PassiveTEX: an update

Sebastian Rahtz
Oxford University Computing Services
13 Banbury Road
Oxford OX2 6NN
sebastian.rahtz@oucs.ox.ac.uk

Abstract

This paper presents an overview of the development and status of 'PassiveTEX'. PassiveTEX is a library of TEX macros which can be used to process the XML which results from transformation of an XML document to the XSL Formatting Objects vocabulary.

PassiveTEX is a library of TEX macros which can be used to process an XML document which results from a transformation of an XML file to the W3C Formatting Objects (FO) XML vocabulary (see `http://www.w3.org/TR/xsl/`). The advantage of this is that it provides a rapid development environment for experimenting with XSL FO, using a reliable pre-existing formatter. Since we can now run with the pdfTEX variant of TEX, generating high-quality PDF files in a single operation, PassiveTEX shows how TEX can remain the formatter of choice for XML, while hiding the details of its operation from the user.

How does it work?

PassiveTEX builds on David Carlisle's XML parser written in TEX (XMLTEX), and was developed from my JadeTEX macros for processing DSSSL via Jade. XMLTEX is a highly complex set of TEX macros which parse XML files directly and apply TEX macros to the elements as defined in a configuration file. Since it is namespace-aware, it can have different configuration files for different XML applications. PassiveTEX is a large configuration file which maps XSL Formatting Objects onto a LATEX-based processing model in TEX, although very few of LATEX's front-end macros are visible. Another configuration file supplied with XMLTEX maps MathML onto TEX, allowing PassiveTEX to support XSL FO documents with embedded MathML.

How does XSL FO work? The language defines two primary objects: page masters, which define named styles of page layout; and page sequences, which reference a named page layout and contain a flow of text. Within that flow, text is assigned to one of five (rectangular) regions: the page body, areas at the top, bottom, left and right. We also have allowance for floating objects (at the top of the page), and footnotes (at the bottom), and the model covers writing in left/right and/or top/bottom modes. Within a region of text, we find one or more blocks, tables, lists and floats, while within a block (the equivalent to a TEX vertical box), we find inline sequences, characters, links, footnotes, and graphics. Associated with all these objects is an immense range of properties, divided into aural properties, borders, spacing and padding, breaking, colors, font properties (family, size, shape, weight, etc.), hyphenation, positioning, special table properties, and special list properties, although supporting absolutely all of them is not mandatory for a conforming processor.

It should be clear that the FO language should be able to describe the layout of most documents, by judicious combination of general purpose objects and their properties. The TEX user should note, however, that a FO document does not go as far as TEX in specifying exactly how pages will come out. It provides a set of constraints, but the exact line-breaking and page-breaking, for instance, can vary between implementations.

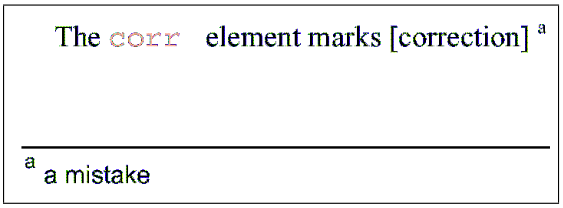
For an example of XSL FO, let us consider this piece of input XML written using the TEI (TEI Consortium, 2002) markup:

```
<p>The <gi>corr</gi> element marks
<corr sic="a mistake">correction</corr></p>
```

The `<gi>` markup says that the word 'corr' should be printed inside angle brackets, and the `<corr>` element should produce a footnote with the value of the 'sic' attribute. This text might be transformed into the following fragment of XSL FO:

```
<fo:block font-size="12pt"
 text-align="justify"
 text-indent="1em"
 space-before="0pt">
The <fo:inline
 color="green"
 font-family="Courier">corr
</fo:inline>
element marks [correction]
<fo:footnote>
<fo:inline font-size="8pt"
 vertical-align="super">a</fo:inline>
<fo:footnote-body>
<fo:block>
<fo:inline font-size="8pt"
 vertical-align="super">a</fo:inline>
<fo:inline
 font-family="Helvetica"
 font-size="10pt">a mistake</fo:inline>
</fo:block>
</fo:footnote-body>
</fo:footnote></fo:block>
```

and the result will look like this:



The corr element marks [correction] [a]

