# Packing TEI CE Toolbox into Oxygen framework

*Magdalena Turska* *2014-11-24*

## 1 TEI Critical Edition Toolbox for oXygen

TEI CE Toolbox addition to oXygen framework packages Marjorie Burghart's on-line tool that offers facilities to visualize critical edition while it is still in the making, and check the consistency of the encoding. The TEI Critical Edition Toolbox can help to check editions encoded in TEI with the parallel-segmentation method. Application offers several 'sanity checks', eg. if using a "positive" apparatus, listing all the readings of all the manuscripts in the @wit attributes of the `<lem>` and/or `<rdg>`, all apparatus entries that do not use all the witnesses listed in a `<listWit>` in the header can be detected. The part of the existing tool that dealt with downloading and transformation of the file (originally in php) has been rewritten as ANT transformation script and everything wrapped into an oXygen action to seamlessly work in oXygen environment.

## 2 TEI Critical Edition Toolbox

- TEI Critical Edition Toolbox is available at http://ciham-digital.huma-num.fr/teitoolbox/
- authored by Marjorie Burghart, MA, MSc, PhD EHESS - CIHAM UMR 5648, Lyon

'The TEI Critical Edition Toolbox is a simple tool offering an easy visualization for TEI XML critical editions. It especially targets the needs of people working on natively digital editions. Its main purpose is to provide editors with an easy way of visualizing their ongoing work before it is finalized, and also to perform some automatic quality checks on their encoding.'

## 3 How does it work?

CE Toolbox requires user to upload a file for it to be converted and presented as html page. What I wanted to do was to pack it into TEI oXygen framework, so all the user needs to do is hit a button while editing the file and get the results immediately - no need to upload the file after every change. The necessary steps were to isolate the relevant bits from php-based TEI Critical Edition Toolbox: the xslts used to convert the original TEI source to feed into the BP, the CSS files and JS libraries. I had to add an ANT transformation scenario in oXygen, enable it in TEI framework and afterwards create custom oXygen action that uses the transformation set up earlier and a button that triggers it to actually perform the operation. End result is a newly generated TEI CE Toolbox page at a click of a button in oXygen's author mode.

## 4 How to do it in oXygen?

- Isolate the necessary resources: XSLT and js scripts, CSS stylesheets (teicetoolbox subdirectory)
- Create ANT build script (build.xml)
- Create oXygen transformation scenario based on the ANT script from previous step
- Create oXygen custom action and button that triggers the action

## 5 Step 1: isolate the resources

TEI CE Toolbox is built upon TEI BoilerPlate. Thus, to work it requires a set of XSLT stylesheets that perform the transformation. The output page then relies on CSS stylesheets

for formatting and JS scripts that enable clicking on and off various features. These resources had to be copied somewhere where oXygen will have access to them. In this example I added subdirectory called teicetoolbox to tei framework directory.

## 6 Step 2: build script

Again, original Toolbox was running in a server environment and used php script to deliver it's interface that allowed users to upload the file and run the transformation. In oXygen there's no need for uploading but we still need to set up transformation parameters.

To this end I created an ANT build script that does the magic. The script accepts a few parameters like input file, output directory etc. Then it runs the XSL transformation, creates new file in the output directory and copies necessary CSS and JS resources there as well.

Here's the relevant bit from the ANT script:

```
<target name="trnsfrm" depends="clean">
 <echo>Make transformation</echo>
 <xslt style="content/custom.xsl"
  in="${inputFile}" out="${outputDir}/${outputFileName}">
  <factory name="net.sf.saxon.TransformerFactoryImpl"/>
  <classpath location="${oxygenlib}/saxon9he.jar"/>
 </xslt>
 <copy todir="${outputDir}">
  <fileset dir=".">
   <include name="css/**"/>
  </fileset>
 </copy>
 <copy todir="${outputDir}">
  <fileset dir=".">
   <include name="js/**"/>
  </fileset>
 </copy>
</target>
```

## 7 Step 3: oXygen transformation scenario

To create new transformation scenario I needed to go to Options > Preferences > Document Type Association and then select framework I work on, then click Edit. In the Document type dialog I chose Transformation tab and clicked + icon to add new transformation selecting ANT Transformation from the drop-down list

