Eine kleine text-encoding

Lou Burnard Consulting *mai 2014*

In this exercise, we'll use oXygen to :

- markup an existing raw text document
- use regular expressions to speed up the process

Our goal is to achieve this without actually typing any tags!

As a case study, we've chosen the start of an 18th century English chapbook called *Guy of Warwick*, one of hundreds of versions of this story. The whole text is available in the Bodleian Library if you want to read it. In your Work folder, you'll find a number of image files (page_01.png, page_02.png etc.) each corresponding with a page in the original source. There are also some image files showing individual woodcut images used on some pages. Take a quick look, to familiarize yourself with this document and think about how it should be marked up in TEI.

1 Make a new TEI document

As you saw in the previous exercise, when creating a new TEI XML document, it's important to consider which schema to use, since this will determine what tags and attributes oXygen makes available. Later on you will learn how to make a schema which is customised for a particular project; for the moment, we will accept the default ptroposed by oXygen, which is to permit any valid TEI element.

- Launch oXygen....
- Click the leftmost icon in the toolbar (or choose New from the File menu, or type CTRL-n) to open the New dialog.
- In the New dialog, select Framework Templates, then TEI P5, then All. Your new document will use the TEI-All schema.
- Click the Create at the foot of the dialog. A skeleton document appears for you to complete.
- We will leave the TEI Header aside for the moment (you will complete it in the next exercise). Our new document has many chapters, so we will need to introduce an additional layer of structure between `<p>` and `<body>` elements
- Select the `<p>` element proposed as the content of the `<body>` with the mouse (the whole thing, incuding its tags).
- Type CTRL-E (or select XML Refactoring -> Surround with Tags from the Document menu).
- oXygen offers a list of possible tags at this point. Choose `div` and press Return.
- Put the cursor inside the opening tag for the `<div>` you have now created, just before the closing `<`and type a single space
- oXygen offers a list of possible attributes at this point. Choose `type` and press Return.
- Type `chapter` as the value for the *@type* attribute you have now added

```
<TEI xmlns="http://www.tei-c.org/ns/1.0">
 <teiHeader>
    <fileDesc>
      <titleStmt>
        <title>Title</title>
      </titleStmt>
      <publicationStmt>
        <p>Publication Information</p>
      </publicationStmt>
      <sourceDesc>
        <p>Information about the source</p>
      </sourceDesc>
    </fileDesc>
 </teiHeader>
 <text>
    <body>
      <div type="chapter"> <p>Some text here.</p>
</div>    </body>
 </text>
</TEI>
```

- Your document is now ready for you to start transcribing the first chapter in place of the text `Some text here`. It should now look something like this..

However, to reduce the amount of typing you have to do, there is a transcript (of sorts) already available for you to use in the file guy.txt in your Work/Guy folder.

- First, delete the words `Some text here`, leaving the cursor inside the now empty `<p>` element
- Choose File -> Insert File from the Document menu (*not* the File menu!) ; then navigate to the file guy.txt in your Work/Guy folder and open it
- The green square at top right turns into a red one because your document is no longer valid! Don't panic: this is an easy thing to fix. Click on the red mark to the right of the editing window to go straight to the location of the error.

Sweet Lady said Guy, I make no doubt but quickly to obtain his Love & Favour, let me have thy Love first fair Phillis, and there is no fear of thy Father's Wrath preventing us.—It is an old saying, Get the Good-will of the Daughter, and that of the Parent will soon follow.

- The error message displayed should remind you that the ampersand is a special character in XML: if you want to use one in your document you have to replace it by what is called an *entity reference*
- Replace the & by `&amp;` and all will be well.
- Your document is now syntactically valid, but it is clearly not honest, since its markup asserts that the document contains just one paragraph and one chapter.

