each element into a coherent document.

For the application developer using the Xylia library, a simple editor can be built by making an instance of JEditorPane and setting its EditorKit to be an instance of XMLEditorKit, and then having the JEditorPane read an XML file. Of course there are many ways of accomplishing the same goal. The Xylia library's API is the same as for HTML.

The chief difficulties in creating this library came from the HTML classes we derived from; they imposed certain limitations due to being hard coded for HTML. Since HTML is composed of only a limited selection of tags, the library has a lot of its behaviour hard coded for each tag. This sort of tailoring to the specific tags isn't applicable to XML, since each XML language has a different set of tags. Also because the structure of HTML is known beforehand, the HTML library is simplified by making formatting (view

creation) decisions based on the behaviour of each HTML tag. This doesn't work for XML because it is unknown what the structure of a document is going to be until the document type definition and style sheet are read in. Because of these behaviours, and others, which are targeted towards HTML, a number of the HTML libraries functions had to be overridden and reimplemented for Xylia

Figures 2 and 3 show two examples of XML rendered in an editor based on Xylia. Figure 2 is the small piece of XML shown earlier in Figure 1, and Figure 3 is taken from the SIMPLE system. The toolkit is not yet fully functional, but it does a decent job of rendering the XML. Once it is rendered, it is of course also editable, and the editor provides the functionality to insert elements, delete elements, etc.

5. Conclusions and Future Work

Java is an excellent development environment for a wide variety of applications, but obviously it isn't right for every situation. XML is also an excellent tool for nearly any situation where a data store is needed, but it can't fill every need. Traditional large-scale data bases are clearly needed in large distributed environments. Xylia fills a gap in the currently available Java libraries, and opens an easy avenue for XML editing to Java developers. All these tools are in constant development, and new and exciting things are definitely in the works.

Xylia fills a need for an XML editing library that isn't readily available. There are several XML editor packages that operate similarly to Xylia, however they are closed source, and unavailable for integration into another software package. Xylia, when completed, should allow researchers and other developers an easy route to XML editor integration.

For the future we hope to further the project, and bring it to a point where it is as functional as the HTML library, and beyond. We plan to design for extensibility, allowing 'plug-in' capabilities, and easily configured behaviour.

The goal of this paper is to introduce the ideas of Java and XML to those that aren't familiar with them, and to give an idea as to what the goals of the Xylia project are. It is hoped that once we finalize an initial version of the Xylia toolkit, we will make it available to those interested in making use of it.

6. References

- [1] World Wide Web Consortium (W3C), "Extensible Markup Language (XML)", July 2001, <http://www.w3.org/XML/>.
- [2] Sun Microsystems, Inc. "java.sun.com – The Source for Java™ Technology", October 2001, <http://java.sun.com/>.
- [3] World Wide Web Consortium (W3C), "Cascading Style Sheets", October 2001, <http://www.w3.org/Style/CSS/>.
- [4] Theodore S. Norvell, Zhikai (Jack) Ding, "An Environment for Proving and Programming," in *Proceedings Of Newfoundland Electrical and Computer Engineering Conference*, 1999, section F, paper 1 (Alternative: <http://www.engr.mun.ca/~theo/Publications/index.html>).