



18th Annual CODI Conference
Pittsburgh Pennsylvania

NOVEMBER 2007

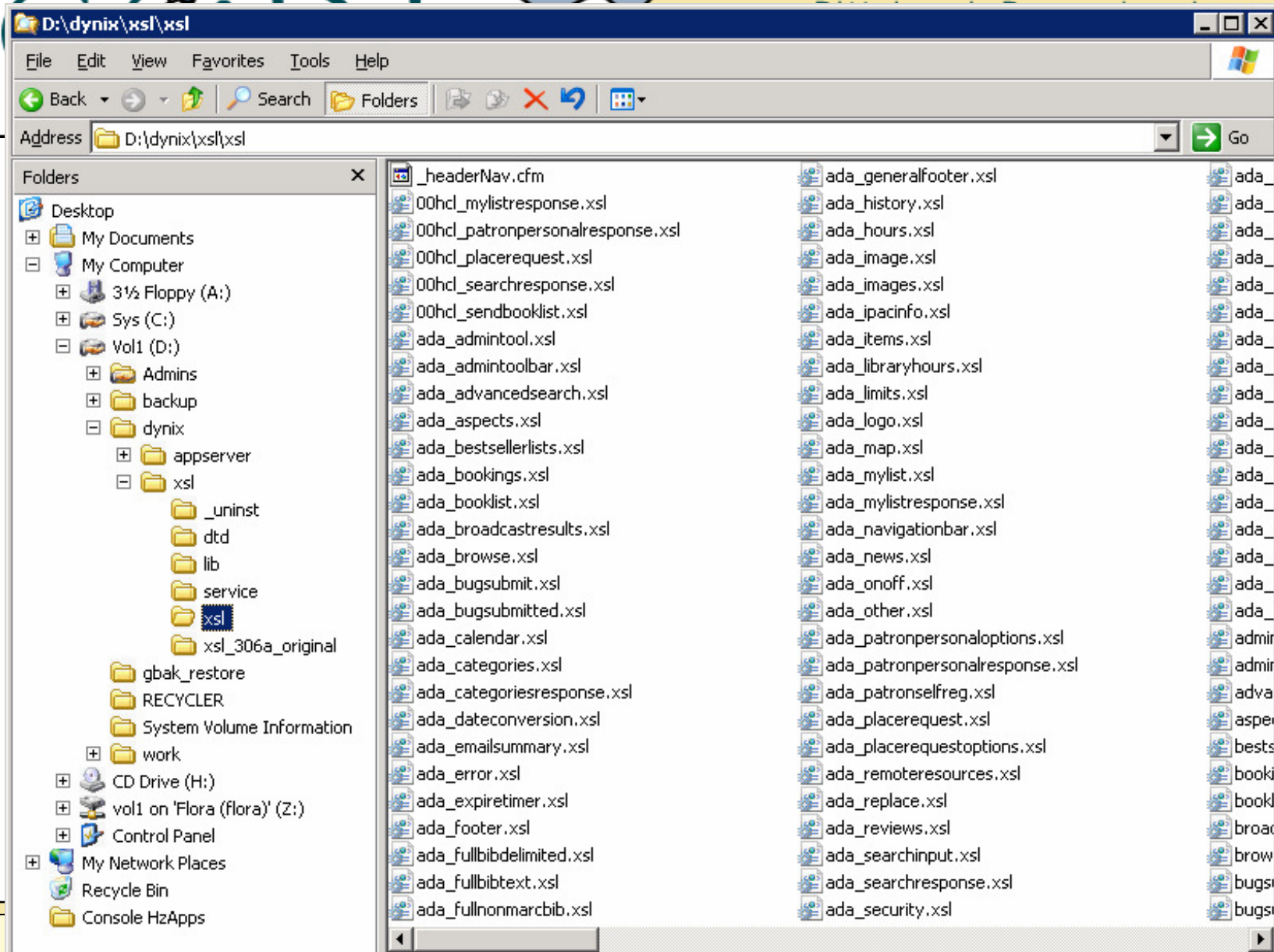
Customizing HIP 2.x/3.x

A Beginner's Guide to Editing XSLT Stylesheets

Phil Feilmeyer
Hennepin County (MN) Library

Scope

- Introduction to customizing the HIP user interface by way of the XSLT Stylesheets
- **XSLT** = e**X**tensible **S**tylesheet **L**anguage for **T**ransformation
 - Tag-based programming language
 - Transforms XML documents into something else
(in this case, HTML)
- Not an XSLT training session, but enough information to get you started



Sample XSLT code

```
<table>
  <xsl:attribute name="border">0</xsl:attribute>
  <xsl:attribute name="width">100%</xsl:attribute>
  <xsl:attribute name="cellspacing">0</xsl:attribute>
  <xsl:attribute name="cellpadding">0</xsl:attribute>
  <xsl:if test="not($totalcount = 0)">
    <xsl:if test="$renewalsallowed = $true">
      <tr>
        <td>
          <xsl:attribute name="width">1%</xsl:attribute>
          <input>
            ...
```

Benefits of Customizing

- Shape the catalog to meet the specific needs of your library
 - Developers need to create a single product that libraries of any type can use
- Override SirsiDynix development decisions
 - especially with regard to layout
- Generates interest in the catalog from staff and the public
 - Sense of ownership when a suggestion is incorporated
- Satisfying

Risks of Customizing

- Not supported, help from SirsiDynix might be billable
 - If you're careful, the risk is small
