The Short-and-Sweet TEI Handout*

Francesca Giannetti

April 5, 2022

XML

XML stands for eXtensible Markup Language and as the word markup implies, it is a tool used to describe data. HTML, which you may be more familiar with, shares many similarities with XML, most importantly the use of tags `< >`. The "data" in HTML consists of instructions for browsers; in XML, on the other hand, there is no predefined use or vocabulary for the tags (hence the "eXtensible" in XML).[1] For example, if you owned a black cat named Spenser, you could "express" it in XML in this way:

```
<cat type="mine" name="Spenser">
    <color>black</color>
    <breed>Domestic short hair</breed>
</cat>
```

In a sense, all texts can be said to contain data of a certain kind. Literature and criticism are no exception to this rule. XML helps you name and organize that data. With XML, the possibilities are legion: we could, for example, name the kinds of content we find (`<metaphor>`, `<character>`, etc.), describe the physical attributes of a book (`<paper>`, `<ink>`, etc.), how a text is laid out (`<column>`, `<page_break>`, etc.) or the logical units of a text (`<line>`, `<paragraph>`, etc).

Because there are so many possibilities, scholars and scientists all over the world have agreed to use standards in their fields. In digital humanities, the most important standard set of predetermined tags, or tag-set, is the one provided by the Text Encoding Initiative (TEI). In this workshop, we will use a subset of that standard called TEI-lite to introduce you to the practice of *tagging*.[2]

A basic TEI file includes a text (the linguistic content) and metadata (information about the text). The first three tags you will learn already express the basic structure of a TEI file. The topmost tag includes all other tags and is named `<TEI>`. The tag that includes the information about the text is called the `<teiHeader>` and always precedes the tag which includes the text, appropriately named `<text>`.

The overall structure of the TEI file then looks something like this:

*With thanks to Alex Gil, Digital Scholarship Librarian at Columbia University Libraries, whose handout I've adapted.

[1] For a longer introduction to XML, see https://www.w3schools.com/xml/.

[2] To deepen your knowledge of TEI and the tags for prose, you can explore TEI by Example. Googling 'tei p5' + name-of-element-or-attribute also works!

```
<TEI>
    <teiHeader>
        [information about the text]
    </teiHeader>
    <text>
        [the text itself]
    </text>
</TEI>
```

teiHeader

In Visual Studio Code, go to **File > Open Folder**. Open the unzipped folder called "bleak-house-exercise". In the Explorer pane to the left, click on `template.xml` to open it. You will notice that some tags for the `<teiHeader>` are already there.

fileDesc

This is where we include information about the text we are encoding. There are three large categories we will use within the `fileDesc` element:

- `<titleStmt>`: The title statement refers to the digital work. For the most part, the digital work preserves most of the information from the print text, but adds information about responsibility for the production of the digital version.
- **Important**. Within the `<titleStmt>` you will find the `<respStmt>`, or responsibility statement. Assign yourself a unique identifier using the @xml:id attribute like so:

```
<respStmt>
  <resp>encoded by</resp> <name xml:id="fg">Francesca Giannetti</name>
</respStmt>
```

- `<publicationStmt>`: The publication statement refers to the channels through which the digital work will be distributed and stewarded.
- `<sourceDesc>`: The source description finally refers to the original printed text that you are encoding.

text

Within the `text` element, we may include tags for the following categories:

- Front matter, body, and back matter
- Sections or divs
- Paragraphs
- Page breaks
- Emphasis
- Graphical elements
- Footnotes

Front, Body and Back

Many texts are divided into front matter, body, and back matter. The oXygen template assumes only the presence of a `body` tag within `text`, but `front` or `back` may be added. Here is the general structure of a `<text>` element:

```
<text>
    <front></front>
    <body></body>
    <back></back>
</text>
```

The `front` may include a title page, a table of contents, and other introductory material. In many books, the front matter is easily identifiable because the pages are marked with lowercase roman numerals.

The `body` is where we find the text proper.

As the name suggests, the `back` comes at the end of the book when the main content can be said to be over. Usual back matter includes an index and a colophon.

Since there is a lot of room for variety, marking sections within these three elements usually takes advantage of the `<div>` element explained below.

titlePage

The title page of the book is very similar to the `teiHeader`. In print technology, this is where we would find our basic metadata, author, title and publisher. Note: the author is declared with the tag `<byline>`. E.g.

```
<titlePage>
  <docTitle>
    <titlePart type="main">A Christmas carol.
      <lb/>
    <hi rend="roman">in prose.</hi>
      <lb/>
    <hi rend="roman">being</hi>
      <lb/>
    <hi rend="gothic">A Ghost Story of Christmas</hi>
    </titlePart>
  </docTitle>
  <byline>by <lb/>
    <docAuthor>Charles Dickens.</docAuthor>
  </byline>
  <byline>With illustrations by John Leech</byline>
  <docImprint>London: Chapman and Hall, Ltd. 1893.</docImprint>
</titlePage>
```

