

Working with TEI Stylesheets

Sebastian Rahtz

September 2014

Aims

- explain to you what the 'TEI Stylesheets' are
- show how they are used in the OxGarage document transformer
- demonstrate how to customize them to your own needs
- suggest how we can improve the stylesheets

Summary

The TEI Consortium loosely owns and manages a family of XSLT stylesheets which operate on TEI XML documents. They can be used:

- to implement an ODD processor, generating schemas and documentation from TEI sources (this is what the Consortium primarily needs)
- to do general-purpose formatting of TEI XML to 'human-readable' formats like HTML, ePub, LaTeX, XSL FO
- to convert between TEI XML and Microsoft Word, and between TEI and Open Office, format
- to convert between TEI and some other XML formats (TEI P4, EEBO TCP, NLM, Docbook)
- to generate JSON, RDF, BibTeX and other strange formats

There is no one right way to render TEI documents

What do the Stylesheets packages actually provide?

- A set of XSLT 2.0 transformations which read and write TEI XML
- A set of Ant scripts to package the transforms
- A set of Unix shell scripts, calling on Ant, to perform all conversions
- Use "trang" library to generate XSD from RELAXNG
- Use a TeX install to write PDF
- Use "profiles" as containers for customization

Do not assume that the conversions cover every feature of the input and output formats!

TEI stylesheet availability

The XSLT files are available:

- for download from Sourceforge
(<https://sourceforge.net/projects/tei/files/Stylesheets/>)
- within oXygen (in the TEI framework which can be updated separately from main oXygen)
- as Debian packages (for Linux users); see
<http://tei.oucs.ox.ac.uk/teideb/>
- in OxGarage (see later)
- on Github (<http://www.github.com/TEIC/Stylesheets>)

Usage examples

- 1 On a command line line, I write

```
docxtotei test11.docx test.xml
```

- 2 Using Ant, I write

```
ant -f docx/build-from.xml -DinputFile=`pwd`/Test/test11.docx -  
DoutputFile=test.xml -lib lib/saxon9he.jar
```

- 3 If I have a way of sending the file using REST, it would go to

```
http://oxgarage.oucs.ox.ac.uk:8080/ege-  
webservice/Conversions/docx%3Aapplication%3Avnd.openxmlformats-  
officedocument.wordprocessingml.document/TEI%3Atext%3Axml/
```

- 4 in oXygen, I choose the transformation scenario called "DOCX TEI"

Script options, for Linux people

The command-line scripts have a set of options, which you see by giving the command name followed by `--help`

```
$ teitohtml --help
TEI conversion: from tei to html

Usage: /usr/bin/teitohtml [options] inputfile [outputfile]

Options, binary switches:
--verbose          # be verbose
--debug           # be verbose, do not delete intermediate files
--apphome=/usr/share/xml/tei/stylesheet # where to find app directory
--
profiledir=/usr/share/xml/tei/stylesheet/profiles # where to find profile directory
--profile=default # which transformation profile to use
--oxygenlib=/usr/share/oxygen/lib # where is oxygenlib
--odd             # perform processing of ODD (if appropriate)
--localsource=DIR # where is local copy of source of TEI Guidelines
--summaryDoc     # only make summary, when doing ODD processing

Options, shown with defaults:
--saxonjar=/usr/share/saxon/saxon9he.jar # location of Saxon jar file
```

TEI Stylesheet family top-level layout

Some of the directories for output formats

docx	Converting to and from Word OOXML
epub	Converting to ePub
fo	Making XSL FO
latex	Making LaTeX
nlm	Converting from NLM
odds	Transforming TEI ODD specifications
odt	Converting to and from OpenOffice Writer
slides	Making slides (HTML and PDF)
tite	Converting from TEI Tite
html	Making HTML

Special directories

profiles	Customizations
common	Templates for any output format

Profile file naming convention

- in a directory hierarchy of the form **name/format/from.xml** or **name/format/to.xml** (indicating whether it is a conversion *from* or *to* the format)
- known formats are: bibtex, csv, docbook, docx, dtd, epub, epub3, fo, html, html5, json, latex, lite, markdown, nlm, odd, oddhtml, oddjson, odt, p4, pdf, rdf, relaxng, slides, tcp, txt, wordpress, xlsx.
- references to the 'master' conversions should be in the form (eg)

```
<xsl:import href="../../../epub/tei-to-epub.xml"/>
```

