Tootomatic with Java 2 v1.4.2_05, and a Self-test feature

Created with Toot-O-Matic

www.mittineague.com/j2v1-4-2

Table of Contents

If you're viewing this document online, you can click any of the topics below to link directly to that section.

Section 1. Introduction

Background Information

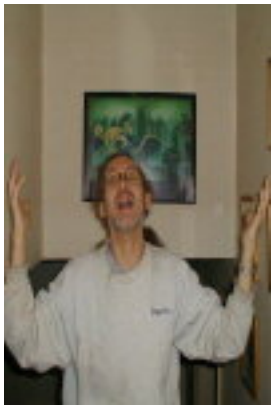
***** NOTE *****These pages use the tootomatic templates but are at www.mittineague.com and are NOT at developerWorks

From the IBM developerWorks' "Building Tutorials with the Toot-O-Matic" tutorial at *https://www6.software.ibm.com/developerworks/education/x-tom/index.html* , an "x-tom" version pdf file, and the "tootomatic.zip" and "jars.zip" files were downloaded.

The zip files were expanded with the Aladdin StuffIt Expander, and the contents were placed into a common folder.

The zip file contained Tootomatic v3.02. The application was run with Java 2 SE v1.4.2_05

Section 2. Problems



Exceptions, Corruption, and Non-functionality

Running the tootomatic application from the MS-DOS prompt gave the following XSLT Error:

(javax.xml.transform.TransformerException): java.lang.ClassCastException

Just the same, the application did create a folder and files.

The Zip file was small in size (4KB) and could not be expanded.

The Feedback form was non-functional.

The E-mail It form was also non-functional.

In addition, and of no consequence, there was an extra index.html file outside of the generated folder. And the folder contained 2 empty folders, "i" and "master"

Also, and of potentially more concern, the main Index page gave 22 W3C Validation Errors. Correcting the application so that it creates valid pages is beyond the scope of this discussion.

Section 3. Fixing the Problems

Eliminating Thrown Exceptions

Changes to tootomatic.xsl

COMMENT OUT

```
<!--

<xsl:apply-templates select="tutorial"
mode="generate-zip-file"/>

-->
```

This will allow the application to run without throwing exceptions.

However, No zip file will be created. If a zip file is neccessary, java files other than the ones supplied with the downloaded files are required, or another utility can be used to create it.

Image-Links to Zip file

Changes to dw-style.xsl

COMMENT OUT

```
<!--

<a border="0">

<xsl:attribute name="href">

<xsl:value-of select="concat($fn, '.zip')"/>

</xsl:attribute>

<img src="../{$images-directory}/i/icon-zip.gif" border="0"
alt="ZIP"/>

</a>

-->
```

If this file is not changed the non-functional zip image-link will be present.

Alternatively, the zip file could be created with another utility and the zip image-link could be used after either naming the zip file the same name as the tutorial's filename attribute value, or by changing the xsl:value-of select's value to the files name.

Feedback Form Functionality

Changes to build-individual-panels.xsl

COMMENT OUT

```
<!--

<xsl:template match="feedback-form"
mode="build-individual-panels">

<xsl:for-each select="$feedback-form">

<xsl:copy-of select="."/>

</xsl:for-each>

</xsl:template>

-->
```

ADD

```
<xsl:template match="feedback-form"
mode="build-individual-panels">

<form action="{@action-url}" method="post">

<input type="hidden" name="RedirectURL" value="{@redirect-url}"
/>

<input type="hidden" name="zone" value="{//tutorial/@zone}" />

<input type="hidden" name="OnlineCourseTitle"
value="{//tutorial/title}" />

<xsl:for-each select="$feedback-form">

<xsl:copy-of select="."/>

</xsl:for-each>

</form>

</xsl:template>
```

The tutorial's feedback page that is online utilizes the feedback-form elements's "action-url" and "redirect-url" attributes. As created by tootomatic, however, the page's form tags are absent. The online page also has the 2 hidden inputs, "zone" and "OnlineCourseTitle". The above changes will create those tags on the feedback page. Depending on what is needed for information, other attributes could be added to the feedback.html file, and/or these other 2 hidden inputs could be left out or ignored and left unused.

A detailed chapter dealing with customization is found in the "t-o-m" version of the tutorial that is created from the downloaded tootomatic.xml file.

***** NOTE ***** Using this modification Without changing the feed-back element's action-url attribute will submit the form to developerWorks.

