XSLT-process minor mode

for version 1.2.1
February 2001

by Ovidiu Predescu

1 What is it and how to use it?

Have you ever developed XML applications using XSLT? If so you probably felt the need of viewing the result of applying the XSLT processor on the XML file using an XSLT sheet you have been working on right inside your (X)Emacs, without having to go to a terminal or to the Web browser. This minor mode allows you to do it!

The *XSLT-process* minor mode allows you, while you're inside a buffer for which this minor mode is enabled, to enter a few keystrokes that will invoke the XSLT processor of choice on the buffer. The result is displayed in another (X)Emacs buffer, that allows you to quickly view and inspect the results.

The XSLT file that's used to process the file should be specified inside the XML file using the XML processing instruction '`xml-stylesheet`', like this:

```
<?xml-stylesheet href="URL/to/XSLT/file" type="text/xsl"?>
```

You can use any URI understood by your Java system, e.g. file, HTTP or FTP, to refer to a stylesheet. The XSLT engine will download the stylesheet from the remote location and make use of it locally. With the TrAX interface (see Section 2.1 [Supported XSLT processors], page 2), the stylesheet is cached in the processor, so further invocations of the XSLT processor won't process the stylesheet again, unless the stylesheet is modified.

You can use URLs which are relative to the location of your XML file. For example if you have an XSLT file '`page-html.xsl`' in the same directory as the XML file you can simply have inside the XML file the following processing instruction:

```
<?xml-stylesheet href="page-html.xsl" type="text/xsl"?>
```

The *XSLT-process* mode was designed to work with any XSLT processor written in the Java language. Support for processors written in programming languages other than Java may require some work.

2 Installation

This mode depends on a few other packages:

- **an XSLT processor**: You definitely need an XSLT processor for this (X)Emacs mode to work! This package doesn't come with an XSLT processor integrated, so you need to download and install one (see Section 2.1 [Supported XSLT processors], page 2 for supported XSLT processors). Installing an XSLT processor is nothing else than placing the jar file in the Java CLASSPATH.

  You can either put the jar file in the system's CLASSPATH environment variable or use *XSLT-process*'s customization to setup the additional classpath. The supporting Java code expects the Java XSLT processor to be in the classpath, so make sure you define it one way or the other.
- **JDE**: *XSLT-process* was tested with JDE version 2.2.2, but it should work with newer versions as well. Please make sure you follow all the installation instructions on JDE's Web site (`http://sunsite.dk/jde/`)
- **Elib**: This package is a package JDE depends on, but I list it here to insure it's installed. If this package is missing, you'll get an error about the '`avltree`' package not found. This is a frequently asked question on the JDE's mailing list, so please make sure you don't ask it one more time.

If the above packages are not already installed on your system, you can install them in an '`emacs`' directory in your home directory. Then add the following lines in your '`.emacs`' configuration file:

```
(setq load-path
      (append (list "~/emacs/xslt-process-1.2.1/lisp"
    "~/emacs/jde/lisp"
    "~/emacs/semantic"
    "~/emacs/speedbar"
    ) load-path))

(autoload 'xslt-process-mode "xslt-process"
          "Run XSLT processor on buffer" t)
```

These lines add the needed packages in your emacs '`load-path`' variable, and make known the *XSLT-process* minor mode to your (X)Emacs.

After the *XSLT-process* mode has been made available to (X)Emacs, you can enable the minor mode on a per-buffer basis by running '`M-x xslt-process-mode`'. Or you can invoke '`xslt-process-mode`' from the mode's hook, see Section 2.2 [Setting up PSGML with XSLT-process], page 3 for an example on how you can do it.

2.1 Supported XSLT processors

In this version the *XSLT-process* minor mode supports the following XSLT processors:

- generic TrAX processor

  Any XSLT processor that implements the TrAX interface as defined in the JAXP 1.1 should work. The currently tested XSLT processors are Saxon 6.2 and Xalan 2.0.
- Saxon (`http://users.iclway.co.uk/mhkay/saxon/`)

  Both Saxon version 5.5.1, 6.0 and 6.1 are supported through the Saxon interface. If you have a version equal or newer than 6.2, the TrAX interface should be used instead. If you use the 6.0.1 version, beware that it prints some annoying messages to stdout which show up in the buffer of additional messages.
- Xalan 1.2 (`http://xml.apache.org/xalan/`)

  Xalan 1.2 is supported using the Xalan1 interface.
- Cocoon 1.8.x (`http://xml.apache.org/cocoon/`)

  Although Cocoon is not an XSLT processor, but rather an XML publishing framework, I added support for it as I'm using it quite extensively. This was in fact the primary reason I started *XSLT-process*: I got really tired of restarting Apache each time I was doing a change in either an XML file or an XSLT sheet.

2.2 Setting up PSGML with XSLT-process

The XSLT-process minor mode works really nice in conjuction with the PSGML major mode for SGML and XML editing.

To setup the XSLT-process minor mode to be automatically enabled whenever you edit an XML file under PSGML, add the below lines in your '`.emacs`'. I assume you have already added the configuration lines mentioned in see Chapter 2 [Installation], page 2.

```
(setq auto-mode-alist
      (append '(("\\.xml$" . xml-mode)) auto-mode-alist))

(add-hook 'xml-mode-hook 'xslt-process-mode)
```

If you're using a different major mode for editing XML documents, you can setup the mode's hook in a similar way as above to automatically enable the *XSLT-process* minor mode.

2.3 Key binding

To invoke the XSLT processor on a file, position the point inside the buffer and type '`C-c C-x C-v`'. You can customize the key binding by invoking '`M-x customize-group RET xslt-process`' and updating the key binding in the appropriate customization option.

