

Using TEI XSLT stylesheets

Sebastian Rahtz

February 2006

TEI Stylesheets

Using TEI
XSLT
stylesheets

Sebastian
Rahtz

A library of stylesheets for transforming TEI documents to

- HTML
- XHTML
- XSL Formatting Objects
- LaTeX

at <http://www.tei-c.org/Stylesheets>, with a form-filling interface for the HTML at <http://www.tei-c.org/tei-bin/stylebear> (also on your CD)

TEI Stylesheets (example)

Using TEI
XSLT
stylesheets

Sebastian
Rahtz

An example of an importing stylesheet:

```
<xsl:stylesheet version="1.0">  
  <xsl:import href="/usr/share/xml/tei/stylesheet/html/tei.xsl"/>  
  <xsl:param name="splitLevel">1</xsl:param>  
  <xsl:param name="numberHeadings"/>  
  <xsl:param name="topNavigationPanel">true</xsl:param>  
  <xsl:param name="bottomNavigationPanel">true</xsl:param>  
</xsl:stylesheet>
```

A more complex example (the TEI site)

Using TEI
XSLT
stylesheets

Sebastian
Rahtz

```
<xsl:param name="oddmode">html</xsl:param>
<xsl:variable name="top" select="/" />
<xsl:param name="STDOUT">>true</xsl:param>
<xsl:param name="alignNavigationPanel">left</xsl:param>
<xsl:param name="authorWord"/>
<xsl:param name="autoToc"/>
<xsl:param name="bottomNavigationPanel">>true</xsl:param>
<xsl:param name="cssFile">Stylesheets/tei.css</xsl:param>
<xsl:param name="feedbackURL">http://www.tei-c.org/Consortium/
<xsl:param name="feedbackWords">Contact</xsl:param>
<xsl:param name="homeURL">http://www.tei-c.org/</xsl:param>
<xsl:param name="homeWords">TEI Home</xsl:param>
<xsl:param name="institution">Text Encoding
Initiative</xsl:param>
<xsl:param name="leftLinks">>true</xsl:param>
<xsl:param name="searchURL">http://search.ox.ac.uk/web/related/
<xsl:param name="searchWords">Search this site</xsl:param>
<xsl:param name="showTitleAuthor">1</xsl:param>
<xsl:param name="subTocDepth">-1</xsl:param>
<xsl:param name="topNavigationPanel"/>
<xsl:param name="numberHeadings">>true</xsl:param>
<xsl:template name="copyrightStatement">Copyright TEI
Consortium, 2004</xsl:template>
```

Result on TEI web site

Using TEI
XSLT
stylesheets

Sebastian
Rahtz



The Text Encoding Initiative

TEI: Yesterday's information tomorrow

[Home](#) [Guidelines](#) [Projects](#) [Tutorials](#) [Software](#) [History](#) [FAQs](#) [P5](#) [Consortium](#) [Activities](#) [SIGs](#) [Wiki](#) [Join in/Contact](#) [Members area](#)



The Text Encoding Initiative (TEI) Guidelines are an international and interdisciplinary standard that facilitates libraries, museums, publishers, and individual scholars represent a variety of literary and linguistic texts for online research, teaching, and preservation.

The TEI standard is maintained by a [Consortium](#) of leading Institutions and Projects worldwide. Information on projects which use the TEI, who is a member, and [how to join](#), can all be found via the links above. Consortium members contribute to its financial stability and elect

members to its Council and Board.

The [Guidelines](#) are the chief deliverable of the TEI Consortium, along with a range of [tutorials](#), [case studies](#), [presentations](#), and [software](#) developed for or adapted to the TEI. The latest release of the Guidelines under development is [P5](#).

The TEI was originally sponsored by the Association of Computers in the Humanities (ACH), the Association for Computational Linguistics (ACL), and the Association of Literary and Linguistic Computing (ALLC). Major support has been received from the U.S. National Endowment for the Humanities (NEH), the European Community, the Mellon Foundation, and the Social Science and Humanities Research Council of Canada.

Want to become active in the TEI Community? Join a [Special Interest Group](#), sign up for the [mailing list](#), and come to our annual meetings.

The [fifth Annual Members Meeting](#) will be held 28-29 October 2005, at the Bulgarian Academy of Sciences, Sofia, Bulgaria and is open to members and non-members alike. ([see announcement](#) for program and registration details.)

