TEI Annotator User Guide

Dan McCreary <dan@danmccreary.com>

Claudius Teodorescu <claudius.teodorescu@gmail.com>

Joseph Wicentowski <joewiz@gmail.com>

Revision History

Revision 1 2010-05-01

This document was created by Dan McCreary for Dr. Joseph Wicentowski (under funding from U.S. Department of State, Office of the Historian) for use in annotating TEI documents with the Office of the Historian's eXist-based content management system (also created by McCreary). The JavaScript code for the "tei-ann" CKEditor plugin was created by Claudius Teodorescu based on his work extending XSLTForms, while McCreary wrote all other aspects of the application. As stipulated in the Office of the Historian's original statement of work, all source code and documentation for the project was to be released to the community when complete.

Revision 2 2010-12-29

McCreary converted the document from MS Word into DocBook and expanded it with additional documentation, for the purpose of sharing with the University of Maryland Institute for Technology in the Humanities (MITH) Code Camp Project. Doug Reside, Associate Director was the coordinator of this project. The new demos were created with much help from Teodorescu. McCreary also added new sections on background and terminology. Wicentowski edited the document.

Table of Contents

Overview

The "TEI Annotator" XRX demonstration and testing application was originally created to fill a gap in the TEI community as there was no adequate browser-based system for editing or annotating TEI documents. Several goals guided this project. Given that different TEI projects have different customizations of the TEI schema and have different software requirements, our goal was to create a system that was configurable for different project's needs. Additionally, because many TEI projects have limited budgets and few technical support staff, it was important to make the configuration possible using simple XML files that do not require extensive JavaScript code. We also hope to build simple web-based interfaces to the XML files to lower the barrier even more. Most of all, we felt it was important to use open W3C standards such as XForms and to find and integrate reliable components such as CKEditor that will be supported in the future. The authors of this software have spent considerable time studying how best to empower non-programmers, and we designed the TEI Annotator with this in mind.

This User Guide is designed as a companion to the "TEI Annotator" XRX demonstration and testing application. It was written to help others quickly come up to speed on the overall structure of this XRX application as well as the "tei-ann" CKEditor annotator plugin that it uses to annotate TEI documents.

Assumptions

This guide assumes the reader is already somewhat familiar with XML and TEI. If you do not have a strong background in these areas it is recommended you search for tutorials on the subjects. We also makes assumptions in the examples about the use of XForms for testing the tei-ann component. Our goal is to minimize the requirements to understand XForms other than to setup basic test programs for loading and saving TEI content. If you are not familiar with XForms we suggest the XForms Tutorial and Cookbook [http://en.wikibooks.org/wiki/XForms]. You may specifically want to view the examples that use the textarea [http://en.wikibooks.org/wiki/XForms/Textarea] example. If you would like to do full round-trip Create, Read, Update and Deletes of TEI data we also recommend the Beginner's Guide to XRX.

To use this guide we strongly recommend using a native XML database such as eXist and an XML editing tool such as the oXygen XML editor. These tools are not required but will make your development much easier.

Terminology

The TEI-Annotator web application is a testing and demonstration application of the TEI annotator (tei-ann) plugin of the CKEditor [http://ckeditor.com/]. This application is designated to demonstrate the many features and functions of the tei-ann plugin. In this document we use the following terms:

- **XForms** - a W3C standard for allowing users to design web forms without the need for JavaScript. XForms is a general specification of the semantics of XML tags used in the web application development process, not a single software package.
- **XSLTForms** - an implementation of the XForms package that allows simple XML to be loaded into the web browser as a model.
- **eXSLTForms** - a set of extensions to the XSLTForms package written by Teodorescu that allows for custom user interface controls such as rich text browsers.
- **textarea** - a region of a web page used for editing large multi-line blocks of text. This system uses the XForms textarea component to perform in-line editing of TEI elements.
- **CKEditor** - a JavaScript library that allows users to perform rich-text editing within a web browser. eXSLTForms binds CKEditor to a XForms text area. CKEditor has been around for a long time and has a large and active open source community behind it. CKEditor also has a very robust plugin architecture that made it the ideal choice for this project.
- **tei-ann** - the name of the CKEditor plugin that does the actual TEI annotations. tei-ann is composed of a set of "TEI Annotators" that can be conditionally added to the tei-ann toolbar.
- **TEI Annotator** - the component of tei-ann that performs a specific annotation function on the text. There is usually one annotator per TEI element such as person. The number of annotators used in a given context will be determined by the rule you set up in your application.
- **TEI Annotator Panels** - These are small dialog screens that pop up within your web browser when some annotators are selected. For example if you select a person annotator the Person Dialog Panel may allow you to select from a list of known persons for each document.
- **TEI Annotator XRX Application** - The XRX application used to test the tei-ann functions. This includes the viewers, reports and the unit testing tools as well as this User Guide. Note the tei-ann CKEditor plugin does not require this to run. It is only used in the development and testing process.