2 Divide and conquer

- Put the cursor at the end of the first paragraph of the text (i.e. after the words `acquainted with his Name.` at the end of line 43
- Type ALT-SHIFT-D (or selectXML Refactoring -> Split Element from the Document menu.
- Repeat for each subsequent paragraph (e.g. after `Black a Moor to her` at the end of line 46, after `and am obliged to you.` at the end of line 58, after `Roses and`

`other Flowers.` at the end of line 60, after `on Conditions which he accepts` at the end of line 64, and so on to the last paragraph, after the chief `Nobles and Barons of the land being present.` on line 156.

This is an improvement, in that we now have many paragraphs instead of one, but we still have only one chapter. To divide that in the same way, we need to put the cursor between the end of the last paragraph in one chapter, and the start of the first paragraph in the next. For example, the start of chapter two looks something like this:

```
The Doctor departed, and left Guy to cast his Eyes on the heavenly Face of his Phillis, as she was
walking in a Garden full of Roses and other Flowers.</p><p>

CHAP. II.

Guy courts fair Phillis, she at first denies, but afterwards grants his Suit, on Conditions which he accept
</p><p>
GUY immediately advanced to fair Phillis, who was reposing herself in an Arbour, and saluted her with
bended Knees. All.hail, fair Phillis, Flower of Beauty, and Jewel of Virtue, I know great Princes seek to wi
```

- Put the cursor after the sequence `Flowers.  </p` and before the characters `<p>`, and type ALT-SHIFT-D again.

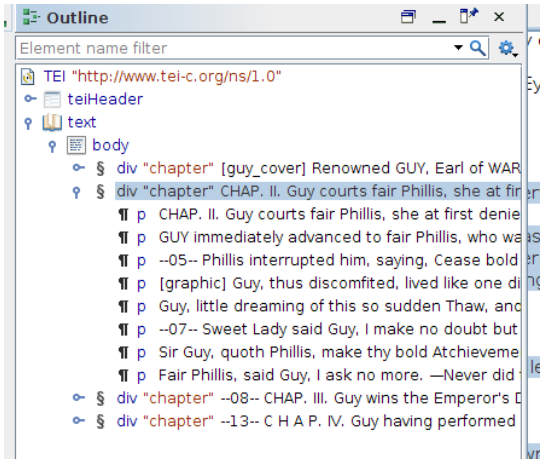
The result should look like this:

```
The Doctor departed, and left Guy to cast his Eyes on the heavenly Face of his Phillis, as she was
walking in a Garden full of Roses and other Flowers.</p></div><div type="chapter"><p>

CHAP. II.

Guy courts fair Phillis, she at first denies, but afterwards grants his Suit, on Conditions which he accepts
</p><p>
```

- While we're here, let's add a number for the chapter. Put the cursor inside the `<div>` start tag, and type a space as before.
- Select `n` from the list of attributes available, and give it the value `2`
- Repeat for the other chapters, numbering them 3 and 4 respectively.

3 Honesty is the best policy

You can check the structure of your document by looking at it in outline view. (Select Show View->Outline from the Window menu to display this). You should have a structure more or less like this:



Several things still need improving! We have marked up the heading text at the start of each chapter as if it were just another paragraph; we have not marked the page divisions at all (they are indicated in the source by lines like this `--03--` at the start of page three), nor have we dealt properly with the illustrations, which are indicated in the source by lines like this `[graphic]`. And we haven't tried to mark up the front matter of the text at all.

- Put the cursor inside one of the headings which has erroneously been tagged as a `<p>`, inside or adjacent to the start or end tag, and select XML Refactoring -=> Rename Element from the Document menu (if this option is not available, the cursor is not inside or adjacent to a tag)
- Select `head` to change this to a more truthful tag (if this generates an error, maybe you are operating on the wrong tag: you should start with the first `<p>` in a `<div>`)
- Repeat for the other chapters. You may wish to give each chapter two `<head>` elements, one containing e.g. `CHAP II` and the other containing the brief summary: you can of course do this easily by splitting the `<head>` if need be.

To mark up the graphics, you need to find each occurrence of the text `[graphic]` in the source file and replace it with (at least) `<figure><graphic url="page_xx_detail.png"/></figure>` where `xx` is the number of the page containing this graphic, since that is how the graphic files have been named. You could do this entirely by hand, since some manual intervention will be needed, or you could do it in two passes, as follows.