The first time you invoke the XSLT processor on a buffer, the *XSLT-process* mode will start-up the Java Bean Shell (`http://www.beanshell.org/`), a Java helper program as a separate process running in the background. This operation may take some time, however further invocations of the processor are very fast as the JVM is started and all the classes are already loaded.

The XSLT processor will look into your XML file for an '`xml-stylesheet`' processing instruction specifying the XSLT file to be used to process the file.

2.4 Customization

There are several things you might want to customize. You can get to the customization page by typing '`M-x customize-group RET xslt-process`' or by choosing the following menu path in XEmacs: '`Options -> Emacs -> Programming -> Tools -> Xslt Process`'.

To choose the XSLT processor of your choice update the *Default Processor* option.

If you're using Cocoon as the processor, you should also specify the location of the properties file, otherwise you'll get an error at runtime.

2.5 Temporarily changing the XSLT processor

If you want to experiment what are the results of your stylesheets using different XSLT processors, going through the customization page and changing the processor can be quite an involved process.

You can specify the processor you want to be applied to your file right inside the file. Just add a *Local Variables* section at the end of your XML file and specify within it what should be the XSLT processor to be invoked using the '`processor`' variable. For example, by adding the following section at the end of your file, you specify *Saxon* to be used as the XSLT processor, no matter what is the global setting in the customization page:

```
<!--
Local Variables:
processor: Saxon
End:
-->
```

In this release, the acceptable values for '`processor`' are '`TrAX`', '`Saxon`', '`Xalan1`', and '`Cocoon1`'. By replacing the value of '`processor`', you can run any of the supported processors on your file.

Note however that in this release the TrAX processor which is chosen is the first processor that appears in the '`CLASSPATH`'. If you want to experiment with multiple TrAX processors, you will need to change the order of the processors in the '`CLASSPATH`' and restart the BSH process (just kill the '`*bsh*`' buffer, the next time you invoke the XSLT processor, *XSLT-process* will automatically restart BSH).

2.5.1 Additional parameters passed to Cocoon

It is possible to pass additional parameters to a Cocoon processor using '`user-agent`' local variable in an Emacs buffer:

```
<!--
Local Variables:
processor: Cocoon1
user-agent: UP.Browser
End:
-->
```

In this example the user agent of the requesting browser appears to Cocoon as being *UP.Browser*. If no user agent is specified, by default Cocoon will consider the requesting browser as being HTML capable, thus transforming the output to HTML.

3 Changes

These are the changes since the 1.2 release:

- Fixed problem in accessing stylesheets referred by *`file:`* URIs on Windows. Reported by Nicolas Kessler (`mailto:kessler@balcab.ch`).

This is the list of changes since the 1.1 release.

- Added support for the TrAX interface, thanks to Allan Erskine (`mailto:a.erskine@cs.ucl.ac.uk`). Currently Saxon 6.2 and Xalan2 have been tested. The TrAX interface caches the XSLT stylesheets in the processor driver in a compiled form, so the speed of the processing is increased.
- The mode is now running with GNU Emacs on Windows NT/2000, thanks to Allan Erskine (`mailto:a.erskine@cs.ucl.ac.uk`) for figuring out the issues.
- Changed again the keyboard binding to *`C-c C-x C-v`*, as *`C-M-x`* doesn't work on Windows systems.
- The documentation has been reorganized a little bit to be more logical.

This is the list of changes since the 1.0 release.

- The '`xslt-process-additional-classpath`' customization variable has been introduced. Setup this variable with any additional Java classpath components you want to be passed to the BeanShell when is first invoked. If you already started a BeanShell, you need to kill the corresponding buffer (named '`*bsh*`') and restart it by invoking *XSLT-process* on a buffer. (Suggestion from T. V. Raman (`mailto:tvraman@almaden.ibm.com`).)
- Allow for passing the user agent to the Cocoon processor so that multiple browser types can be simulated. This works with a patch I submitted against Cocoon 1.8-dev; it was incorporated and should be available in the 1.8.1 release. If you need the patch before this release, feel free to contact me, I'll gladly send it to you.
- The way the error messages are displayed has changed, now error messages messages encountered during the JVM startup process also go in the '`*xslt-output*`' buffer.
- The default keybinding has been changed to *`C-M-x`* instead of *`C-c x`*, to conform to the (X)Emacs keybinding standards.

4 Future enhancements

I have few ideas on how this mode could be improved, however they may or may not appear in the next version of this package.

- Add check to observe changes in the Java additional classpath and restart BSH if such changes occur while it's running.
- Add an additional buffer local variable to specify additional arguments to the XSLT processor. These arguments could then be read in the XSLT stylesheet using the '`xsl:param`' element.
- Introduce the ability to process an XML document with an XSLT stylesheet, without using the '`xml-stylesheet`' processing instruction inside the XML document. Define user customizable menu entries for the XSLT stylesheets.

  This would be really handy for people doing DocBook editing for example, as the DocBook stylesheets live somewhere on the disk and you don't want to put references to them in the XML documents.
- Provide the ability to view the result of processing in a browser, in addition to viewing them in (X)Emacs. This again would be handy for people doing HTML or WML authoring.

5 Feedback

I would appreciate any feedback on the *XSLT-process* mode; please send it to Ovidiu Predescu (`mailto:ovidiu@xemacs.org`).

The home page of the *XSLT-process* package is:

`http://www.geocities.com/SiliconValley/Monitor/7464/emacs/xslt-process/` .