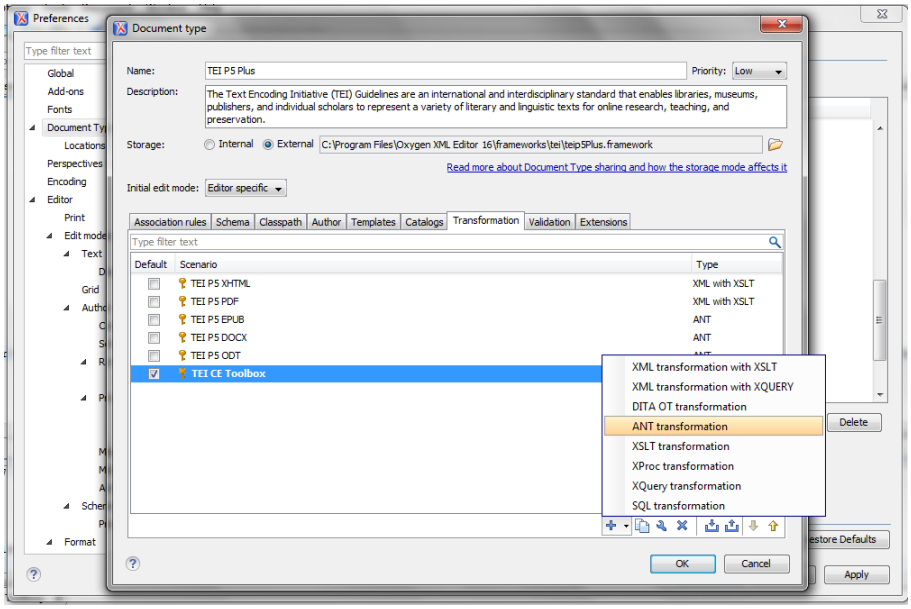
In the Edit Ant Scenario dialog I needed to set up scenario parameters: name of the scenario (TEI CE Toolbox), working directory and the location of the build file in the Options tab, parameters for the build script in the Parameters tab and directives what to do with the output in the Output tab.

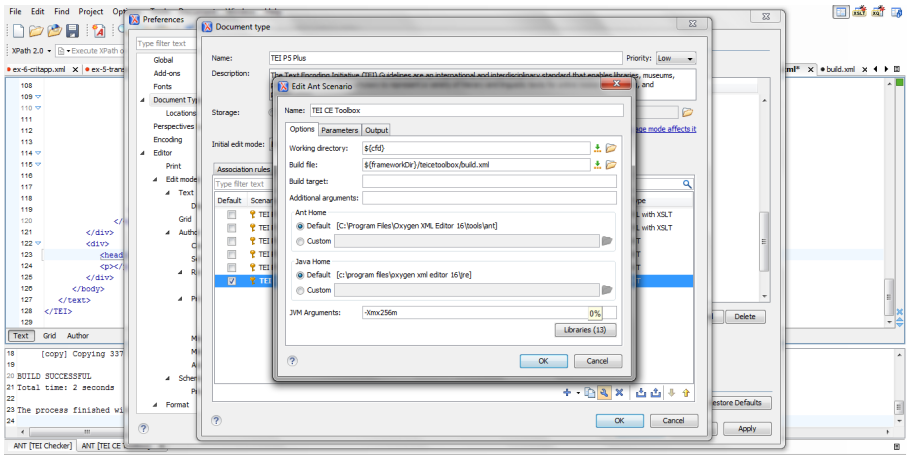
After this is complete oXygen will offer TEI CE Toolbox transformation scenario as one of the default scenarios for the selected framework (and thus for the documents matching that framework).