- Upgrades/patches are more work and take longer
 - Potentially a lot more work and a lot longer
 - Not too bad if a good upgrade plan is developed
- A major change in system architecture could have serious consequences
- Moving to a new OPAC might be a step backwards

Customizing at Hennepin County Library

- HIP 3.08 running on Windows 2003
- Horizon 7.34
- HCL is a single library with 26 branches
 - Each branch has a HIP profile, but they are essentially identical
 - Decision-making aspects of customization are centralized
 - Only one [strings.xsl](#), [other.xsl](#), [onoff.xsl](#)
- Experience with HTML and ColdFusion
 - ColdFusion is a web application platform that uses templates and a tag-based programming language

Customizing at Hennepin County Library

- Resources available for customization effort
 - Development server
 - Staff with experience in web application development
- Started with cosmetic changes
 - Reduction of white space
 - Removal of extraneous information
 - Color conformity with the library's web pages

Customizing at Hennepin County Library

- Breakthrough moment: realizing there are two parts to every HIP page:
 - Functional elements
 - links, form tags, input boxes, submit buttons, JavaScript
 - Display elements
- As long as the **functional elements** are left intact, you can do as you like with the **display elements**
- Good place to start (and experiment) is **generalfooter.xsl**
- First major customization: **toolbar2.xsl**

Header Example (toolbar2.xsl)

