RegexXMLReader Documentation

An Implementation of the XMLReader Interface

Elizabeth Barham

RegexXMLReader Documentation: An Implementation of the XMLReader Interface

by Elizabeth Barham

$Revision: 1.12 $ - $Date: 2003/09/20 03:29:32 $

To Jesus Christ: You were there when I needed You.

Table of Contents

List of Tables

List of Examples

Chapter 1. Introduction

1.1. Purpose

RegexXMLReader exists to facilitate turning an arbitrary text document into XML by implementing the XMLReader interface. This basically means that given a special XML stylesheet and text file, the RegexXMLReader will read in the text document and generate SAX events, SAX events which can be read by a transformation engine, a custom application, or an XML serializer, for example.

1.2. History

RegexXMLReader came about because I was working with a non-standardized text document in MacRoman format. In order to parse it, the parser needed to discern what each part is by character make-up (such as all-caps, a line ends with a certain number, etc.). Further, since the incoming text-file was in a rather unusual encoding, especially in the GNU/Linux environment, special care was needed upon reading that document. Fortunatly, Java handles MacRoman with no problem and a special InputSource was created that utilizes Java's text decoding ability.

1.3. Rationale

Essentially, RegexXMLReader views the incoming text file as simply a stream of text, to which it applies certain regular expressions and associated commands. These commands help control the flow of the exectution path and a copy is made of the relevent parts of the stylesheet.

Chapter 2. The RegexXMLReader Stylesheet

2.1. Introduction

In order to appropriately parse an incoming text file, RegexXMLReader needs information to apply to the incoming text file. This information is an XML document that I call a Regular Expression Stylesheet, or simply "the Stylesheet" although there are major differences between this type of stylesheet and an XSLT stylesheet, which it it somewhat patterned after.

The stylesheet is made up of various directives which are essentially commands that are in the required namespace of RegexXMLReader, "http://regexxmlreader.sourceforge.net/1.0". Using these commands, the document is transformed into a series of SAX events thus turning the arbitrary text file into an XML document.

One concept that must be made clear is what I refer to in the documentation is *contextual text*. The contextual text is the entire incoming text file at the beginning of processing - it is just one large stream of data. As processing ensues, this stream of text usually changes to the more relevent parts; that is to say, the *context* changes for each inner child in the RegexStylesheet. This contextual text normally becomes less and less depending on the directives that are placed on its parent.

For example, consider the `replace` directive. This directive modifies the contextual text stream and that directive's children are then processed *on that modified version*. At the same level of `replace` *there is no change in the contextual text stream*. Rather, the change is only apparent for the directive's children:

```
<!--
    assume that the contextual text at this
    level is: "abcdefg hijklmno pqrstuvwxyz"
 -->
<re:replace regex="^." with="X" xmlns:re="http://regexxmlreader.sourceforge.net/1.0">
  <!--
       the contextual text is now:
       "Xbcdefg hijklmno pqrstuvwxyz"
   -->
  <re:for-each split=" ">
     <!--
          In this case the contextual text will change
          for each itteration through the split parts,
          namely:
                1) Xbcdefg
                2) hijklmno
                3) pqrstuvwxyz
           etc.
      -->
  </re:for-each>
  <!--
       The contextual text is what it was prior to
```

```
        the previous directive:
        "Xbcdefg hijklmno pqrstuvwxyz"
    -->
</re:replace>
<!--
     And now it is the same way it was at the
     beginning: "abcdefg hijklmno pqrstuvwxyz"
 -->
```

2.2. Directives

2.2.1. for-each

This directive splits up the textual context stream on certain criteria found in its one of two attributes and then applies its children upon the relevant text.

Table 2-1. for-each Attributes

regex	Each matching part of this elements contextual text is separated and the children of this node are processed using each individual part of the matching text.
split	The contextual text is broken up on this attribute's regular expression contents and the children of this node are applied on each individual part.

During each itteration against each matching part of the incoming text stream or split part of it, the contextual text stream is modified to only that part that is pertinent. For example, `<for-each regex=".">`. Beacause this particular regular expression (".") matches each and every character in the incoming contextual text stream, its children are processed upon each and every character and *that is they are aware of and have access to* (there is one exception to this, however).

2.2.2. replace

This directive modifies the current textual context according to the given criteria and applies the modified textual context upon its children. It is useful for such things as stripping out newlines or certain characters as well as normalizing text.

Table 2-2. replace Attributes

regex	The regular expression used to match the current contextual stream.
with	The character data that replaces the regular expression supplied with regex.
trim	Invoke the Java String command trim() upon the newly generated text prior to processing the children.