- With the cursor at the start of your document, type CTRL-F (or select Find/Replace on theFind menu)
- In the Find/Replace dialog which opens, type `[graphic]` in the upper "Text to find" window, and `<figure><graphic url="page_xx_detail.png"/></figure>` in the lower "Replace with" window
- Click the Replace all button
- There should be 5 matches found. Click Close to remove the dialog box
- You now need to review each of the replacements made, changing the `xx` it contains into the number of the page on which it appears (e.g. `14` for the last one). The numbers are given by lines like this `--14--` and by convention always *precede* the graphic.
- For extra marks, you may also like to add a `<figDesc>` element within the `<figure>` giving a brief description of the content of the woodcut.

Try switching to the 'Author' mode if you would like to see the graphics displayed along with the text.

Next, we suggest you correct the start of the document. At present, the title page appears as part of the first chapter, which cannot be right.

- Place the cursor between the `<text>` start-tag and the `<body>` start-tag at the beginning of your document, and type a <
- From the list of available elements which appears, select `<front>`
- Move the cursor inside the `<front>` element, and type another <
- This time select `<titlePage>`.

- Repeat this manouevre to insert the following elements within your title page:
  - a `<docTitle>` containing a `<titlePart>` into which you transfer the document title ("THE H I S T O R Y OF GUY, Earl of WARWICK."), another `<titlePart>` containing the `<figure>` which contains the woodcut from page 1
  - a `<docImprint>` ellement containing the text of the imprint ("LONDON: Printed for the Booksellers.")
  - Once you have moved these components by cut and paste, the start of your document should look something like this:

```
<text>
  <front>
    <titlePage>
      <docTitle>
        <titlePart>THE H I S T O R Y OF GUY, Earl of WARWICK.</titlePart>
      </docTitle>
      <figure>
        <graphic url="page_01_detail.png"/>
      </figure>
      <docImprint>LONDON: Printed for the Booksellers. </docImprint>
    </titlePage>
  </front>
  <body>
    <div type="chapter" n="1">
      <head>--02-- The HISTORY of Guy, Earl of Warwick. </head>
      <head> CHAP. I.</head>
      <head> Guy's Praise. He falls in Love with fair Phillis.</head>
      <p> IN the blessed Time when Athelstone wore the Crown of the English Nation, Sir Guy
        (Warwick's Mirror and all the World's Wonder) was the chief Hero of the Age; whose
```

4 Page numbering

So far our tagging has focussed on the logical organization of the document : but it is also very useful to inclde in our tagging some information about its physical organization, notably to indicate where each page begins. As we noted above, these are indicated in the source file with lines like this `--12--`, which appears at the start of the page numbered 12.

To deal with these, we suggest using one of the more sophisticated features of oXygen : its support for regular expression matching in the find an replace dialogue you have already seen. We will use a *regular expression* to identify these lines and then replace them with an appropriate `<pb>` tag.

- With the cursor at the start of your document, type CTRL-F (or select Find/Replace on theFind menu)
- In the Find/Replace dialog which opens, type `--(\d+)--` in the upper "Text to find" window, and `<pb facs="page_\1.png" n="\1"/>` in the lower "Replace with" window
- At the bottom of the dialog box, select the option Regular expression
- Click the Replace all button
- There should be 14 matches found. Click Close to close the dialog box

5 And finally ...

There are of course many more things we could do to enhance this encoding. Here are just a few suggestions: ...

- The text contains many passages of direct speech which are not signalled conventionally with quotation marks. IIt would be useful to capture these in our markup all the same. Selec each of the spoken passages with the mouse and (using CTRL-E) tag them as `<said>` elements.

- The text also contains many proper names, both of people and of places. You may like to tag the former as `<persName>` elements and the latter as `<placeName>` elements (Later on we will investigate in more detail ways of handling these 'named entities'

When you are done, don't forget to save your work! We'll be using it again later.