Exercise: Advanced XSLT

The task in this exercise is to generate different views of the TEI XML, and to enhance it.

1 Summarizing of Punch

- Produce an HTML table (<table> with <tr> for row, <td> for cell) at the beginning of the document listing each section of the document, with its sequence number, heading, and type. Process each <div> twice, the first time using a ‘mode’ on the template.

- Repeat this, but now make an HTML anchor at the start of each section, and link to that from the table. You can use `generate-id()` to work out the identifier. So generating the link in the summary may look like this:

```xml
<a href="#{generate-id()}
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:tei="http://www.tei-c.org/ns/1.0"> ...</a>
```

and the anchor like this:

```xml
<a name="{generate-id()}
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:tei="http://www.tei-c.org/ns/1.0"/>
```

- Now produce an index of all the distinct names in the Punch text. You can do this using XSLT 2.0’s `<for-each-group>` element. You can create the output as an unordered list (start a `<ul>` and put each name inside a `<li>`).

2 Enhancing Punch

Our final task is to generate a preliminary `<listPerson>` catalogue in the `<teiHeader>` which uses the first occurrence of each distinct name. We will generate a TEI XML output file containing just the catalogue. Here are the steps to take

- add `xmlns="http://www.tei-c.org"` to the outermost `<stylesheet>`, to say you want the output elements to be in the TEI namespace.

- Add `<xsl:output method="xml" indent="yes"/>` above the first template, to make the output look prettier

- in the template for `<TEI>`, create a `<listPerson>` element, and then put in the code you used in the last exercise for the name index. This time, generate something like this:

```xml
<person
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:tei="http://www.tei-c.org/ns/1.0">
  <persName>Joseph</persName>
</person>
```

- But you should also add a unique ID to each element. One way of doing this is to use the `translate()` function to remove spaces, and make everything uppercase:
but we should also remove ””, “-” and “:” characters. Unfortunately we cannot say

```
upper-case(translate(normalize-space(.),' -"',"'))
```

as the the multiple single quotes are not allowed, but this

```
<xsl:variable name="Lose"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
 xmlns:tei="http://www.tei-c.org/ns/1.0">."'</xsl:variable>
...
```

does work. Note that we also use `normalize-space()` to make line endings also get treated as spaces.