Sections

Sections are organized differently in different texts. This is why the generic tag `<div>` is used to divide a text into parts. Since the tag is generic, we must give an attribute describing the content it is enclosing. An attribute is written inside a tag using the following syntax:

```
<element attribute="value">
```

In the case of `<div>` elements describing sections in a text we often find something like this:

```
<div type="chapter" n="1">
  <p>It is a truth universally acknowledged, that a single man in possession
  of a good fortune must be in want of a wife.</p>

  <p>However little known the feelings or views of such a man may be on his first entering a
</div>
```

Notice that there are two attributes being named inside the tags: type and number. These attributes tell us this particular `<div>` refers to chapter numbers. Divs are usually embedded within each other to form a hierarchy. For example, a `<div type="act">` may include several `<div type="scene">`. **Note**: Usually every section of a book begins with a header. Headers are marked with the tag `<head>`. Ex.:`<head>Letter XXVIII</head>`.

Here is a sample hierarchy that approximates many modern books:

```
<text>
  <front>
    <titlepage>
      <div type="toc">
        <head>Table of contents</head>
        <list>[etc...]</list>
      </div>
    </front>
  <body>
    <div type="part" n="1">
      <div type="chapter" n="1">
        <head>Chapter I</head>
        <p>content</p>
        <p>content</p>
        [etc...]
      </div>
      <div type="chapter" n="2">
        <head>Chapter I</head>
        <p>content</p>
        <p>content</p>
        [etc...]
      </div>
    </div>
    <div type="part" n="2">
    [etc...] </div>
```

```
  </body>
  <back>
    <div type="index">
      <p>content</p>
    </div>
  <back>
</text>
```

Paragraphs

As you may have noticed already, there are two types of elements (or tags): Those that nest content, e.g. `<some_element>some content</some_element>`, and those without content, e.g. `<some_element/>`.

The tags for paragraphs always contain text in between the opening tag `<p>` and the closing tag, `</p>`. This text is said to be *nested* in `<p>`. The `<p>`tag is also shared with HTML and along with `<div>` is perhaps the most common tag out there. Here is the `<p>` tag in action:

```
<p> It is a truth universally acknowledged, that a single man
in possession of a good fortune must be in want of a wife.</p>
```

In your source, prose can be enclosed by the `<p>` tag. It is very important that no content exists outside of the hierarchy.

Lists and Tables

Marking lists in XML follows a straightforward pattern. We use `<list>` and `<item>`. Here is an example of a list used to declare a table of contents:

```
<div type="toc">
  <list type="ordered">
    <item>Prologue</item>
    <item>Chapter 1</item>
    <item>Chapter 2</item>
    <item>Chapter 3</item>
  </list>
</div>
```

You can also have a list within a list:

```
<list rend="numbered">
  <item>a butcher</item>
  <item>a baker</item>
  <item>a candlestick maker, with
    <list rend="bulleted">
      <item>rings on his fingers</item>
      <item>bells on his toes</item>
    </list>
  </item>
```

```
</list>
```

Tables get a bit more fiddly because you have to specify the contents of both rows and individual cells within those rows. But in principle, the idea is the same as with lists. Note the use of the `@role` attribute to categorize the contents of the table cells. Since the writer is quoting from another source, we use the `<q>` element, which means quoted.

```
<table>
  <row>
    <cell role="statement">To Mrs. PAMELA ANDREWS.</cell>
    <cell role="reply">This is my ANSWER</cell>
  </row>
  <row>
    <cell role="statement"><q>The following ARTICLES are proposed to your serious Consideratio
    <cell role="reply"><q>Forgive, good Sir, the Spirit your poor Servant is about to shew in
  </row>
</table>
```

Page and line breaks

Most paragraphs won't require more tweaking than placing them inside `<p>` tags. Some content you will encounter, though, will require you to mark the presence of a carriage return. This is marked in TEI with the line break tag `<lb/>`. Notice this is a tag without content. The use of this tag might not be necessary in your particular text, but it is nice to know it's there.

As opposed to the `<lb/>`, you will use a page beginning tag, `<pb/>` , every time you encounter the start of a new page (NB: these have already been supplied in `template.xml`). It is also here, in this element, that you mark the page number in the form of an `@n` attribute. The value of the `@n` should correspond to the page number. E.g.

```
  <p>Here, what a sad Thing it is! I have been brought up wrong, as Matters
stand. For, you know, my good Lady, now in heaven! lov'd Singing and
  Dancing; and,</p>
  <pb n="93" />
  <p>and, as she would have it, I had a Voice, she made me learn
both; and often and often has she made me sing her an innocent Song, and
  a good Psalm too, and dance before her.</p>
```

Because it is a tag without content, it can be placed anywhere in the hierarchy. In this particular instance, it was placed inside a `<p>` element.

