# Brief Introduction to TEI: Exercise 4

Kevin S. Hawkins (based on an exercise created by Susan Schreibman)

15.01.2010 • University College Cork, Ireland



This exercise explains how to use the <oXygen/> XML editor to encode "When You Are Old" by W.B. Yeats using TEI Lite and how to view the resulting file using CSS in a web browser, transformed to HTML, and transformed to PDF.

The exercise assumes that you have a working installation of <oXygen/> version 11.1 as well as the following files all in a single directory:

- `yeats_template.xml`
- `yeats_poem.txt`
- `yeats_challenge.txt`
- `teilite.dtd`
- `yeats.css`

## Part A: Getting started

1. Open <oXygen/>.
2. If there are a lot of panels to the left and right of the main window content, let's first clear them out of the way: they're for advanced users. Close all of them. (If you ever want to reopen any of them, you can find them all under **Perspective → Show View**.)
3. In <oXygen/>, open the file `yeats_template.xml`.
4. To avoid overwriting the template (in case you want to consult it later), choose **File → Save As…** and save the file as `myfirstTEI.xml` in the same directory as the template.

Note that if you click the little triangles next to certain line numbers, you can "fold" elements, hiding all of their content. If you fold line 5 and then line 42, you will see that, at the highest level, this TEI document consists of a `TEI` element (at "the root") and two child elements: `teiHeader` and `text`. In Part B we will fill in `teiHeader`, and in Part C we will fill in `text`.

## Part B: Encoding the bibliographic information ("the header")

The `teiHeader` contains metadata about the TEI document and its source. It has four child elements: `fileDesc`, `encodingDesc`, `profileDesc`, and `revisionDesc`. Only the first is required in a TEI document (according to the schema), but if any others appear, they must be in the order just stated. For our exercise, we will create metadata for only the `fileDesc` and, in particular, for the following child elements:

| | |
|---|---|
| `title` | the title of the work (book, article, poem, etc.), including any subtitles |
| `author` | the author of the work |
| `respStmt` | statement of responsibility for the intellectual content of an edition (in this case, the electronic edition) |
| `resp` | phrase describing the nature of responsibility |
| `name` | proper noun or noun phrase |
| `publicationStmt` | information concerning the publication or distribution of an electronic or other text |
| `sourceDesc` | a bibliographic citation of the work being encoded |

In `yeats_template.xml`, the skeleton of `fileDesc` is already in place. In this exercise you'll first fill in small pieces of information in `titleStmt` and `publicationStmt`, and then you'll encode most of `sourceDesc` yourself.

You should see a number of places in `yeats_template.xml` with two question marks, followed by a short phrase, and then another two question marks. This signals a place that you need to fill in a piece of information.

1. In between the opening and closing `title` tags, you'll find `??title here??`. Replace this text with the title of the poem, `When You Are Old`. Be sure that you leave the opening and closing tags in place. No need to enclose the poem title in inverted commas: this punctuation is not really part of the title but just a convention from print for displaying poem titles.
2. Do the same for `author` element: replace `??author here??` with `William Butler Yeats`.
3. This document has three `respStmt` elements. One is already filled in, but you should supply the content of the `name` elements in the other two.

Your titleStmt should now look like this:

```
<titleStmt>
        <title>When You Are Old</title>
        <author>William Butler Yeats</author>
        <respStmt>
                <resp>Creation of machine-readable text by</resp>
                <name>Susan Schreibman</name>
        </respStmt>
        <respStmt>
                <resp>Header creation by</resp>
                <name>Kevin S. Hawkins</name>
        </respStmt>
        <respStmt>
                <resp>Encoded by</resp>
                <name>Kevin S. Hawkins</name>
        </respStmt>
</titleStmt>
```

Continuing …

4. The next text to fill in is in the `publicationStmt`. In this example, you are affiliated with a fictitious publisher, the Online Text Archive. All the information has already been filled in except today's date, which you should supply in any format you like as the content of the `date` element.
5. Because dates can be written in so many formats, TEI provides the `when` attribute on the `date` element. This attribute is used to give the date in a standard format as defined by the World Wide Web Consortium (W3C) so that the date can be understood by a computer. The template contains `YYYY-MM-DD` as the default attribute value; change this to give today's date (using the Gregorian calendar with Arabic numerals).
6. Further down you will find a place to fill in bibliographic information about the book the poem was taken from. You'll find this information in `yeats_poem.txt` (near the bottom of the file), which you should open in addition to the encoded text you are working on. (This file also contains a transcription of the poem in plain text, which will save you the time of retyping it from scratch as you work.) Copy the book's title, editor, and publication information from yeats_poem.txt and paste in between the opening and closing `bibl` elements in `myfirstTEI.xml`.