2.2.3. group

The group directive is used for processing either a regular expression that contains grouping commands (e.g. "([^,]+)") or an individual part of the split directive. The text of the group becomes the current contextual text.

A group's location determines which part of the match or split that is used unless the item attribute is used.

Table 2-3. group Attributes

item	Use the numerical value in this attribute as the determinent for which part of the matched grouped text or split part and not the order that the group node appears under its parent.

2.2.4. match-string

Sends the entire contextual text string to the output document.

2.2.5. text

Sends literal text to the output document.

Example 2-1. text Example

```
<output xmlns:re="http://regexxmlreader.sourceforge.net/1.0">
  <re:text>Text Sent to Output Document</re:text>
</output>
```

2.2.6. warning

Sends any textual-node children to the ErrorHandler as a warning.

Example 2-2. warning Example

```
<output xmlns:re="http://regexxmlreader.sourceforge.net/1.0">
  <title re:match="^[A-Z]+$">
    <re:match-string/>
  </title>
  <re:otherwise>
    <re:warning>
      <re:text>No match for: </re:text>
      <re:match-string/>
    </re:warning>
  </re:otherwise>
</output>
```

2.2.7. error

Sends textual-node children (not raw-text) to the ErrorHandler as a non-fatal error, an exception is thrown and processing stops (thus, this *is* a fatal error although it is not reported to the ErrorHandler as such.

Example 2-3. error Example

```
<output xmlns:re="http://regexxmlreader.sourceforge.net/1.0">
  <title re:match="^[A-Z]+$">
    <re:match-string/>
  </title>
  <re:otherwise>
    <re:error>
      <re:text>No match for: </re:text>
      <re:match-string/>
      <re:text>Please correct this and try again.</re:text>
    </re:error>
  </re:otherwise>
</output>
```

2.2.8. otherwise

During the course of processing, a count is kept for each match upon a contextual text stream for each group of siblings (or, in other words, each group of children of a node has their own count beginning with zero [0]). The children of this directive are not processed unless the count is zero (0), which is to say that this node should only be processed if there has not been a match prior to this element.

Example 2-4. otherwise Example

```
<output xmlns:re="http://regexxmlreader.sourceforge.net/1.0">
  <title re:match="^[A-Z]+$">
    <re:match-string/>
  </title>
  <content re:match="[A-Za-z]+$">
     <re:match-string/>
  </content>
  <re:otherwise>
    <unknown>
      <re:match-string/>
    </unknown>
  </re:otherwise>
</output>
```

2.2.9. attribute

This adds an attribute to the previous output element; it is not for adding an attribute to any stylsheet directive.

Table 2-4. attribute Attributes

name	The name of the attributue.

Example 2-5. attribute Example

```
<output xmlns:re="http://regexxmlreader.sourceforge.net/1.0">
  <part re:match="^([A-Z]+)(.*)$">
    <re:group>
      <re:attribute name="title">
        <re:match-string/>
      </re:attribute>
    </re:group>
    <re:group>
      <re:match-string/>
    </re:group>
```

```
  </part>
</output>
```

The above example places the first matching group in an attribute entitled "title" within the <part> element.

2.2.10. match

Causes the given regular expression in the supplied regex attribute to be applied upon the current contextual text stream and if there is a match then children of this directive are processed.

Table 2-5. match Attributes

regex	The regular expression that must match for the children of this directive to be processed.

Example 2-6. match Example

```
<output xmlns:re="http://regexxmlreader.sourceforge.net/1.0">
  <re:match re:regex="^([A-Z]+)(.*)$">
    <re:group>
      <title>
        <re:match-string/>
      </title>
    </re:group>
    <re:group>
      <description>
        <re:match-string/>
      </description>
    </re:group>
  <re:match>
</output>
```

2.2.11. split

The `split` causes the current contextual text to be split using the regular expression found in one of the two attributes `split` or `regex`. It is important to note that both of these attributes do the exact same thing; two different attributes are provided for convenience.

The `split` does not itterate at all over the resulting parts of the contextual text. Rather, it simply breaks up whatever that contextual text stream is based on the given criteria. `split` is usually soon followed by one or more `group` directives. For itteration, however, the `for-each` directive is provided.

Table 2-6. split Attributes

split	The regular expression for which the contextutal text is split upon.
regex	The regular expression for which the contextutal text is split upon.