[a] a mistake

The PassiveTeX control file for XMLTeX which processes this XSL FO markup consists of a series of rules, one per element in the language. Each rule has three parts: a) any handling of attributes, b) what happens at the start of the element, and c) what happens at the end. This is demonstrated in the rule for floats:

```
\XMLelement{fo:float}
 {\XMLattributeX{float}{\FOfloat}{float}}
 {\ifx\FOfloat\att@none
     \begin{figure}[!htp]
   \else
     \begin{figure}
   \fi
   \FOlabel}
  {\end{figure}}
```

which does a fairly straightforward mapping of `<fo:float>` to LaTeX's figure environment. A slightly more complex example is this rule for `<fo:inline>`:

```
\XMLelement{fo:inline}
 {}
 {\xmlgrab}
 {
 \ifx\FOverticalalign\att@auto
    \let\FOverticalalign\FObaselineshift
 \fi
 \FOlabel
 \ifx\FOborderstyle\att@solid
   \ifx\FOborderwidth\att@thin
      \def\FOborderwidth{0.4pt}
   \fi
   \ifx\FOborderwidth\att@medium
      \def\FOborderwidth{0.8pt}
   \fi
   \ifx\FOborderwidth\att@thick
      \def\FOborderwidth{1.2pt}
   \fi
   \FOboxedsequence{#1}%
  \else
      \FO@inlinesequence{#1}%
  \fi}
```

which shows some of the problems in mapping from word values for properties like ‘medium’. The macros like `FO@inlinesequence` are defined in a large auxiliary file of helper macros for PassiveTeX.

Running PassiveTeX

Assuming you have created a file of XML using XSL FO vocabulary, you can use XMLTeX on a file called (say) `article.fo` in one of two ways:

1. Build an `xmltex` format file for pdfTeX with

   ```
   pdftex -ini "&pdflatex" pdfxmltex.ini
   ```

   and process your file with

   ```
   pdflatex "&pdfxmltex" article.fo
   ```

   Obviously you can create a command `pdfxmltex` to do this, which is just a script containing

   ```
   tex -fmt=pdfxmltex -progname=pdfxmltex
   ```

   Or,

2. Make a wrapper file called (say) `article.tex` along these lines:

   ```
   \def\xmlfile{article.fo}
   \input xmltex
   ```

   and run pdfTeX on it as normal with

   ```
   pdflatex article.tex
   ```

Do not worry, XMLTeX knows how to find the PassiveTeX macros as it needs them.

For reference, the PassiveTeX package consists of the following files:

- The core XMLTeX configuration files for XSL FO XML:

  ```
  fotex.xmt
  fotex.sty
  ```

- Support for direct formatting of TEI XML with XMLTeX

  ```
  tei.xmt
  teixml.sty
  ```

- Some support files, shared with JadeTEX:
  `unicode.sty`
  `ucharacters.sty`
  `mlnames.sty`
  `dummyels.sty`

Note that TEX has a limit on the length of line it can read, and some .fo files you generate may cause TEX to die with an message about increasing buf˜size. If you get that, edit your `texmf.cnf` file, increase the size of `buf_size` (mine is 200000), and remake any format files.

LATEX package dependencies This setup assumes you have a decent modern TEX setup. The TEX Live 7 CD-ROM is up to date (see `http://www.tug.org/texlive/`). Table 1 lists the packages loaded in a typical run of PassiveTEX, with their version numbers where known.

Extensions As explained above PassiveTEX effectively interprets MathML natively (elements must use the MathML namespace), and also supports a `bookmark` element in the `fotex` namespace, used to make PDF bookmarks. Usage is like this:

```
<fotex:bookmark
 xmlns:fotex="http://www.tug.org/fotex"
 fotex-bookmark-level="2"
 fotex-bookmark-label="ID">
 text of bookmark
</fotex:bookmark>
```

Notes on conformance to the XSL specification

The following general limitations apply to most of the PassiveTEX implementation of XSL FO:

1. The ‘px’ unit is not recognised.
2. Absolute dimensions always work, but proportional ones are often not recognized.
3. The functions allowed in attribute values are usually not recognized.
4. There is no error checking at all, and although all properties are recognized, do not assume that they do anything!

Most of the formatting objects are implemented more or less, except for the following:

1. `fo:bidi-override`
2. `fo:color-profile`
3. `fo:declarations`
4. `fo:initial-property-set`
5. `fo:instream-foreign-object`
6. `fo:multi-case`
7. `fo:multi-properties`
8. `fo:multi-property-set`
9. `fo:multi-switch`
10. `fo:multi-toggle`
11. `fo:region-end`
12. `fo:region-start`
13. `fo:table-footer`

The coverage of the myriad properties and valid values listed in the XSL FO specification is variable. All those that are straightforward to implement have been done; some are simply not relevant in TEX (e.g., the aural properties); some are just plain hard (repeatable column and rows in tables); others need help from (for example) Omega (bi-directional text). In some cases the TEX model just does not seem to fit—FO tables, for instance, work on the basis of cell properties, rather than TEX’s idea of thinking about columns.

Tables are (unsurprisingly) the weakest area of PassiveTEX. Where column widths are specified, it does a reasonable job, but it has as yet no system for deriving column widths from data, as required by XSL FO. This is because TEX’s table model has been abandoned in favour of the simple hbox and vbox constructs which can handle the endless variations on padding, borders and spacing.