Now you'll need to place the appropriate tags around this bibliographic information. We'll start with the title:

7. Put the cursor in front of the title (before 'The Collected Works of'). Type a left angle bracket (<). A menu will pop up showing the names of elements allowed at this point in the document according to the schema. (As you can see, there are quite a few!). We're going to use the `title` element, and as you start typing its name, the choice of elements matching

your search will decrease. When there's only one element left, you can press **Enter** to choose it. <oXygen/> automatically inserts both the opening and closing tags. You'll need to move the closing tag to after the end of the title (after 'The Poems.').

Let's use another method of inserting elements to tag the name of the editor and the publication information. This method won't require us to move the closing tag.

8. Highlight the name of the editor (but not 'Edited by').
9. Type ⌘ + E (on a Mac) or Ctrl + E (in Windows). A dialogue box opens with a list of the elements which may be inserted at this point according to the schema. Choose editor. The tags will be inserted before and after the text you highlighted.
10. Tag the name of the publisher, the place of publication, and the date of publication using the appropriate elements given in the table above.

When you are marking up (encoding) texts, don't worry about extra spaces, tabs, or carriage returns (blank lines). These are all called 'whitespace', and XML ignores them, reducing all of these to a single space. Feel free to add blank lines or spaces between elements to make them more readable as you are working on them. Also feel free to use the **Format and Indent** feature of <oXygen/> to have the program clean up your markup for ease of reading.

11. Use **File** → **Save** to save changes made so far.

Your `sourceDesc` should look like this:

```
<sourceDesc>
    <bibl>
        <title>The Collected Works of W.B. Yeats, Volume I: The Poems.</title> Edited by
            <editor>Richard J. Finneran</editor>. <publisher>Macmillan</publisher>:
            <pubPlace>New York</pubPlace>, <date>1989</date>. </bibl>
</sourceDesc>
```

Now let's check that you haven't made any errors in element names or how they are nested. We'll validate the document against the schema.

12. Click the icon with the red check mark. In the status bar of the <oXygen/> window (at the bottom), it will say "Document is valid" or "Validation – failed." If the latter, <oXygen/> will tell you what errors exist and what line numbers to find these on. Fix any errors you encounter. (If you don't encounter any errors, try creating one by intentionally misspelling a tag. Once you validate, you'll see that <oXygen/> immediately detects this problem.)

Congratulations! You've just finished encoding your first header!

## Part C: Encoding the body of the document

1. Check that you have removed all instances of '??' in your document in the course of completing Part B. Use **Find** → **Quick Find…** to do this. This menu command opens a small dialogue box at the bottom of the <oXygen/>

window where you can type text to search for. As you see, the **Find** menu also offers more advanced search options. If you find any question marks, finish following the instructions in Part B.

2. Scroll down to the `text` element. (You may choose to collapse the `teiHeader` if you would prefer to keep this text out of sight while working with the body of the document.)

The text currently has only one child element, a `body`. (In TEI, a text can have a `front` and a `back` as well, used for front and back matter such as a preface, forward, appendix, or index.) Inside of this, you'll find a single `div` element (a division of text), which contains the title of the poem as it appears in the book (in a `head` element), followed by the first stanza, which has already been encoded for you.

The first stanza is in an `lg` element ('line group'), which contains four `l` ('line') child elements. Note the use of the `n` attribute ('number') on the `lg` and `l` elements, indicating the stanza number and the line number.

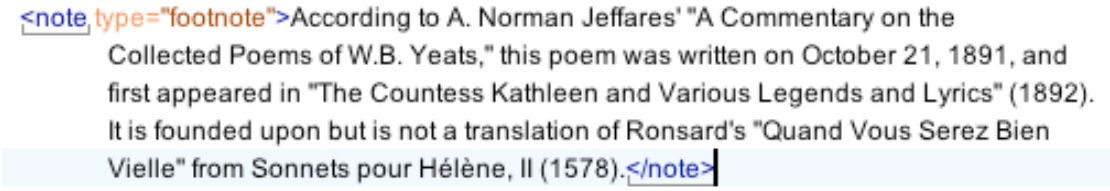
3. Put your cursor after the closing `lg` tag and press Enter to create a new blank line.
4. Copy the next two stanzas from `yeats_poem.txt` and insert them in on the new blank line.
5. Using the methods for inserting XML elements given in Part B, insert `lg` and `l` elements around each stanza and line of text that you just pasted in. When you're done, you might want to use the **Format and Indent** feature and then validate the document. (Both are described above.)