Example 2-7. split Example

```
<output xmlns:re="http://regexxmlreader.sourceforge.net/1.0">
  <information>
    <re:split re:split=",">
      <re:group>
        <re:attribute name="title">
          <re:match-string/>
        </re:attribute>
      </re:group>
      <re:group>
        <part-number>
          <re:match-string/>
        </part-number>
      </re:group>
    </re:split>
  </information>
</output>
```

2.2.12. call-template

This directive causes the execution path to change to the node referenced by the ref-id attribute, which must be an `id` in the processor's namespace (http://regexxmlreader.sourceforge.net/1.0) although it does not need to be a directive.

Table 2-7. call-template Attributes

ref-id	A reference to a `id` attribute in some other node.

Example 2-8. call-template Example

```
<output xmlns:re="http://regexxmlreader.sourceforge.net/1.0">
```

```
  <page-number re:id="process-page-number" re:match="[0-9]+">
    <re:match-string />
  </page-number>
  <entry re:match="^([A-Z]+) +([0-9]+)$">
    <re:group>
      <title>
        <re:match-string />
      </title>
    </re:group>
    <re:group>
      <re:call-template ref-id="process-page-number"/>
    </re:group>
  </entry>
</output>
```

The above is operates *exactly* the same as:

```
<output xmlns:re="http://regexxmlreader.sourceforge.net/1.0">
  <page-number re:match="[0-9]+">
    <re:match-string />
  </page-number>
  <entry re:match="^([A-Z]+) +([0-9]+)$">
    <re:group>
      <title>
        <re:match-string />
      </title>
    </re:group>
    <re:group>
      <page-number re:match="[0-9]+">
        <re:match-string />
      </page-number>
    </re:group>
  </entry>
</output>
```

2.2.13. wrapper

This directive is simply a place holder and is meant to be used in conjunction with the `call-template` directive. `wrapper` is simply a place holder.

Example 2-9. wrapper Example

```
<output xmlns:re="http://regexxmlreader.sourceforge.net/1.0">
  <information>
    <title re:match="^[A-Z]+$" re:id="handle-title">
      <re:match-string/>
    </title>
    <re:wrapper id="deal-with-pages">
      <page re:match="^[0-9]+$">
        <re:match-string/>
      </page>
      <page-range re:match="^([0-9]+)-([0-9]+)$">
        <re:group>
          <start>
            <re:match-string/>
          </start>
        </re:group>
        <re:group>
          <end>
            <re:match-string/>
          </end>
        </re:group>
      </page-range>
      <!-- note that wrapper resets the match count -->
      <re:otherwise>
        <re:warning>
          <re:text>Pages did not match: </re:text>
          <re:match-string/>
        </re:warning>
      </re:otherwise>
    </re:wrapper>
    <re:match regex="^([A-Z]+) ([-0-9]+)$">
      <re:group>
        <re:call-template ref-id="handle-title"/>
      </re:group>
      <re:group>
        <re:call-template ref-id="deal-with-pages"/>
      </re:group>
    </re:match>
  </information>
</output>
```

2.2.14. pull-out-following-split

FIXME:

Table 2-8. pull-out-following-split Attributes

regex	FIXME:
inforce-order	FIXME:

2.3. Embedded Attributes

2.3.1. match

FIXME:

Chapter 3. Using RegexXMLReader from within an Application

Using the RegexXMLReader from within an application is not very much different than using other XMLReader implementions except that it is necessary to give the reader a DOM Node representing the regular expression stylesheet via the `setProperty` function using the `http://regexxmlreader.sourceforge.net/stylesheet-node` name, like this:

```
  String regexStylesheetName = "some-stylesheet.rxl";
  String incomingTextFile    = "some-file.txt";

  File inFile = new File(incomingTextFile);

  XMLReader reader = new RegexXMLReader();
  DOMParser domParser = new DomParser();
  domParser.parse(regexStylesheetName);
  Node stylesheetNode = (Node) domParser.getDocument();
  reader.setProperty("http://regexxmlreader.sourceforge.net/stylesheet-node", stylesheetNod
  reader.setContentHandler(this); // for example
  InputSource inputSource = new InputSource(inFile.toURL());
  reader.parse(inputSource);
```

Other than that, you should be able to use RegexXMLReader in any place that expects an implementation of the XMLReader interface.

Chapter 4. Using RegexXMLReader from the Command-Line

FIXME:

Chapter 5. Tutorial

5.1. A Simple CSV Example

As you may or may not know, CSV stands for "comma separated values" and it is a common method to group fields and records; the fields are separated by commas and the records are separated by new lines.

For this example, we shall use the following text file:

```
one,two,three,four,five
six,seven,eight,nine,ten
eleven,twelve,thirteen,fourteen,fifteen
sixteen,seventeen,eighteen,nineteen,twenty
```

In order to turn this into XML, we'll need to itterate over each newline and then split up each line based on the comma.

