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
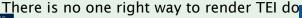- suggest how we can improve the stylesheets

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 direc
  --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
```

What is OxGarage?

OxGarage is a web app
(http://oxgarage.oucs.ox.ac.uk:8080/ege-webclient) which
provides document transformations, featuring

- Web and REST interface
- Chained XSLT conversions
- Uses headless OpenOffice for binary conversions
- Uses TEI XML as pivot format
- Supports Stylesheets "profiles" for variations
- Open source across the board

History and dependencies

- Built at the University of Poznan for EU-funded ENRICH project's EGE for converting manuscript descriptions, using conversions and libraries from Oxford
- Now developed and maintained as a fork by the University of Oxford
- Java servlet, running under Tomcat in current instances
- Almost all work done as XSLT transforms using Saxon library
- Uses OpenOffice to read/write .doc, .xls, .ppt files etc

`http://www.github.com/sebastianrahtz/oxgarage`

Matrix of OxGarage conversions (1)

Presentation documents

	Microsoft PowerPoint Presentation (.ppt)	Microsoft PowerPoint Presentation (.pptx)	Open Office Presentation (.odp)	OpenOffice.org 1.0 Presentation (.sxi)
Inputs:				
Outputs:				
Macromedia Flash (.swf)	✓	✓	✓	✓
Open Office Presentation (.odp)	✓	✓	✓	
OpenOffice.org 1.0 Presentation (.sxi)	✓	✓	✓	
PDF Document	✓	✓	✓	✓
Microsoft PowerPoint Presentation (.ppt)	✓	✓	✓	

Spreadsheet documents

	Comma-Separated Values (.csv)	Microsoft Excel Document (.xls)	Microsoft Excel Document (.xlsx)	Open Office Spreadsheet (.ods)	OpenOffice.org 1.0 Spreadsheet (.sxc)	Tab-Separated Values (.tsv)
Inputs:						
Outputs:						
TEI P5 XML Document	✓	✓	✓	✓	✓	✓
ePub Document	✓	✓	✓	✓	✓	
LaTeX Document	✓	✓	✓	✓	✓	
Microsoft Excel Document (.xls)	✓	✓	✓	✓	✓	
xHTML Document	✓	✓	✓	✓	✓	
Microsoft Word Document (.doc)	✓	✓	✓	✓	✓	
Microsoft Word Document (.docx)	✓	✓	✓	✓	✓	
National Library of Medicine (NLM) DTD 3.0	✓	✓	✓	✓	✓	
Open Office Spreadsheet (.ods)	✓	✓	✓		✓	
Open Office Text Document (.odt)	✓	✓	✓	✓	✓	
OpenOffice.org 1.0 Spreadsheet (.sxc)	✓	✓	✓	✓		
OpenOffice.org 1.0 Text Document (.sxw)	✓	✓	✓	✓	✓	
PDF Document	✓	✓	✓	✓	✓	
Plain Text (.txt)	✓	✓	✓	✓	✓	
Rich Text Format (.rtf)	✓	✓	✓	✓	✓	
Tab-Separated Values (.tsv)	✓	✓	✓	✓	✓	✓
XSL-FO Document	✓	✓	✓	✓	✓	
Comma-Separated Values (.csv)	✓	✓	✓	✓	✓	✓

Text documents

Matrix of OxGarage conversions (2)

	Inputs:												
	Compiled TEI ODD Document	DocBook Document	Microsoft Word Document (.doc)	Microsoft Word Document (.docx)	ODD Document	Open Office Text Document (.odt)	OpenOffice 1.0 Text Document (.sxw)	Plain Text (.txt)	Rich Text Format (.rtf)	TEI P4 XML Document	TEI P5 XML Document	WordPerfect Document (.wpd)	xHTML Document
Outputs:													
ODD documentation as TEI Lite	✓				✓								
Comma-Separated Values (.csv)		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DTD created from ODD	✓				✓								
ePub Document	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LaTeX Document	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Microsoft Excel Document (.xls)	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
xHTML Document	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
Microsoft Word Document (.doc)	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
Microsoft Word Document (.docx)	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓		✓
National Library of Medicine (NLM) DTD 3.0	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓		✓
ODD documentation as HTML	✓				✓								
Open Office Spreadsheet (.ods)	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓		✓
Open Office Text Document (.odt)	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓		✓
OpenOffice 1.0 Spreadsheet (.sxc)	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓		✓
OpenOffice 1.0 Text Document (.sxw)	✓	✓	✓	✓	✓				✓	✓	✓		✓
PDF Document	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓		✓
Plain Text (.txt)	✓				✓								
RELAXNG schema created from ODD	✓				✓								
Rich Text Format (.rtf)	✓	✓	✓	✓	✓	✓	✓		✓		✓	✓	✓
Tab-Separated Values (.tsv)	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
XSL-FO Document	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓		✓
TEI P5 XML Document	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓		✓
Compiled ODD Document					✓								

OxGarage: constructing a path

`http://oxgarage.oucs.ox.ac.uk:8080/ege-`
`webservice/Conversions/`*format*`/`*format*`/?`*properties*

'formats' are a name followed by a mime type. For example:

format	code
ePub	application%3Aepub+zip
XSL FO	application%3Axslfo+xml
LaTeX	application%3Ax-latex
TEI LITE	text%3Axml
ODD HTML	application%3Axhtml+xml
ODD Json	application%3Ajson
ODT	application%3Avnd.oasis.opendocument.text
RDF	application%3Ardf+xml
RELAX NG	application%3Axml-relaxng
TEI ODD	ODD%3Atext%3Axml/
TEI P5	TEI%3Atext%3Axml/
Word	docx%3Aapplication%3Avnd.openxmlformats-officedocument.wordprocessingml.document/

OxGarage web service example (1)

Process ODD to compiled ODD, then to TEI Lite, then to DOCX