We should add `n` attributes to these new stanzas and lines. TEI does not define whether numbering begins anew with each line group or runs continuously throughout the text—it's up to the encoder. Let's number continuously. You can insert attributes on elements by typing them in directly; <oXygen/> has automatic completion features similar to those for elements.

6. Insert n attributes on each stanza and line.
7. Optionally, validate again.

Now we need to encode the note found in yeats_poem.txt, which begins 'According to' and ends with '(1578)'.

8. Copy the note text into the XML document after the last `</lg>`.
9. Wrap this chunk of text in a `note` element with a `type` attribute whose value is `footnote`. That is, you'll have:




10. Validate and save your changes.

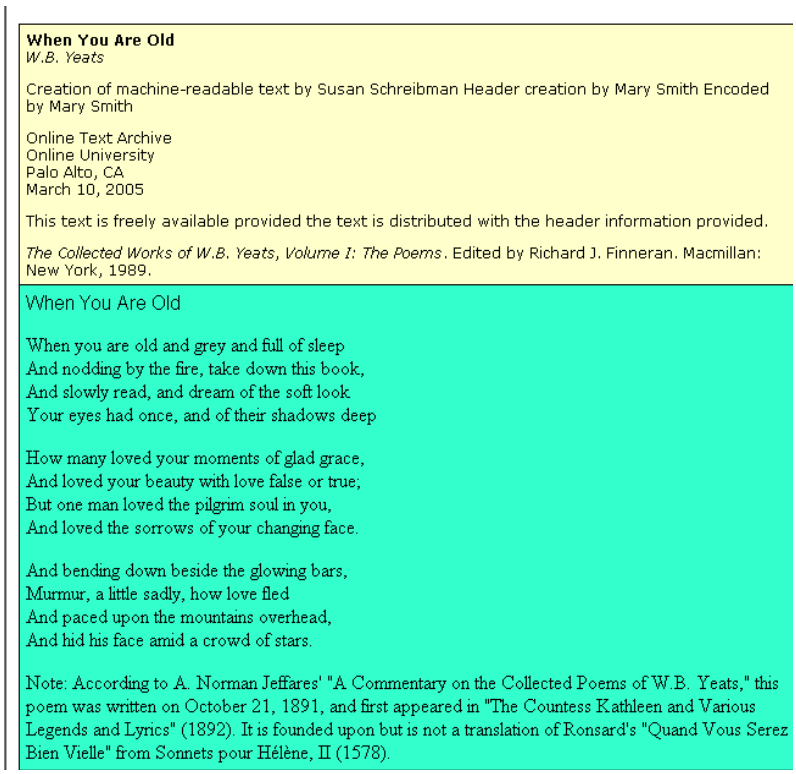
## Part D: Using CSS with the TEI document

Cascading Style Sheets (CSS) is a language for describing the appearance of documents written in XML or other markup languages. It is commonly used to control the appearance of webpages in HTML, but you can use it tell your browser how to display TEI elements as well.

Your file `myfirstTEI.xml` has a  declaration on line 3 that points to a stylesheet called `yeats.css`. When you open `myfirstTEI.xml` in a browser, the browser looks for this stylesheet and uses it to control the appearance.

We'll look at and modify `yeats.css` in the following steps.

1. In a web browser, open `myfirstTEI.xml`. You should see something like this:



**When You Are Old**
*W.B. Yeats*

Creation of machine-readable text by Susan Schreibman Header creation by Mary Smith Encoded by Mary Smith

Online Text Archive
Online University
Palo Alto, CA
March 10, 2005

This text is freely available provided the text is distributed with the header information provided.

*The Collected Works of W.B. Yeats, Volume I: The Poems*. Edited by Richard J. Finneran. Macmillan: New York, 1989.

When You Are Old

When you are old and grey and full of sleep
And nodding by the fire, take down this book,
And slowly read, and dream of the soft look
Your eyes had once, and of their shadows deep

How many loved your moments of glad grace,
And loved your beauty with love false or true;
But one man loved the pilgrim soul in you,
And loved the sorrows of your changing face.