Skeleton of result

Using TEI
XSLT
stylesheets

Sebastian
Rahtz

The image shows a screenshot of a TEI document skeleton, likely generated from XSLT. The document is titled "TEI U5: Encoding for Interchange: an introduction to the TEI". The main content area is divided into two columns: a left column for a table of contents (TOC) and a right column for the main text. The TOC is titled "For Printing" and lists the document's sections. The main text area is titled "11. Names, Dates, Numbers and Abbreviations" and contains a paragraph about the TEI scheme's features. The document is annotated with several labels and arrows:

- #hdr**: Points to the header area containing the title "h1.maintitle".
- #hdr2**: Points to the second header area, which is empty.
- #hdr3**: Points to the third header area, which is empty.
- #lh-col**: Points to the left column of the TOC, which contains the following items:
 - Prefatory note
 - 1. Introduction
 - 2. A Short Example
 - 3. The structure of a TEI document
 - 4. Encoding the Body
 - 5. Page and Line Numbers
 - 6. Marking Highlighted Phrases
 - 7. Cross References and Links
 - 8. Editorial Interventions
 - 9. Omissions, Deletions, and Additions
 - 10. Lists
 - 11. Bibliographic
- #rh-col**: Points to the right column of the TOC, which contains the following items:
 - 11.1. Names and Referring Strings
 - 11.2. Dates and Times
 - 11.3. Numbers
 - 11.4. Addresses
 - 11.5. Lists
 - 11.6. Bibliographic

The main text area contains the following content:

11. Names, Dates, Numbers and Abbreviations

The TEI scheme defines elements for a large number of 'data-like' features which may appear almost anywhere within almost any kind of text. These features may be of particular interest in a range of disciplines; they all relate to objects external to the text itself, such as the names of persons and places, numbers and dates. They also pose particular problems for many natural language processing (NLP) applications because of the variety of ways in which they may be presented within a text. The elements described here, by making such features explicit, reduce the complexity of processing texts containing them.

11.1. Names and Referring Strings

A *referring string* is a phrase which refers to some person, place, object, etc. Two elements are provided to mark such strings:

<rs> contains a general purpose name or referring string. Attributes include:

- type** indicates more specifically the object referred to by the referencing string. Values might include person, place, ship, element, etc.

<name> contains a proper noun or noun phrase. Attributes include:

- type** indicates the type of the object which is being named by the phrase.

The **type** attribute is used to distinguish amongst (for example) names of persons, places and organizations, where this is possible:

Example:

```
<q>My dear <rs type="person">Mr. Bennet</rs>, </q>  
said his lady to him one day, <q>have you heard  
that <rs type="place">Matherfield Park</rs> is 'let  
at last'</q>
```

Skeleton of result (2)

```
<div id="hdr">
  <xsl:call-template name="hdr"/>
</div>
```

```
<div id="hdr2"> <xsl:call-template name="hdr2"/> </div>
```

```
<div id="hdr3"> <xsl:call-template name="hdr3"/> </div>
```

```
<div id="lh-col">
<div id="lh-col-top">
<xsl:call-template
name="lh-col-top"/>
</div>
<div id="lh-col-bottom">
<xsl:call-template
name="lh-col-bottom">
<xsl:with-param
name="currentID"
select="$currentID"/>
</xsl:call-template>
</div>
</div>
```

```
<div id="rh-col">
<div id="rh-col-top">
<xsl:call-template name="rh-col-top"/>
</div>
<div id="rh-col-bottom">
<xsl:call-template name="rh-col-bottom">
<xsl:with-param name="currentID" select="$currentID"/>
</xsl:call-template>
</div>
</div>
<div id="lh-col">
<div id="lh-col-top">
<xsl:call-template name="lh-col-top"/>
</div>
<div id="lh-col-bottom">
<xsl:call-template name="lh-col-bottom">
<xsl:with-param name="currentID" select="$currentID"/>
</xsl:call-template>
</div>
</div>
```

What is XSLT good for?

Yes, it is useful for making web pages from TEI texts, but it is also a good tool for

- Selecting subsets of our texts for further processing
- Summarizing aspects of our texts
- Checking our text in ways that DTDs and schemas cannot
- Converting text into formats other than HTML or XSL FO

we can solve many of the problems with a small range of techniques.

Finding any occurrence of an element

Very often, we will sit on the root element and process all the occurrences of a specific element by using the descendant axis:

```
<xsl:template match="/">
  <html>
    <body>
      Pages:
      <xsl:value-of select="count(descendant::tei:pb)"/>
    </body>
  </html>
</xsl:template>
```

here we use the count function. We continue to generate an HTML document to provide a convenient reporting format.

Converting to other formats

We can write a template to list the size of paragraphs:

```
<xsl:template match="tei:p">
  <xsl:variable name="contents">
    <xsl:apply-templates select="."/text()"/>
  </xsl:variable>
  <xsl:number level="any"/>:
  <xsl:value-of select="string-length($contents)"/>
</xsl:template>
```

Adding `<xsl:output method="text"/>` will produce pure text output which could be loaded into a spreadsheet or database.

XSLT extensions

Although we will not be covering them here, most XSLT processors support various extensions, which will be formalized in version 2.0:

- The ability to create *multiple* output files from one input
- The ability to ‘escape’ to another language (eg Java) for special purposes
- The ability to turn results into input trees for further processing