```
curl -s  -F upload=@test.odd  -o test.docx
http://oxgarage.oucs.ox.ac.uk:8080/
ege-webservice/Conversions/
ODD%3Atext%3Axml/
ODDC%3Atext%3Axml/
TEI%3Atext%3Axml/
docx%3Aapplication%3Avnd.openxmlformats-
officedocument.wordprocessingml.document/
```

OxGarage web service example (2)

ODD to HTML, in French

```
curl -s  -F upload=@test.odd  -o test.html
http://oxgarage.oucs.ox.ac.uk:8080/ege-webservice/Conversions/
ODD%3Atext%3Axml/
ODDC%3Atext%3Axml/
oddhtml%3Aapplication%3Axhtml%2Bxml/
?properties=<conversions><conversion%20index='1'>
<property%20id='oxgarage.lang'>fr</property></conversion></conversions>
```

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.xsl or name/format/to.xsl
  (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.xsl"/>
```

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/`

eg

5.1.1. Variables

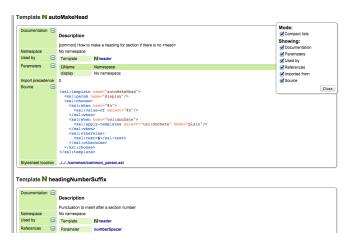| Type	Name	Description	Default
	department	Name of department within institution [string]	
	homeLabel	Name of link to home page of application [string]	Home
	homeURL	Project url [anyURI]	/
	homeWords	Project name [string]	Home
	institution	Institution or project name [string]	(unknown project)
	parentURL	Institution link [anyURI]	
	parentWords	Name of overall institution [string]	Parent Institution
	searchURL	Link to search application [anyURI]	
	feedbackURL	Link for feedback [anyURI]	
html	*alignNavigationPanel*	How to align the navigation panel at the bottom of the page [string]	right
html	*bottomNavigationPanel*	Display navigation panel at bottom of pages [boolean]	true
html	*htmlTitlePrefix*	Fixed string to insert before normal page title in HTML meta <title> element [string]	
html	*linkPanel*	Make a panel with next page/previous page links. [boolean]	true

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
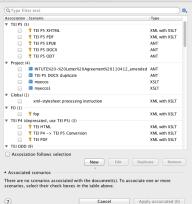
Invoking an XSLT transform from oXygen

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

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

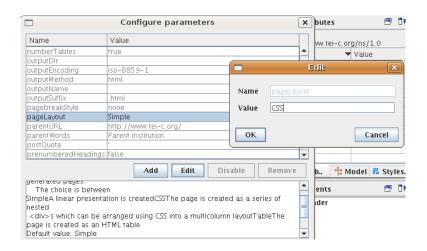Remember that in HTML a lot of styling can be done with CSS and JavaScript

Change pageLayout

2 column display

Changing things around a bit

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

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
<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='upsidedown']">
  <span class="upsidedown">
   <xsl:apply-templates/>
  </span>
 </xsl:template>
</xsl:stylesheet>
```