The common target: HTML, CSS and Javascript

Your final web page consists (probably) of

- the body of HTML created by running a transformation on your TEI XML
- standard navigation, search box, header, footer, menu items authored in HTML
- one or more CSS files controlling look and feel
- one or more Javascript scripts which do something clever (like providing sortable tables)

So

- XSLT transforms one document model to another
- CSS *decorates* the HTML document
- Javascript *changes* the HTML document dynamically

Ways of using HTML

Contrast:

```
<h1>6. Introduction to <b>R</b>
</h1>
```

with

```
<style type="text/css"> h1 {counter-increment: div1; }h1:before {content:
counter(div1) ". "; }span.package {font-weight:bold; }</style> ....
```

```
<h1>Introduction to <span class="package">R</span>
</h1>
```

Understanding the customization

There are six levels of interaction with the stylesheet family:

- 1 setting parameters
- 2 overriding templates provided for this purposed (listed in customization guide)
- 3 writing templates which implement the empty 'hooks' (listed in the customization guide)
- 4 adding new templates for elements not covered by the family
- 5 providing complete replacements for low-level templates

Documentation

You can find documentation on the web at
<http://www.tei-c.org/release/doc/tei-xsl/>

Documented XSL, produced by oXygen


Many parameters

There are dozens and dozens of parameters which affect the stylesheet output; you can set values for these by

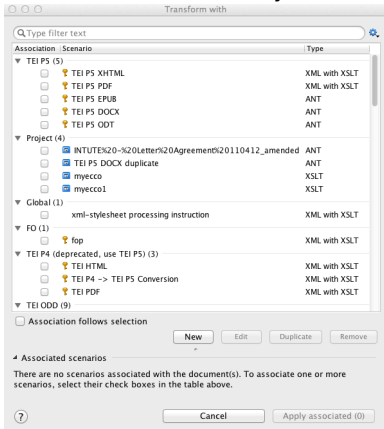
- specifying parameter names and values directly in oXygen
- setting them on a command line
- constructing a small local stylesheet which imports the public one, and adds overrides

Invoking an XSLT transform from oXygen



When you have loaded an XML file, look for the  symbol in the menu and press it.

The first time, it will ask you which transformation scenario to use:



Simple result

A TEI Project

Punch, or the London Charivari, Vol. 147, July 1, 1914


Table of contents

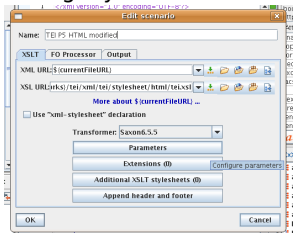
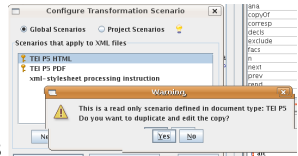
1. [PROGRESS.](#)
2. [THE ENCHANTED CASTLE.](#)
3. [Correspondence.](#)
6. [EGYPT IN VENICE,"La Légende de Joseph2](#)
8. [ENIGMA.](#)
10. [A SCANDALMONGRIAN ROMANCE.\(By Francis Scribble.\)](#)
12. [CHARTIVARIA.](#)
13. [THE COLLECTORS.](#)
14. [KINDNESS TO SUBJECTS.](#)
16. [THE WALKERS.](#)
17. [King Peter of Servia.](#)
19. [ESSENCE OF PARLIAMENT.\(Extracted from the Diary of Toby, M.P.\)](#)
22. [DISGUISE.](#)

Find: feed Previous Next Highlight all Match case

Configuring the scenario in oXygen

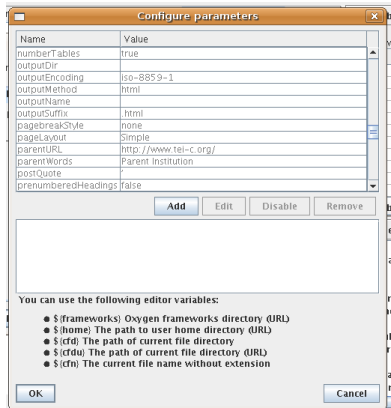


Look for the  symbol. This produces asking if you want to change the setup. Choose yes, and you see



Changing parameters in oXygen

Now you can supply values for parameters:



Areas of customization (HTML)