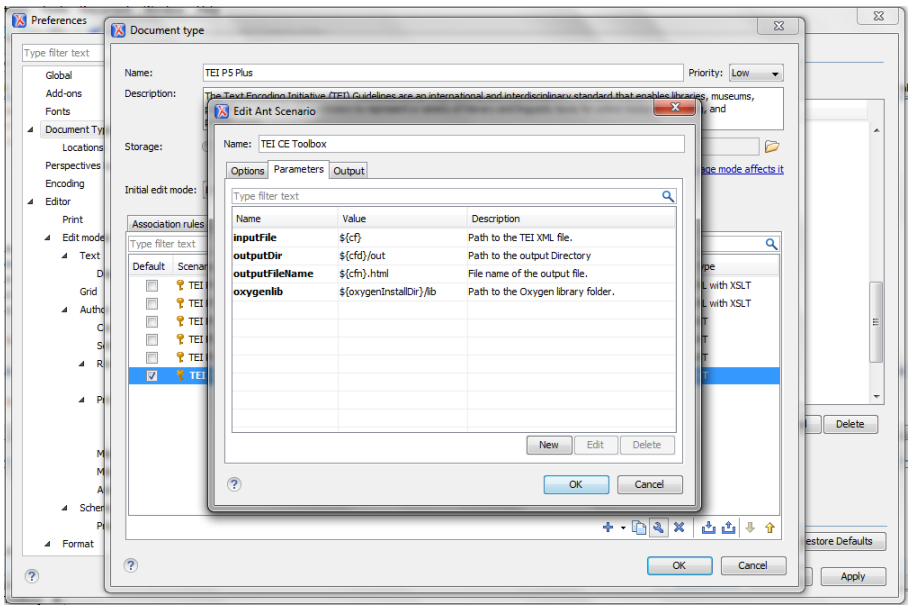
## 8 Step 4: custom oXygen action

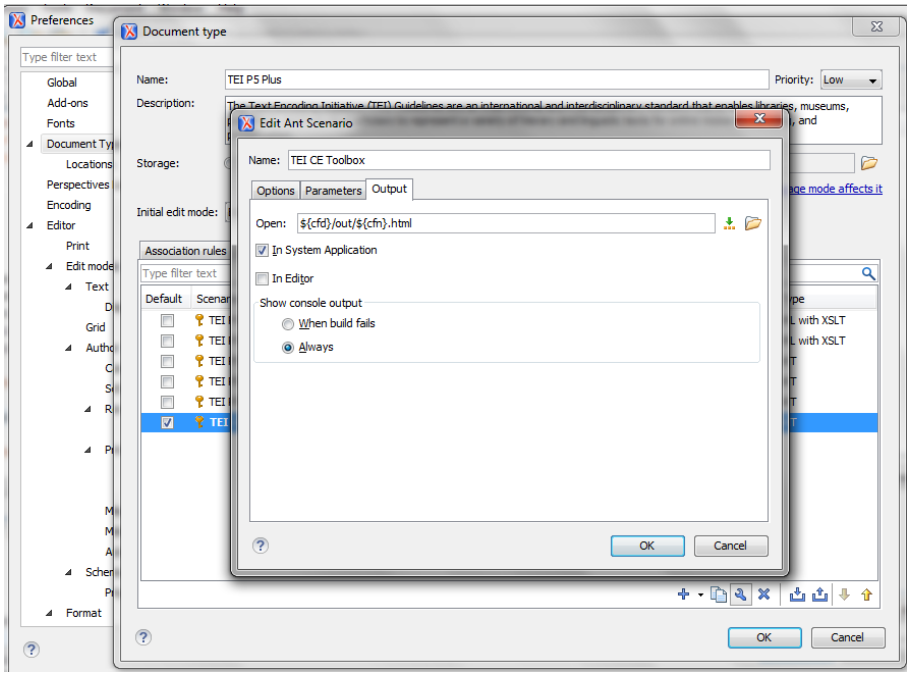
In the Document type dialog (like previously from the Option > Preferences > Document Type Association) we need to choose the Author tab to configure actions and custom buttons for the Author mode.

First in the Action sub-tab choose the new action and give it ID (TEICEToolbox in my case) and assign the operation to trigger (invoke the operation). In this case I used the built-in oXygen operation called ro.sync.ecss.extensions.commons.operations.ExecuteTransformationScenariosOperation that can 'Run a named transformation scenario defined in the associated document type.'