And bending down beside the glowing bars,
Murmur, a little sadly, how love fled
And paced upon the mountains overhead,
And hid his face amid a crowd of stars.

Note: According to A. Norman Jeffares' "A Commentary on the Collected Poems of W.B. Yeats," this poem was written on October 21, 1891, and first appeared in "The Countess Kathleen and Various Legends and Lyrics" (1892). It is founded upon but is not a translation of Ronsard's "Quand Vous Serez Bien Vielle" from Sonnets pour Hélène, II (1578).

2. In <oXygen/>, open `yeats.css`.

A CSS file lists elements, followed by rules (in curly braces) governing their display. The elements are listed in alphabetical order, but they need not be listed that way. Some rules are given all on one line, and others are formatted to be more readable by using multiple lines. As with XML documens, whitespace in CSS files is irrelevant.

As you can see in the web browser, the header has a light yellow background, and the body has a bright green background. Let's change the green to a more attractive shade.

3. In <oXygen/>, replace the color code for `body`, replacing `#33FFCC` with the new code `#CCFFCC`. (These are hexadecimal red-green-blue (RGB) color codes used to define colors in CSS.)

4. Save the file.
5. In the web browser, refresh/reload the page. Hold down the shift key while doing this to clear the browser cache, ensuring that it won't just show you the same version again. The background color should have changed to something more palatable.

Let's now set the title of the poem in the body (not in the header) to be bold.

6. Find the CSS rule for the `head` element. Add a `font-weight` property with value `bold` to the rule. The resulting rule should be:

   ```
   head { display: block; font-family: arial; margin-bottom: 1em; font-weight: bold}
   ```

7. Save the file and refresh the page in your browser. The first line in the yellow box should now be bold!

## Part E: Using XSL-FO to generate a PDF

The Extensible Stylesheet Language (XSL) lets you transform an XML document into another type of XML document (using XSL Transformations or 'XSLT') or into a non-XML format (using XSL Formatting Objects or 'XSL-FO'). <oXygen/> includes some XSL-FO stylesheets for generating PDFs from TEI documents. Let's use one of the default stylesheets to make a PDF from `myfirstTEI.xml`.

1. In <oXygen/>, choose **Document → Transformation → Configure Transformation Scenario…**
2. In this dialogue box, create a new **XML transformation with XSLT** by clicking **New**.
3. The **XSLT** tab should be open by default, with `${currentFileURL}` filled in as the **XML URL**. This is the name of the input file. Leave this as it is. Below this is a blank for **XSL URL**. Here you need to specify the stylesheet to be used to do the transformation. Click the folder icon to the right of this box.
   - On Mac OS X, choose **(your hard drive) : Applications : (name of <oXygen/> folder) : frameworks : tei : xml : tei : stylesheet : fo2 : teiCustom.xsl**.
   - In Windows, choose **(your hard drive)\Program Files\(name of <oXygen/> folder)\frameworks\tei\xml\tei\stylesheets\fo2\teiCustom.xsl**.
4. Click the **FO Processor** tab. Check the box next to **Perform FO Processing** and leave the other options as their defaults (**XSLT result as input**, **pdf** and **Apache FOP**).
5. Click the **Output** tab. Click the radio button next to **Prompt for file** so that you will be asked to name the output file when you run the transformation.
6. Click **OK**.
7. You should now see a scenario in the list called **myfirstTEI.xml**. Click **Transform now** to run it. You will be prompted to save the file. Call it **myfirstTEI.pdf**.
8. Open **myfirstTEI.pdf** in a PDF viewer.

Setting up the transformation scenario involves many steps, but know that if you create a TEI document from scratch within <oXygen/>, this transformation scenario is already configured for you!

### Part F: A challenge (in case you have time)

Let's say you have been asked to encode an anthology of poems rather than a single poem. You'll find two more poems in `yeats_challenge.txt`. Add these to `myfirstTEI.xml`.

The challenge here is that we want the encoding to reflect the fact that this is a collection of poems as opposed to a single poem. Also note that at the end of 'To a Shade' there is a dateline to the poem. That too needs to be reflected in the markup.

Go to the TEI Consortium website ( http://www.tei-c.org/ ), navigate to the page of customizations, and choose the TEI Lite documentation (in HTML or PDF format). Look at this documentation for advice on how to encode an anthology.