Software: CommentaryToEpidoc.py

Author: J W Boyle

Introduction

A Python module called CommentaryToEpidoc has been written by Jonathan Boyle (University of Manchester) to convert transcribed commentaries to EpiDoc compatible XML. The version of Python is Python 3.5.1 (using Anaconda Python 2.4.1). For more information on EpiDoc see <http://www.stoa.org/epidoc/gl/latest/>and <http://sourceforge.net/p/epidoc/wiki/Home/>.

The commentaries were transcribed by Kamran Karimullah and colleagues at the University of Manchester, Kamran also specified the requirements for the module. The commentaries are Arabic, however the module can process text in any language.

The work was funded from an ERC funded project studying Arabic commentaries on the Hippocratic Aphorisms, the Principal Investigator was Peter E. Pormann (The University of Manchester).

Format for commentaries

It is assumed the commentaries are utf-8 text files consisting of a main body of text, followed by a list of numbered and ordered footnotes describing (i) omissions, (ii) additions, (iii) correxi, (iv) conieci and (v) other textual variations from two source documents.

The main body of text should have a block of text containing an optional introduction and the document title followed by a series of numbered aphorisms and commentaries. If an introduction is included it should be separated from the title by a line containing the two plus symbols, i.e. ‘++’, the title can be more than one line of text.

The aphorisms and commentaries should have the following format.

1. A single line contains the aphorism number followed by a period ‘.’ character, e.g. 1. for aphorism 1
2. The next line contains the aphorism text
3. The next line (or lines) contain one or more commentaries associated with that aphorism, with each commentary on a single line
4. This is repeated for any additional aphorisms

This main text (title, aphorisms and commentaries) contains symbols referring to (1) witnesses and (2) footnotes.

1. Witness symbols have the form [WW LL] where WW is a code to identify the document and LL is a location in the document. Witness codes should be listed in the <listWit> element in the XML template file described below.
2. Footnote symbols have one of two forms depending on whether they refer to single word or multiple word variations. If tttt represent a word of text without a variant, vvvv represent a word of text with a variant, and n is the footnote number then single word variants use form (a) and multiple word variations use form (b) where the # marks the end of the variation.
   1. ttt tttt \*n\*vvvv tttt tttt
   2. ttt tttt \*n\*vvvv vvvv vvvv# tttt tttt

After the main body of text is a list of numbered and ordered footnotes, each footnote has a corresponding reference in the main text. A footnote is a single line with the following format.

1. At the start is the footnote number enclosed within a pair of asterisks, e.g. for footnote 1 the line starts with \*1\*
2. Following the footnote number is a mix of witness text (i.e. title, aphorism or commentary) and symbols defining omissions, additions, correxi, conieci and other variations
3. The footnote line ends with a period (.) character

The formats for the five footnote types are now described where n is the footnote number, W1 and W2 are witness codes, and ssss, tttt and uuuu represent segments of witness text (segments can be one or more words in length).

1. Omissions have only one form, i.e.   
   \*n\*ssss ] W1: om. W2.  
   This means the segment of text 'ssss' is found in witness W1 but not W2.
2. Additions have three forms depending on whether the addition applies to one or both witnesses, and for the latter case whether the addition is the same or not for both witnesses.  
   Form 1: \*n\*ssss ] add. tttt W1.  
   This means both witnesses have 'ssss' and W1 adds 'tttt'.  
   Form 2: \*n\*ssss ] add. tttt W1, W2.  
   This means both witnesses have 'ssss' and both add 'tttt', however the editor felt the need to omit tttt.  
   Form 3: \*n\*ssss ] add. tttt W1: uuuu W2.  
   This means both witnesses have 'ssss', W1 adds 'tttt' whereas W2 adds 'uuuu'.
3. Correxi can have two forms, depending on whether witness texts are the same or not.  
   Form 1: \*n\*ssss ] correxi: tttt W1, W2.  
   This means the text 'tttt' is found in witnesses W1 and W2 and the editor has corrected this to 'ssss'.  
   Form 2: \*n\*ssss ] correxi: tttt W1: uuuu W2.  
   This means the text 'tttt' is found in witness W1, whereas W2 has 'uuuu'. The editor has corrected these to 'ssss'.
4. Conieci can have two forms, depending on whether the witness texts are the same or not.  
   Form 1: \*n\*ssss ] conieci: tttt W1, W2.  
   This means the text 'tttt' is found in witnesses W1 and W2, the editor conjectures that this should be 'ssss'.  
   Form 2: \*n\*ssss ] conieci: tttt W1: uuuu W2.  
   This means the text 'tttt' is found in witness W1, whereas W2 has 'uuuu'. The editor conjectures that these should be 'ssss'.
5. Standard variations have only one form.  
   \*n\*ssss ] W1: tttt W2.  
   This means witness W1 has text 'ssss' whereas W2 has 'tttt'.

A text file base name should end in an underscore followed by a numerical value representing the document number, e.g. file\_1.txt, file\_2.txt, etc. The numerical value after the underscore is subsequently used to create the title section <div> element, e.g. <div n="1" type="Title\_section"> for file\_1.txt.

XML files

The Python module generates EpiDoc XML for insertion within the <body> element. Therefore the Python module requires a suitable XML template file as input which contains all XML other than the contents of the <body> section. The template file should contain the string '#INSERT#' at the location where the new XML should be inserted i.e. the template should contain:

<body>

#INSERT#

</body>

in the XML.

After successful processing each text file two XML files are created in a folder called XML. The new files contain the main body of XML and the apparatus XML linked using the double-end-point-attached method as described here <http://www.tei-c.org/release/doc/tei-p5-doc/en/html/TC.html#TCAPLK>.

The names for the new files is the text file base name plus ‘\_main.xml’ (for the main XML) and ‘\_apps.xml’ (for the apparatus XML). For example for text file file\_1.txt the XML will be file\_1\_main.xml and file\_1\_app.xml.

Python module

The expected way of using the module is via the process\_text\_files(…) function which attempts to process all files with the .txt extension within a specified directory. Input arguments for this function are:

1. text\_folder - the folder containing the text file
2. template\_file - the name of the XML template file
3. n\_offset - the number of offsets to use when inserting XML in the <body> element (default is 0)
4. offset\_size -the number of space characters for each XML offset (default value is 4)

Additional functions:

1. process\_file – called by process\_text\_files to process a single text file
2. save\_error – called by process\_file to output error messages to file
3. test\_footnotes – called by process\_file to test the footnotes for errors
4. get\_next\_non\_empty\_line – called by process\_file to get the next line containing text
5. process\_references – called by process\_file to process generate XML for references from a single line of commentary text
6. process\_footnotes – called by process\_file to process footnote symbols and generate XML from a single line of commentary text
7. process\_omission – called by process\_footnotes to generate XML for an omission
8. process\_addition – called by process\_footnotes to generate XML for an addition
9. process\_correxi – called by process\_footnotes to generate XML for a correxi
10. process\_conieci – called by process\_footnotes to generate XML for a conieci
11. process\_standard\_variant – called by process\_footnotes to generate XML for a standard variation

The file driver.py demonstrates how to use the module

GitHub repository

To clone the GitHub repository first set up Git on the local hardware (e.g. see <https://help.github.com/articles/set-up-git/>) and run the following command

git clone https://github.com/Jonathan3145/CommentaryToEpidoc.git