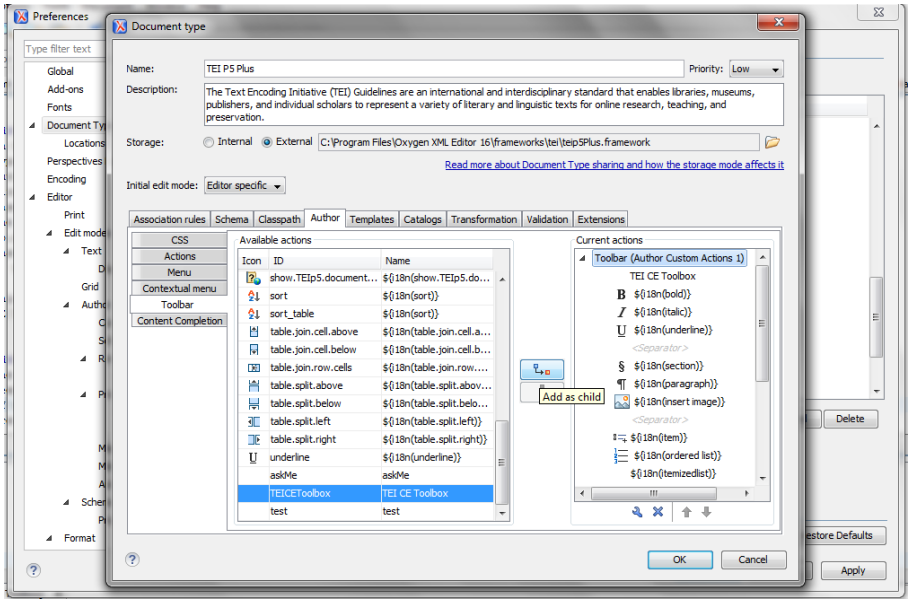

Preferences
Document type
Name:
TEI P5 Plus
Priority:
Low
Description:
The Text Encoding Initiative (TEI) Guidelines are an international and interdisciplinary standard that enables libraries, museums, publishers, and individual scholars to represent a variety of literary and linguistic texts for online research, teaching, and preservation.
Storage:
Internal
External
C:\Program Files\Oxygen XML Editor 16\frameworks\tei\teip5plus.framework
Read more about Document Type sharing and how the storage mode affects it
Initial edit mode:
Editor specific
Association rules
Schema
Classpath
Author
Templates
Catalogs
Transformation
Validation
Extensions
Default
Scenario
Type
TEI P5 XHTML
XML with XSLT
TEI P5 PDF
XML with XSLT
TEI P5 EPUB
ANT
TEI P5 DOCX
ANT
TEI P5 ODT
TEI CE Toolbox
XML transformation with XSLT
XML transformation with XQUERY
DITA OT transformation
ANT transformation
XSLT transformation
XProc transformation
XQuery transformation
SQL transformation
Delete
OK
Cancel
Apply


Preferences
Document type
Edit Ant Scenario
Name:
TEI CE Toolbox
Options
Parameters
Output
Working directory:
${cfd}
Build file:
${frameworkDir}/teicetoolbox/build.xml
Build target:
Additional arguments:
Ant Home
Default
Custom
Java Home
JVM Arguments:
-Xmx256m
Libraries (13)
OK
Cancel


Preferences
Document type
Edit Ant Scenario
Name:
TEI CE Toolbox
Options
Parameters
Output
Name
Value
Description
inputFile
${cf}
Path to the TEI XML file.
outputDir
${cfd}/out
Path to the output Directory
outputFileName
${cfn}.html
File name of the output file.
oxygenlib
${oxygenInstallDir}/lib
Path to the Oxygen library folder.
New
Edit
Delete
OK
Cancel



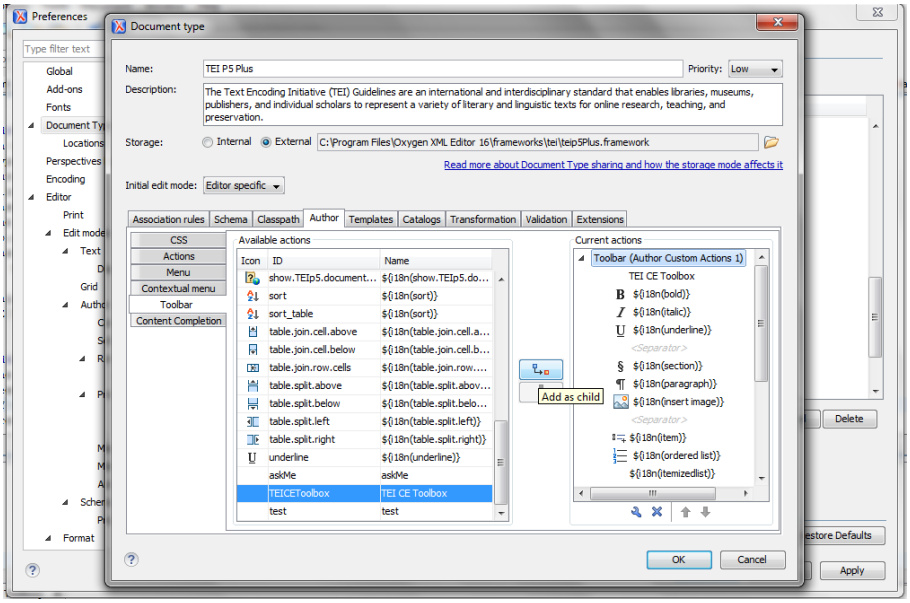

This operation only uses one parameter, the name of the scenario, so I choose my TEI CE Toolbox defined earlier in the arguments section.




# 9 Step 5: custom Author mode button

The previous step only prepares the custom action. To invoke it in the Author mode we need a button or menu option. To create a button we'll work again in the Author tab of the Document type dialog but this time with the Toolbar section of it.

The list of actions now should include our newly created TEICEToolbox action so it's only a question of adding it to the toolbar to the right via Add as a child.




# 10 The happy end

With all this in place now your oXygen when used in Author mode should present you with a nice button that says TEI CE Toolbox that, when clicked, opens a web browser with CE Toolbox version of your current file! Happy editing!

If something goes amiss, check this out:

- do you work in Author mode? custom actions have no effect on Text mode behaviour
- does the framework you edited fit the document you are working with?
- does this framework have highest priority from all frameworks that fit the document you are editing?