- **SCAYT** - Spell Check As You Type. The option of CKEditor that enables spell checking as new text is entered into a textarea. Many of the configuration options reference this term.
- **Encoded XML** - Because web browsers are parsing all incoming HTML tags, we take special precautions to prevent the browser parser from treating any TEI content as part of the web page. The way we do this is to encode all of the elements using standard XML escape characters for angle brackets (e.g. instead of "<" we use an "<"). This encoding process makes the files very difficult for humans to read, but it is the only way we can load text into CKEditor. It is also difficult to encode these examples within DocBook.

Roles in TEI Annotator Project

This guide attempts to address several distinct roles in a TEI annotation project. Some of the following roles might include:

- **TEI Annotator End Users** - Individuals who will be adding annotations to the actual TEI documents. Our goal is to shield these users from the complexities of the system.
- **TEI Web Site Administrators** - Individuals who will set up the TEI web site. These users will need to be able to setup a system such as eXist and install the TEI Annotator XRX application and configure the TEI annotator toolbars to meet site-specific annotation functions. These uses will need to be able to edit the TEI Annotator XML configuration files but should not be required to understand JavaScript, although in some cases the editing of JSON files will be needed. Our goal is to eventually allow these users to configure the site using only XForms.
- **TEI Annotator Developer** - These are experienced HTML and JavaScript developers who build or extend the individual TEI annotator components. Much of this can be done by adding new XML content to the Annotator-Specification file however, knowledge of HTML forms is needed to create custom model panels.

TEI-Annotator XRX Application Structure

The TEI-Annotator XRX application has several directories and files the following is a summary:

- utils - This contains the software components that are used together to make the tei-ann work.
- utils/xsltforms - The XSLTForms package.
- utils/ckeditor - The CKEditor package.
- utils/exsltforms - The eXSLTForms package. This package allows you to bind the CKEditor to the XForms textarea.
- unit-tests - A series of unit tests.
- views - Transforms of the tei-ann configuration files.
- edit - Tools that will edit the data sets including save and update functions for both configuration files and sample TEI documents.

tei-ann Structure

Within the TEI Annotator is an instance of the tei-ann CKEditor plugin. The following are the critical directories and files in this plugin:

- utils/ckeditor - The CKEditor package.
- utils/ckeditor/plugins - The CKEditor plugins directory.
- utils/ckeditor/plugins/tei-ann - The tei-ann plugin directory.

- utils/ckeditor/plugins/tei-ann/config - The tei-ann plugin configuration directory.
- utils/ckeditor/plugins/tei-ann/config/Annotator-Specifications.xml [../utils/ckeditor/plugins/tei-ann/config/Annotator-Specifications.xml] - The tei-ann plugin configuration file.
- utils/ckeditor/plugins/tei-ann/config/generalConfigOptions.xml [../utils/ckeditor/plugins/tei-ann/config/generalConfigOptions.xml] - General configuration options including where to load the specification file and the CSS file.
- utils/ckeditor/plugins/tei-ann/config/lang/en.xml [../utils/ckeditor/plugins/tei-ann/config/lang/en.xml] - A file of English Language labels used when you hover over the buttons.

Overview of XForms textarea

The following is an example of a typical XForms file:

```
<html
   xmlns="http://www.w3.org/1999/xhtml"
   xmlns:xf="http://www.w3.org/2002/xforms">
   <head>
      <title>XForms textarea</title>
      <xf:model>
         <xf:instance xmlns="">
            <data>
               <MyTextElement>Hello World!</MyTextElement>
            </data>
         </xf:instance>
      </xf:model>
   </head>
   <body>
      <xf:textarea ref="MyTextElement">
         <xf:label>My Text:</xf:label>
      </xf:textarea>
   </body>
</html>
```

Note that the model includes an instance of an XML document. In the example above, the instance is in the null namespace. Within this instance is a root data element, and then an element called MyTextElement. This is the actual XML node that will be edited in the text area. The instance is bound to the textarea using the ref attribute.

The standard XForms textarea is customized by adding attributes to the textarea element and additional content within the textarea body, usually following the label. The following is an example of this:

```
<xf:textarea ref="MyTextElement" myCustomAttribute="myCustomValue">
   <xf:label>TEI Content:</xf:label>
   ... add custom attributes here...
</xf:textarea>
```

This is the key structure that we will be manipulating in the following examples.