- Standard page features
- Layout
- Headings
- Numbering
- Output
- Table of contents generation
- Internationalization
- CSS
- Tables
- Figures and graphics
- Inline style

Remember that in HTML a lot of styling can be done with CSS and JavaScript

Change pageLayout

Configure parameters

Name	Value
numberTables	true
outputDir	
outputEncoding	iso-8859-1
outputMethod	html
outputName	
outputSuffix	.html
pagebreakStyle	none
pageLayout	Simple
parentURL	http://www.tei-c.org/
parentWords	Parent Institution
postQuote	'
prenumberedHeadings	false

Edit

Name: pageLayout
Value: CSS

OK Cancel

Add Edit Disable Remove

generated pages.
The choice is between
SimpleA linear presentation is createdCSSThe page is created as a series of
nested
<div>s which can be arranged using CSS into a multicolumn layoutTableThe
page is created as an HTML table
Default value: Simple

2 column display

Slip links

Home

Parent Institution

1. PROGRESS.
2. THE ENCHANTED CASTLE.
3. Correspondence.
6. EGYPT IN VENICE."La Légende de Joseph2
8. ENIGMA.
10. A SCANDALMONGRIAN ROMANCE.(By Francis Scribble.)
12. CHARIVARIA.
13. THE COLLECTORS.
14. KINDNESS TO SUBJECTS.
16. THE WALKERS.



Changing things around a bit

Punch, or the London Charivari, Vol. 147, July 1, 1914

[Skip links](#)

[Punch central](#)

[The Land of Charivari](#)

PROGRESS.

**THE ENCHANTED
CASTLE.**

Correspondence.

EGYPT IN VENICE."La
Légende de Joseph"

ENIGMA.

**A SCANDALMONGRIAN
ROMANCE.**(By
Francis Scribble.)

CHARIVARIA.

THE COLLECTORS.

**KINDNESS TO
SUBJECTS.**

THE WALKERS.

King Peter of Servia.

ESSENCE OF



Using the a wrapper stylesheet

The simplest example of making a wrapper for the HTML stylesheets is:

```
<xsl:stylesheet xpath-default-namespace="http://www.tei-c.org/ns/1.0"
  version="2.0">
  <xsl:include href="http://www.tei-
c.org/release/xml/tei/stylesheet/html/html.xsl"/>
</xsl:stylesheet>
```


Using the a wrapper stylesheet (2)

Now you can build on it:

```
<xsl:stylesheet xpath-default-namespace="http://www.tei-c.org/ns/1.0"
  version="2.0">
  <xsl:include href="http://www.tei-
c.org/release/xml/tei/stylesheet/html/html.xsl"/>
  <xsl:param name="logoFile">../../logo.png</xsl:param>
  <xsl:param name="logoWidth">60</xsl:param>
  <xsl:param name="logoHeight">60</xsl:param>
  <xsl:param name="cssFile">myTEI.css</xsl:param>
  <xsl:param name="pageLayout">Complex</xsl:param>
  <xsl:param name="outputMethod">xml</xsl:param>
  <xsl:param name="parentWords">The Punch
    Project</xsl:param>
  <xsl:param name="institution">The University of
    Punch</xsl:param>
</xsl:stylesheet>
```

Using the a wrapper stylesheet (3)

And start to add your own templates:

```
<xsl:stylesheet xpath-default-namespace="http://www.tei-c.org/ns/1.0"
  version="2.0">
  <xsl:include href="http://www.tei-
c.org/release/xml/tei/stylesheet/html/html.xsl"/>
  <xsl:param name="logoFile">../../logo.png</xsl:param>
  <xsl:param name="logoWidth">60</xsl:param>
  <xsl:param name="logoHeight">60</xsl:param>
  <xsl:param name="cssFile">myTEI.css</xsl:param>
  <xsl:param name="pageLayout">Complex</xsl:param>
  <xsl:param name="outputMethod">xml</xsl:param>
  <xsl:param name="parentWords">The Punch
  Project</xsl:param>
  <xsl:param name="parentURL">http://tei.oucs.ox.ac.uk/Punch/</xsl:param>
  <xsl:param name="institution">The
  University of Punch</xsl:param>
  <xsl:template match="hi[@rend='upside-down']">
    <span class="upside-down">
      <xsl:apply-templates/>
    </span>
  </xsl:template>
</xsl:stylesheet>
```