hocl.tk Documentation

Release alpha0.5

hocl.tk

January 20, 2015

Contents

Contents:

CHAPTER 1

Documentation for hocl.tk

This documentation is composed in reStructuredText and is hosted on Read the Docs. You can also visit the blog at http://hocl.tk which has some information about how hocl.tk has been deployed.

1.1 Adding documentation

Documentation for hocl.tk is housed in two locations:

1. In the top-level project directory as `README.md`.
2. As a Sphinx project under the `docs` directory

To add a page about supported TEI document elements to the documentation, include an entry in the list under `toctree` in `docs/index.rst` like:

```
tei-elements
```

and create the file `tei-elements.rst` under the `docs` directory and add a line:

```
.. _tei-elements:
```

to the top of your file, remembering to leave an empty line before the rest of your text.

You can get a preview of what your documentation will look like when it is published by running `sphinx-build` on the docs directory:

```
sphinx-build -w sphinx-errors docs <build_destination>
```

The docs will be compiled to html which you can view by pointing your web browser at `<build_destination>/index.html`. If you want to view the documentation locally with the ReadTheDocs theme you'll need to download and install it.

1.2 API Documentation

Documentation is included in Java source files as Javadoc comments (the ones that start with `/**`) and can be built locally with the Maven `javadoc:javadoc` goal:

```
mvn javadoc:javadoc
```

1.3 Substitutions

Project-wide substitutions can be (conservatively!) added to allow for easily changing a value over all of the documentation. Currently defined substitutions can be found in `docs/conf.py` in the `rst_epilog` setting. More about substitutions is provided in the reStructuredText documentation.

1.4 Conventions

If you'd like to add a convention, list it here and start using it.

Currently there are no real conventions to follow for documentation style, but additions to the docs will be subject to style and content review by project maintainers.

CHAPTER 2

Contributing to hocl.tk

2.1 XSLT Recommendations

This project is largely composed of XSLT documents. The most useful and authoritative reference for XSLT is the W3C XSL Transformations recommendation. Below is a list of recommendations for authoring XSLT documents (called "stylesheets") in this project.

2.1.1 Testing and Debugging

I strongly recommend using a command line tool, or a simple GUI if you prefer, to develop and debug stylesheets. Re-deploying the application just to test simple changes to a stylesheet is huge waste of time. As long as you make sure any extensions you use are supported by your command line tool and the Java XSLT library (they may even be the same), then you shouldn't have any problems. I use xsltproc for this purpose.

2.2 HTML Recommendations

For HTML used in the project, try to make the pages readable without extensive use of Javascript and CSS. In general, it is better to support a wide range of readers, some of which may not be modern personal computers or which may have Javascript disabled or which just display differently. For example, the table format of the poem presented by the `TEIHtml` transformation still holds up with or without the Javascript.

2.2.1 Naming

These rules help to keep the relationships between resources and code from becoming obscured. They do not, however, affect functionality.

The stylesheet should have a base name, (i.e., file name, excluding the `.xslt` extension), matching the last element of the associated URL path. The URL path is determined either by the `WebServlet` annotation on a `XSLTTransformer` subclass or in the web.xml file. In addition, the main stylesheet associated with an `XSLTTransformer` should have a file name that corresponds to the `XSLTTransformer` subclass name like in these examples:

```
tei-html.xslt --> TEIHtml
table-to-tei.xslt --> TableToTEI
include-author-date.xslt --> IncludeAuthorDate
add-ids.xslt --> AddIDs
```

In other words:

- remove the hyphens,
- capitalize individual words,
- and for abbreviations that are typically in all caps, put them in all caps in the class name, but not in the stylesheet name.

For a stylesheet which takes parameters, a particular `XSLTTransformer` may be a parameterized version of that stylesheet. In this case, the name of the `XSLTTransformer` does not need to have the correspondence with the stylesheet like that described above.

2.3 Adding transformations

To add a document transformation, you have to do three things:

1. Make the stylesheet and put it in the resources folder.
2. Make the servlet corresponding to the stylesheet and put it in the controllers folder.
3. Set the URL pattern for the servlet either using the WebServlet annotation (servlet API version >= 3.0) or in the web.xml file.

2.3.1 Source code

See `src/main/java/controllers` for the Java web servlets and the transformation. The XSLT files are in `src/main/resources` and get called by the sub classes of XSLTTransformer.

Javadoc comments should be added to every class and to private members as well as public as these are for internal documentation.

Subclasses of XSLTTransformer should only include documentation for the transformation as whole, and not refer to implementation. As an exception, special use of URL query parameters beyond the default name-based usage (e.g., URL parameter 'g' corresponds to transform parameter 'g'), should be documented on the subclass.

2.3.2 Specifications

In describing requirements, please utilize the keywords defined in RFC 2119 .

2.4 Things to do

A longer list of to-dos and considerations is given in the TODO file in the root of the hocl.tk project directory. The list below is intended to highlight immediate needs for the project that may or may not be listed in the TODO file or as Github issues (yet!).

1. Make a Wordpress plugin for showing the "Regiment of Princes" time referents in a concordance table on the Hoccleve Archive Wordpress site.
2. Write more transformations for creating CTables and HTML documents from TEI. See issue #11.
3. Harden server against abuses from users. Some problems to start on are listed on Github.

CHAPTER 3

Supported TEI markup

The TEI standard is large and composed of many modules, not all of which are required or expected to be used in a given project. The TEI Consortium recommends for users of TEI that they define a subset of TEI markup which is supported by their application. For our case, the important thing is to be able to identify when there is an error in an XSL Transformation (XSLT) and to tell the user what that error is. For performing an XSL transformation, it is not required to match a schema exactly as long as we can process the document, so writing a schema for the subset of TEI which we support is unnecessary. Focus should, instead, be placed on generating appropriate error messages and warnings in the XSLT.

This is the documentation for the Hoccleve Archive concordance table subproject. The project is hosted on github as hocl.tk. This project uses a combination of XSLT, Java Server Pages, and Java WebServlets to modify and present documents composed in TEI XML markup.