Please do NOT do that, (unless it should be, of course)

Avoid the "white-space" problems involved with copying the above code and Cut and Paste from the zipped Text files. The modified xsl:template code for the build-individual-panels.xsl file is included in the *File Modifications* folder

---

E-mail It Form Functionality - Considerations

1. Javascript Dependency
2. Pop-up Blocking
3. Email Filtering
4. Spamming Prevention

This feature requires that the browser has Javascript enabled. Accommodating browsers without Javascript would involve quite a bit of work, and is beyond the scope of this discussion.

This feature requires that the browser allows Pop-ups. Providing an alternative panel is also beyond the scope of this discussion.

Because of Spam, many email programs utilize filtering. Although it is presumed that users would only be emailing known contacts, the receiving address(es) would have to accept email from the entered fromEmail address.

It is important to evaluate security issues. On the IBM developerWorks' site, registration is required to access the tutorial and it's e-mail it form. Although the form provides for multiple recipients, it's use is thereby limited.

However, Note that the tutorial's e-mail it form that is online does not work either!

E-mail It Form Functionality - Not Using it

Changes to dw-style.xsl

COMMENT OUT

```
<!--
<a border="0" href="javascript:newWindow()">
<img src="../{$images-directory}/i/icon-email.gif" border="0" alt="e-mail"/>
</a>
```

```
-->
```

Although not used, the emailfriend.js file will still be referenced unless commented out in the build-main-index.xsl, build-section-indexes.xsl (twice), and build-individual-panels.xsl files.

```
<!--
<script language="javascript"
src="../{$images-directory}/i/emailfriend.js"/>
-->
```

E-mail It Form Functionality - Using it

Changes to emailfriend.js

The Javascript file creates a form tag with an action attribute's value containing the url of an IBM server file. This needs to be changed to point to a working file.

- The form also uses the following inputs:
  - body - from the tutorial's abstract attribute
  - subject - from the tutorial's title element
  - url - from the tutorial's email-link attribute
  - email - from the form's Send to: input
  - fromName - from the form's Your name: input
  - fromEmail - from the form's Your e-mail address: input
  - comments - from the form's Comments: input

If the email server file in the form's action attribute does not handle multiple addressees the text should be changed to indicate such.

Suggestions:

Modify the pop-up's text by changing the output variable here:

```
output += "<td colspan='2'><font FACE='HELVETICA, HELV, ARIAL'
SIZE='-1'>Share this developerWorks content with others who you
think will find it interesting, useful, or even amusing. Be sure
to separate multiple e-mail addresses with a comma.</font><br
/><br /></td>";
```

Add scrollbars to the pop-up:

```
emailWindow =
window.open("","subwindow","HEIGHT=500,WIDTH=600,resizable=yes,scrollbars=yes");
```

Provide clues indicating the use of Javascript and the pop-up by changing the alt attribute in the **dw-style.xsl** file:

```
<a border="0" href="javascript:newWindow()">

<img src="../{$images-directory}/i/icon-email.gif" border="0"
alt="Javascript e-mail it pop-up"/>

</a>
```

Section 4. Self-Test Feature

Adding Self-Test xsl:template

to the build-individual-panels.xsl file, ADD

```
<xsl:template match="self-test" mode="build-individual-panels" >
<script type='text/javascript'>
<![CDATA[
<!--
function replaceSpan]]><xsl:value-of
select='@span'/><![CDATA[(displayedText,spanNumber){
var newSpan = document.createElement("span");
var newText = document.createTextNode(displayedText);
newSpan.appendChild(newText);
var para = document.getElementById("area"+spanNumber);
var spanElm = document.getElementsByTagName("span")[spanNumber];
var replaced = para.replaceChild(newSpan,spanElm);
}
var testtext = "<form><center><u>]]><xsl:value-of
select='@question'/><![CDATA[</u></center>";
testtext += "<input type='radio' name='c' value=''
onClick='replaceSpan]]><xsl:value-of
select='@span'/><![CDATA[(&#34;]]><xsl:value-of
select='@answer1'/><![CDATA[&#34;,&#34;]]><xsl:value-of
select='@span'/><![CDATA[&#34;)' />]]><xsl:value-of
select='@choice1'/><![CDATA[<br/>";
testtext += "<input type='radio' name='c' value=''
onClick='replaceSpan]]><xsl:value-of
select='@span'/><![CDATA[(&#34;]]><xsl:value-of
select='@answer2'/><![CDATA[&#34;,&#34;]]><xsl:value-of
select='@span'/><![CDATA[&#34;)' />]]><xsl:value-of
select='@choice2'/><![CDATA[<br/>";
testtext += "<input type='radio' name='c' value=''
onClick='replaceSpan]]><xsl:value-of
select='@span'/><![CDATA[(&#34;]]><xsl:value-of
select='@answer3'/><![CDATA[&#34;,&#34;]]><xsl:value-of
select='@span'/><![CDATA[&#34;)' />]]><xsl:value-of
select='@choice3'/><![CDATA[<br/>";
testtext += "<input type='radio' name='c' value=''
```

```
onClick='replaceSpan]]><xsl:value-of
select='@span'/><![CDATA[(&#34;]]><xsl:value-of
select='@answer4'/><![CDATA[&#34;,&#34;]]><xsl:value-of
select='@span'/><![CDATA[&#34;)' />]]><xsl:value-of
select='@choice4'/><![CDATA[<br/>";

testtext += "<center><table><tr><td><font color='#ff0000'>";

testtext += "<div id='area]]><xsl:value-of
select='@span'/><![CDATA['><span></span></div></font></td></tr></table></center>

document.write(testtext);

// -->

]]>

</script>

</xsl:template>
```