Lastly, it should be noted the XSL FO inherits properties from Cascading Style Sheeets. CSS has a system of short-hands and composite values (“Times 12pt bold”) which is painful to parse in TEX, and thus are largely not supported in PassiveTEX.

Things for LATEX users to remember

- No use is made of LATEX high-level constructs. No sections, no lists, no cross-references, no bibliographies; on the other hand, some extensions in the `fotex:` namespace have been implemented (for example, to get Acrobat bookmarks).
- XSL FO’s underlying character set is Unicode; by default, entities are mapped to their Unicode position.
- All vertical and horizontal space is explicit in the specification.
- Page and line breaking is left to TEX: the rest is up to you.

References

[1] TEI Consortium, *Guidelines for Electronic Text Encoding and Interchange (TEI P4)*. Ed. C. M. Sperberg-McQueen and Lou Burnard. Chicago, Oxford: Text Encoding Initiative, 2002.

Table 1: LATEX packages needed by PassiveTEX

amsbsy.sty	1999/11/29 v1.2d
amsfonts.sty	1997/09/17 v2.2e
amsgen.sty	1999/11/30 v2.0
amsmath.sty	2000/03/29 v2.08 AMS math features
amsopn.sty	1999/12/14 v2.01 operator names
amssymb.sty	1996/11/03 v2.2b
amstext.sty	1999/11/15 v2.0
array.sty	1998/05/13 v2.3m Tabular extension package (FMi)
article.cls	1999/09/10 v1.4a Standard LaTeX document class
bm.sty	1999/07/05 v1.0g Bold Symbol Support (DPC/FMi)
color.sty	1999/02/16 v1.0i Standard LaTeX Color (DPC)
fontenc.sty	(version not available)
graphics.sty	1999/02/16 v1.0l Standard LaTeX Graphics (DPC,SPQR)
graphicx.sty	1999/02/16 v1.0f Enhanced LaTeX Graphics (DPC,SPQR)
hpdftex.def	2000/05/08 v6.70f Hyperref driver for pdfTeX
hyperref.sty	2000/05/08 v6.70f Hypertext links for LaTeX
ifthen.sty	1999/09/10 v1.1b Standard LaTeX ifthen package (DPC)
keyval.sty	1999/03/16 v1.13 key=value parser (DPC)
longtable.sty	1998/05/13 v4.09 Multi-page Table package (DPC)
multicol.sty	1999/10/21 v1.5w multicolumn formatting (FMi)
nameref.sty	2000/05/08 v2.18 Cross-referencing by name of section
ot1phv.fd	2000/01/12 PSNFSS-v8.1 scalable font definitions for OT1/phv.
pd1enc.def	2000/05/08 v6.70f Hyperref: PDFDocEncoding definition (HO)
pifont.sty	2000/01/12 PSNFSS-v8.1 Pi font support (SPQR)
rotating.sty	1997/09/26, v2.13 Rotation package
size10.clo	1999/09/10 v1.4a Standard LaTeX file (size option)
stmaryrd.sty	1994/03/03 St Mary's Road symbol package
t1enc.def	1999/12/08 v1.9x Standard LaTeX file
t1phv.fd	2000/01/12 PSNFSS-v8.1 scalable font definitions for T1/phv.
t1ptm.fd	2000/01/12 PSNFSS-v8.1 font definitions for T1/ptm.
t2acmr.fd	1999/01/07 v1.0 Computer Modern Cyrillic font definitions
t2aenc.def	1999/11/29 v1.0c Cyrillic encoding definition file
t3enc.def	(version not available)
textcomp.sty	1999/12/08 v1.9x Standard LaTeX package
times.sty	2000/01/12 PSNFSS-v8.1 Times font as default roman (SPQR)
tipa.sty	1996/06/10 TIPA version 1.0
trig.sty	1999/03/16 v1.09 sin cos tan (DPC)
ts1cmr.fd	1999/05/25 v2.5h Standard LaTeX font definitions
ts1enc.def	1998/06/12 v3.0d (jk/car/fm) Standard LaTeX file
ts1ptm.fd	2000/01/12 PSNFSS-v8.1 font definitions for TS1/ptm.
ulem.sty	1997/04/21
umsa.fd	1995/01/05 v2.2e AMS font definitions
umsb.fd	1995/01/05 v2.2e AMS font definitions
upsy.fd	2000/01/12 PSNFSS-v8.1 font definitions for U/psy.
upzd.fd	2000/01/12 PSNFSS-v8.1 font definitions for U/pzd.
url.sty	1999/03/28 ver 1.5x Verb mode for urls, etc.
Ustmry.fd	(version not available)
uwasy.fd	1999/05/13 v1.0iWasy-2 symbol font definitions
wasysym.sty	1999/05/13 v1.0i Wasy-2 symbol support package