TEI Publishing Options

Lou Burnard Consulting *July 2014*

In this exercise, we'll look at two different ways of delivering richly encoded TEI XML documents to a web browser.

1 Using TEI Boilerplate

TEI Boilerplate, developed by John Walsh of Indiana University, is a simple tool for displaying TEI-XML in any web browser. You can download it from `http://teiboilerplate.org`, where you will also find much more detailed documentation and discussion.

- Download the file TEI-Boilerplate.zip to your desktop
- Click on it to unzip it. It will create a folder called TEI-Boilerplate-master, containing a folder called content amongst other things. Files to be displayed by TEI Boilerplate must be copied into this folder.
- Boilerplate also requires you to add a special *processing instruction* to the start of each file you want displayed

Boilerplate uses your browser to look at files, either on a web server, or locally. The same operations are necessary in either case. Apart from its simplicity, the other main attraction of TEI Boilerplate is that it allows you to control the way your document is displayed directly, by using the standard style language CSS, without any need to carry out an intermediate HTML conversion.

- Open a fully-tagged TEI XML document with oXygen. Any document containing some `<persName>`s and `<placeName>`s will do. If it has some linked page images, so much the better. We suggest using the document you have been working on in the previous exercises, but as this is quite small, you may prefer to use the file guy_final.xml in your Work/Guy folder (and in this case you should definitely also copy the image files from the same folder into the content folder).
- Add the following magical incantantion to the start of the file (after the `<xml version="1.0" encoding="UTF-8"?>` line) `<?xml-stylesheet type="text/xsl" href="teibp.xsl"?>`
- Save this version of the file in the content folder within the folder you created when you installed TEI Boilerplate
- If your XML document references any image files, you will need to copy these to the same folder as well .
- Open your favourite web browser (Chrome, Firefox, Explorer...) and look for the option to open a local file. For Firefox, this is the Open command on the File menu; or (for Chrome or Firefox) just type CTRL-O.
- Navigate to the version of your file which you just saved in the content folder, and open it

By default, quite a lot of the markup in your document is automatically handled by Boilerplate. Additions, deletions, and notes for example are all displayed appropriately. Page-images (if there are any) will be displayed if you click the icon on the right hand of the screen. But, by default, the names of people or places you have so painstakingly marked up are not displayed any differently from the rest of the text. We will fix that, by adding some specifications to the header of the document.

- Return to your XML file in oXygen
- Place the cursor just inside the `<encodingDesc>` element which follows your `</fileDesc>` (if there is no `<encodingDesc>` element in your header, you can insert one here)
- Select File -> Insert file from the Document menu, navigate to the file renditions.xml in your Work folder, and click Open to insert this into your document inside its `<encodingDesc>`
- Save this new version of your document in the right place, i.e. the folder content
- Reload the file in your web browser. Is that better?

How does it work? The `<tagsDecl>` element you have just added to your TEI Header has two parts. First, there is a series of declarations like this

```
<rendition xml:id="blue" scheme="css"> color:blue; </rendition>
```

, each of which defines a particular display style and gives it an identifier, in this case, `blue`. The display style itself is expressed in the CSS language (CSS is a W3C standard for definition of formatting styles). As you probably guessed, the style called `blue` displays text in blue, the style called `i` displays text in italic etc.

Secondly, there is a `<namespace>` element containing a series of declarations for elements taken from this namespace, for example like this:

```
<namespace name="http://www.tei-c.org/ns/1.0">
 <tagUsage gi="placeName" render="#blue"/>
 <tagUsage gi="orgName" render="#i #blue"/> ...
</namespace>
```

These declarations specify that every occurrence of the TEI `<placeName>` element will by default be displayed in blue (or, more precisely, using the style called `blue`). Every occurrence of `<orgName>` will be displayed in blue and also in italic.

To override this default behaviour for a single element in your document, you can either supply the CSS code you want directly on the element concerned, using the *@style* attribute, or (more economically) you can use the *@rendition* attribute to point to a predefined `<rendition>` element. To test this, try making the second paragraph in your document appear in italic. You should need just to modify its start tag like this: `<p rendition="#i">`

The file teibp.xml in your TEI-Boilerplate-master folder contains a fuller list of styles which you can add to your header if you would like to use them.

2 Extracting data with an XSLT stylesheet

TEI Boilerplate allows us to display our XML document in many different ways, but it does not easily enable us to add extra information to it. An XSLT stylesheet however can easily extract and reorganize the content of the input document. In this exercise we'll run a pre-existing XSLT stylesheet to enrich our document with a separate index of the names referenced in the document.

First, you need to add some markup to your XML document to indicate whereabouts you want this generated list of names to appear.

- With your XML document open in oXygen, add an additional line like this as the first component of the `<text>` element (i.e. *before* the `<body>` element begins)

```
<front>
 <divGen type="namelist"/>
</front>
```

- Remember to save the file!

Now you need to run an XSLT transformation which will pick up the names referenced in the document, sort them alphabetically, and display a list in the place where the `<divGen>` element has been inserted. The transformation will also convert your XML to HTML so that it can easily be displayed. This sort of task is easily done by customizing the existing TEI Stylesheets, and in fact has already been done for a French project working on the OuLiPo archive. The stylesheet is accessible using Oxgarage, and you can also find it in the Oxygen TEI framework.

- Visit `http://www.tei-c.org/oxGarage/`
- Select TEI P5 XML as input format, and XHTML as output format
- Navigate to the file you modified in the previous step and select it
- Before pressing the Convert button, click Show advanced options
- Select oulipo from the dropdown list of conversion profiles
- To save time, check the box that says "Convert text only, don't include any images from the document"
- Now press convert and wait
- Save the output file which oxGarage sends you and open it with your browser.

You should see a list of the names mentioned in the document, at the location where you inserted the `<divGen>` element, plus an automtically generated table of contents.

Finally, we invite you to consider the following open questions :

- What further modifications would you like to make in the way this document is rendered?
- Would these enhancements require any change in the markup of your document?
- What are the pros and cons of transforming your TEI XML source into a separate HTML version (the second approach) rather than rendering it directly (as Boilerplate does)?