Note: For the most part you will erase line-breaking hyphens when you encounter them in the text and just type the word whole. Once in a while you may find a hyphen at the end of a page. In such cases you should replace the hyphen with the following: `&#x2011;` This is what's called a hexadecimal HTML entity. This one in particular represents the non-breaking character. Ex.:
`...Swinburne's essay on Ford epito&#x2011;<pb n="22"/>mizes the standards...`

Emphasis

One of the most common attributes in TEI is the `@rend` attribute (notice that the `@` symbol precedes attributes when we name them in documentation). The `@rend` attribute usually names the way a particular text segment is rendered in the source. For our text we will probably only use one: italics. Whenever you encounter words in italics in the original we express these most of the times by adding the `@rend` attribute to a `<hi>` (highlight) element (or directly to the `<p>` element if all the content in that paragraph is emphasized). It is another convention to use the language of CSS, which is used to describe how HTML pages are to be rendered online, to express the value of `@rend`. The CSS value for italicized text is "italic." Here is an example:

```
<p> <hi rend="italic"> And this Indenture further witnesseth </hi>
that the said <hi rend="italic"> Walter Shandy </hi> , merchant,
in consideration of the said intended marriage...
</p>
```

Graphical elements

TEI XML isn't just about text. You can also encode information about graphical elements. These special purpose elements are used to describe graphic images within a document:

- `<figure>` (figure) groups elements representing or containing graphic information such as an illustration, formula, or figure.
- `<graphic>` (graphic) indicates the location of a graphic or illustration, either forming part of a text, or providing an image of it.
- `<figDesc>` (description of figure) contains a brief prose description of the appearance or content of a graphic figure, for use when documenting an image without displaying it.

For example:

```
<figure>
 <graphic url="path/to/your/image/file.png"/>
 <figDesc>The figure shows an elaborately decorated room, at least
   twenty-five feet side to side and fifty feet long, with ornate
   mouldings and Corinthian columns on the walls, overstuffed armchairs
   and loveseats arranged in several conversational groupings, and two
   large chandeliers.</figDesc>
</figure>
```

Footnotes

Footnotes may be included in your source text, or you may wish to add your own formal editorial commentary via footnotes. We represent this cross-reference between a place in the source and the place of the footnote by using unique identifiers. The attribute used for unique identifiers in TEI-XML is the `@xml:id` attribute. Whenever we assign an `@xml:id` to an element, we are declaring that no other element in the TEI document has this unique identity. By doing this, we are able to link another place to the one we named by invoking the `@xml:id` using a `@target` attribute.

The two elements you need to know here are the `<ref>` (short for reference) and the `<note>` elements. The `<ref>` element belongs to the place in the source text that points to the footnote. This element usually has the attribute `@target`. The content of the `@target` is the value of the `@xml:id` given to the footnote preceded by the number sign #. The content of the footnote goes inside the `<note>` element. This is also where we assign an `@xml:id`. Here is an example:

```
  <p>This is text that needs to be explained,<ref target="#fn1" type="noteAnchor"><hi rend="su
<!-- ... -->
<note xml:id="fn1" type="footnote" resp="#fg">This is the explanation you were looking for.</n
```

Notice that we also have `@type="noteAnchor"` attribute in the `<ref>` element. It is important to mark this `@type` because we want to leave room in the future for other scholars to expand your encoding and use the `<ref>` for other sorts of cross-references.

Time permitting, write an annotation attached to a place in the source text. Ask for help locating reference sources, if needed.

Comments

Lastly, it is important to know that there is a way to be perfectly free with your own comments on the text you are working with (and that don't get rendered in the HTML output). You will find this is a good way to mark trouble spots you want someone else to look at, for you to explain the rationale behind an unusual tagging decision, or even for you to offer your reading of a particular passage if the spirit moves you! To make a comment, you simply include it within the following characters `<!-- -->`. E.g.

```
<p>...yes I said yes I will Yes.<p>
<!-- lol, Molly should really make up her mind! -->
```

And finally, transformation

XML can be a little hard on the eyes, especially at first. The reason I had you download Firefox is because it has a built-in XSLT processor. In a nutshell, what this means is that if you have paired your XML file with an XSLT stylesheet (and we have, it's called `stylesheet.xsl`) you can transform your TEI-XML into HTML within Firefox. It is a good error-checking practice to transform your text regularly as you encode. And it's also just nice to admire your handiwork!

In Visual Studio Code, save the TEI-XML document you've been working on. Then, either right click on the file and open with Firefox, or go to File > Open File in Firefox and select your XML file. A more reader-friendly HTML version of your encoded text will (hopefully) appear.

As with anything, the stylesheet I have supplied makes certain assumptions about how to format and style your text. You may not agree with all of them. You may also decide to use elements that I have failed to anticipate in the stylesheet. Transforming TEI-XML for the web is beyond the scope of this workshop, but suffice it to say there are many ways to customize the presentation of your source text.