The itteration bit is rather simple so lets flesh out a Regular Expression Stylesheet using the root element of "csv-data" along with the iteration over the lines themselves and then prints out each line encapsulated in a "line" tag:

```
<?xml version="1.0"?>
<csv-data xmlns:re="http://regexxmlreader.sourceforge.net/1.0">
  <re:for-each regex="(?m)^.*$">
    <line>
      <re:match-string />
    </line>
  </re:for-each>
</csv-data>
```

One of the first things to note is the `(?m)` at the beginning of the `regex` in the `for-each` directive. This tells the regular-expression compiler within Java that we are to match multi-lines. Thus, with this regular expression, we are matching from the beginning of each line to the end of that very same line, and for each match found the children of this `for-each` directive are applied.

Let's take a look at the output by invoking the command line processor like: `java net.sourceforge.regexxmlreader.Process -in tutorial-example-01.txt -rxl tutorial-example-01.rxl`

```
<?xml version="1.0" encoding="us-ascii"?>
<csv-data>
  <line>one,two,three,four,five</line>
  <line>six,seven,eight,nine,ten</line>
  <line>eleven,twelve,thirteen,fourteen,fifteen</line>
```

```
  <line>sixteen,seventeen,eighteen,nineteen,twenty</line>
</csv-data>
```

Now we are close to what we want to do but not quite there; we also need to break up each line on the comma and there are a few ways that we can do this:

1. We can split the text up using the `split` directive and access each part with the `group` directive like so:

```
<re:split split=","/>
  <re:group>
    <!-- process the first matching part -->
    <first>
      <re:match-string/>
    </first>
  </re:group>
  <re:group>
    <!-- process the second matching part -->
    <second>
      <re:match-string/>
    </second>
  </re:group>
</re:split>
```

2. We can split the text up iterating over each part with the `for-each` directive:

```
<re:for-each split=",">
  <part>
    <re:match-string />
  </part>
</re:for-each>
```

3. We can use grouping within the regular expression of a `match` attribute within an external element:

```
<line re:match="^([^,]+),([^,]+),([^,]+),([^,]+),([^,]+)$">
  <re:group>
    <!-- process the first matching part -->
    <first>
      <re:match-string/>
    </first>
  </re:group>
  <re:group>
    <!-- process the second matching part -->
    <second>
      <re:match-string/>
    </second>
  </re:group>
</line>
```

4. We can do essentially the same thing using the `match` directive:

```
<re:match regex="^([^,]+),([^,]+),([^,]+),([^,]+),([^,]+)$">
  <line>
    <re:group>
      <!-- process the first matching part -->
      <first>
        <re:match-string/>
      </first>
    </re:group>
    <re:group>
      <!-- process the second matching part -->
      <second>
        <re:match-string/>
      </second>
    </re:group>
  </line>
</re:match>
```

In fact, there may be numerous other ways to do this but this is all we shall explore for right now.

Now for the final bit of this part of the tutorial, we shall do change the CSV file into XML using the `split` directive as well as place an attribute in the containing, out-going element. Here is the stylesheet:

```
<?xml version="1.0"?>
<csv-data xmlns:re="http://regexxmlreader.sourceforge.net/1.0">
  <re:for-each regex="(?m)^.*$">
    <line>
      <re:split regex=",">
<re:group>
  <element-one>
    <re:match-string />
  </element-one>
</re:group>
<re:group>
  <re:attribute name="element-two">
    <re:match-string />
  </re:attribute>
</re:group>
<re:group>
  <element-three>
    <re:match-string />
  </element-three>
</re:group>
      </re:split>
    </line>
  </re:for-each>
</csv-data>
```

And here is the result:

```
<?xml version="1.0" encoding="us-ascii"?>
<csv-data>
  <line element-two="two">
    <element-one>one</element-one>
    <element-three>three</element-three>
  </line>
  <line element-two="seven">
    <element-one>six</element-one>
    <element-three>eight</element-three>
  </line>
  <line element-two="twelve">
    <element-one>eleven</element-one>
    <element-three>thirteen</element-three>
  </line>
  <line element-two="seventeen">
    <element-one>sixteen</element-one>
    <element-three>eighteen</element-three>
  </line>
</csv-data>
```

Appendix A. GNU Free Documentation License

A.1. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

A.2. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject.

(Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this

License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

A.3. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

A.4. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

A.5. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document’s license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

A.6. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

A.7. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

A.8. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

A.9. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

A.10. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

A.11. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

A.12. ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.