The complexities involved with parsing CDATA, Javascript, and Entities are beyond the scope of this discussion.

Avoid the "white-space" problems involved with copying the above code and Cut and Paste from the zipped Text files. The above xsl:template code for the build-individual-panels.xsl file is included in the *File Modifications* folder

---

Adding Self-Test Elements

to your tutorial's xml file, ADD the following to the desired panel:

```
<self-test span="0" question="?" choice1=" " answer1=" "
choice2=" " answer2=" " choice3=" " answer3=" " choice4=" "
answer4=" " />

<noscript>Javascript must me enabled to see the self-test
example</noscript>
```

The Self-Test feature's answers can be seen by viewing the page's source.

This feature is written by Javascript. If the page is viewed with a browser that does not have Javascript enabled, it will not be visible, the noscript text will be. The tutorial's PDF files will aslo show the noscript text.

Spans are numbered beginning with Zero. If the Self-Test is added to a page that has span tags prior to it's location, the attribute's number should be changed accordingly. Subsequent self-test elements on the same panel would likewise be incremented.

Use Entities with care. Any tags entered into the attribute values need the "left angle" to be entered as < and quotes need to be either ' for single or " for double.

For example, the next panel has:

```
<self-test span="0" question="How many Files need to be
changed?&lt;br />A: the tootomatic.xsl file.&lt;br />B: the
dw-style.xsl file.&lt;br />C: the build-individual-panels.xsl
file&lt;br />D: the dwtut.dtd and tootomatic.xsd files."
choice1="only A" answer1="If you don&amp;#39;t mind creating a
zip file or having a non-functional image-link" choice2="A and
B" answer2="That fixes the zip file problems but leaves a
non-functional feedback form" choice3="C and D"
answer3="Pragmatically, the dtd and xsd files don&amp;#39;t need
to be changed. And you still have the Zip file problem."
choice4="All of them" answer4=" Correct, if you&amp;#39;re a
perfectionist." />

<noscript>What files need to be changed to eliminate thrown
exceptions?</noscript>
```

Avoid the "white-space" problems involved with copying the above code and Cut and Paste from the zipped Text files. The code for the self-test element for the tutorial's xml file is included in the *File Modifications* folder

---

Self-Test Example

Review Questions

When using the Toot-O-Matic application with Java 2 SE v1.4.2_05 ...

What files need to be changed to eliminate thrown exceptions?

What files need to be changed to restore the feedback form's functionality?

What files need to be changed to restore functionality to the e-mail it feature?

How is the Self-Test feature added and used?

Add the self-test xsl:template to the build-individual-panels.xml file.

Add self-test elements to the desired panels

The panel's spans are numbered beginning with 0

Use Entities in the self-test attributes carefully

Section 5. Feedback

Feedback

Please send your feedback on this tutorial by using either, the tutorial form below, or the contact form at *http://www.mittineague.com/cont.php* Happy Coding!

"*I have not failed. I've just found 10,000 ways that won't work.*" ~ Thomas Edison

---

Colophon

This tutorial was written entirely in XML, using the developerWorks Toot-O-Matic tutorial generator. The open source Toot-O-Matic tool is an XSLT style sheet and several XSLT extension functions that convert an XML file into a number of HTML pages, a zip file, JPEG heading graphics, and two PDF files. Our ability to generate multiple text and binary formats from a single source file illustrates the power and flexibility of XML. (It also saves our production team a great deal of time and effort.)

You can get the source code for the Toot-O-Matic at www6.software.ibm.com/dl/devworks/dw-tootomatic-p. The tutorial Building tutorials with the Toot-O-Matic demonstrates how to use the Toot-O-Matic to create your own tutorials. developerWorks also hosts a forum devoted to the Toot-O-Matic; it's available at www-105.ibm.com/developerworks/xml_df.nsf/AllViewTemplate?OpenForm&RestrictToCategory=11. We'd love to know what you think about the tool.