Overview of tei-ann CKEditor plugin

Because the CKEditor plugin operates on HTML text, the **tei-ann** CKEditor plugin transforms the text between its source TEI form into HTML elements. For example, the TEI <persName corresp="#HAK"> element is transformed into the HTML . The conversion is round-trippable and lossless. Even when viewing the HTML

form of the data, a user can easily recognize TEI elements already presented in text and annotate the text using new TEI elements. The plugin can be easily extended in order to use new TEI elements.

Customizing the textarea

In the above example **ref** is a XPath expression to the element within the model that holds the TEI text to be annotated. The label in the screen label for on the screen. The following changes need to be made to your XForms to use the TEI annotator.

```
<xf:textarea ref="myTEItext" appearance="exfk:CKEditor">
   <xf:label>TEI Content:</xf:label>
   <xf:extension>
       <exfk:rteOptions>
          {
             skin:'office2003',
             width: 1000,
             height: 300,
             extraPlugins : 'tei-ann',
             toolbar: [[ 'teiannBoldBtn','teiannPersonBtn']]
           }
        </exfk:rteOptions>
     </xf:extension>
</xf:textarea>
```

What the above is indicating is that the textarea includes an extension. This extension has several options for the Rich Text Editor (exfk:rteOptions). In this case the options include the skin and layout of the CKEditor (one of several rich-text editors that can be added to XForms), and it also instructs the CKEditor to load an extra plugin which is the TEI editor. This plugin also allows you to specify the tei-ann toolbar additions, for example, for Bold Text and Person.

Customizing the tei-ann Toolbar

The TEI Annotator toolbar can be customized by adding structures to the JSON structures within the toolbar elements within the textarea of the XForms application. A complete list of all the annotators available in your configuration can be found by running the list-annotators.xq [views/list-annotators.xq] script in the views collection.

The following is a full listing of the TEI Annotators in the current demo system:

```
[['teiannBoldBtn', 'teiannCellBtn', 'teiannDateBtn', 'teiannEditEntityBtn',
'teiannGeoLocationBtn', 'teiannGlossaryBtn', 'teiannHeadBtn', 'teiannHyperlinkB
'teiannItalicBtn', 'teiannItemBtn', 'teiannLineBreakBtn', 'teiannListBtn',
'teiannPageBreakBtn', 'teiannParagraphBtn', 'teiannPersonBtn', 'teiannRemoveEnt
'teiannRoleBtn', 'teiannStrikethroughBtn', 'teiannTableBtn', 'teiannUnderlineBt
'Source' ]]
```

Note that you must put two square brackets around this list. The last line also adds the "View Source" button to allow users to see the source code view of the HTML encoded TEI tags.

Customizing the Rich Text Editing Component

There are several options of the overall CKEditor configuration options that you may want to consider when enabling textarea editing. These can be added directly to the JSON components within the textarea.

- **toolbarCanCollapse** (true, false) - This tell the toolbar if it can be "collapsed" to save screen area. If set to true a small upward pointing triangle icon will be visible to the right of the toolbar. Selecting this will collapse the toolbar. This option is important when you have a large toolbar and your users just want a large text area but do not need the toolbar buttons. Note that control and function keys can always be used, even if the toolbar is not visible. The toolbar can always be expanded.
- **entities**: (true, false) - This tells the toolbar if entities can be edited. (TODO - need more info)
- **scayt_autoStartup** :(true, false) - Tells the system if the Spell-Check-As-You-Type (SCAYT) should scan all the text on startup. The default value is true..
- **disableNativeSpellChecker**:(true, false) - Indicates if the in-browser spell check should be enabled or not. The default value is false so by default spell checking is enabled.

Unfortunately the CKEditor documentation, although auto-generated from the JavaScript source, is not very helpful. Some additional hints may be gleaned from the CKEditor JavaScript API [http://docs.cksource.com/ckeditor_api/symbols/CKEDITOR.config.html].

Typical TEI Annotation Project Tasks

The following is a list of typical project steps used in a TEI annotation project and how these tools might fit into this project.

1. Download the eXist native XML database
2. Load the TEI Annotator XRX application
3. Run the unit tests to make sure all the configuration files are setup correctly
4. Customize the style module to include your site-wide header and footers
5. Customize the site-wide style.css CSS style sheet
6. Load your TEI data into a data collection for example /db/project/apps/tei/data
7. Write an XQuery program that lists the TEI documents in your collection and another that lists all the items you want to edit in the document.
8. View the TEI documents and adds an "Edit" button to the structures you want to edit such as divs or paragraphs
9. Pass the paragraph or div ID to an XQuery that generates an XForm that edits the paragraph
10. Connect a "Save" button to a script that takes the HTTP POST data and stores it in your database