HORIZON
Information Portal

 Login  My List - 0  Help

Search

My Account

Basic | Keyword | Format | Advanced | History

Basic Search



START OVER HOURS LIBRARY CARDS INTERLIBRARY LOAN SUGGEST A TITLE FEEDBACK

BASIC SEARCH

SEARCH WITH LIMITS

ADVANCED SEARCH

BOOKSPACE

MY ACCOUNT

 Exit  Login

 Help  My List - 0



Basic Search

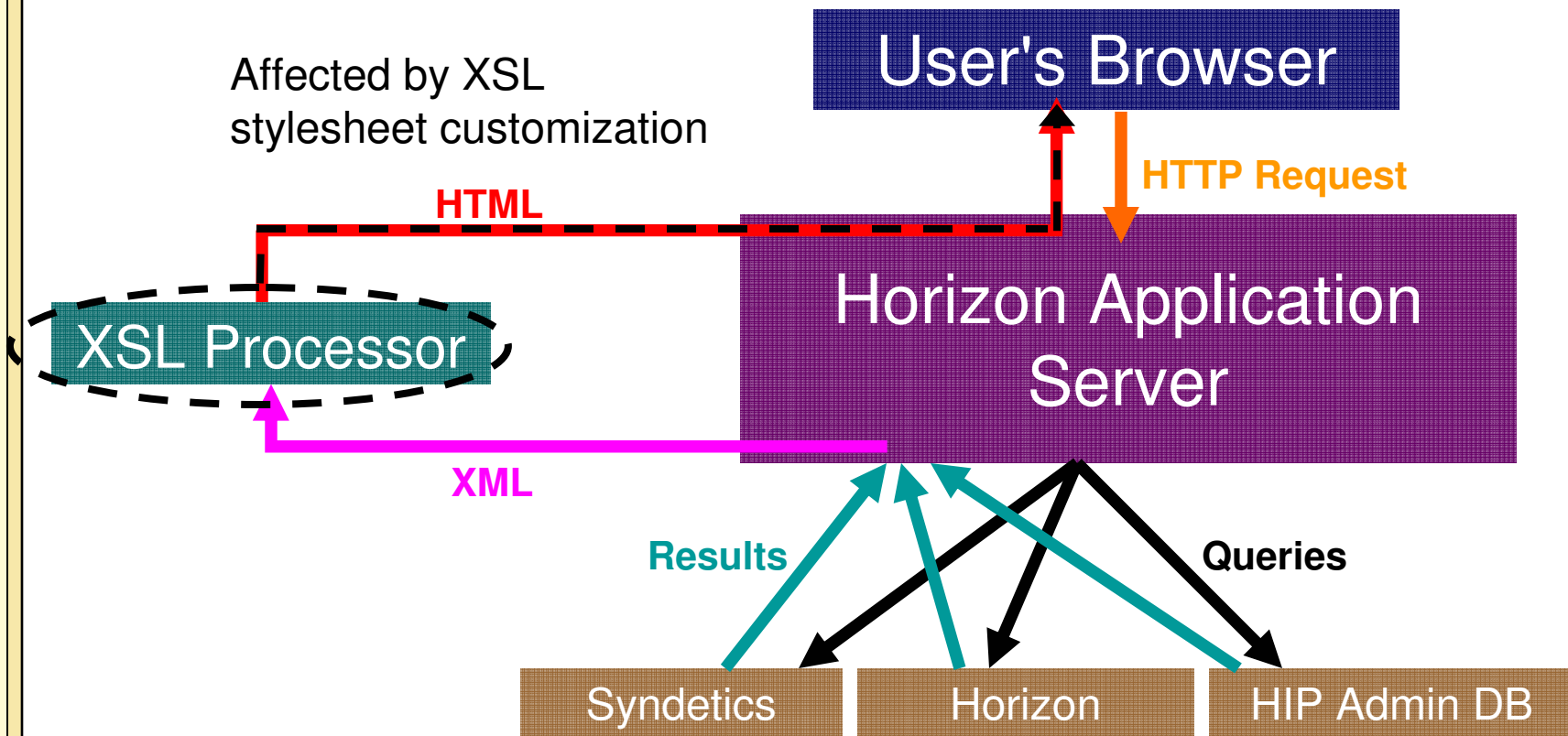
Prerequisites to Customizing

- A solid working knowledge of HTML
 - Primarily working backwards from the final product
- HTML tables
 - HIP uses tables extensively as the framework for each page
 - Tables are used to display variable sets of information
 - Tabs in the tabset, Items in the item display, Subjects in the sidebar
- Basic understanding of how Cascading Stylesheets (CSS) work

Prerequisites to Customizing

- Access and rights to the XSLT stylesheets
- Development HIP
 - Separate installation of HIP for development purposes
 - Develop, test and tweak customizations without bothering users of the production system
 - Very easy to copy pre-tested stylesheets to the production HIP
 - On a separate server
 - Desktop PC with 1GB of RAM (or less?)
 - On the production server?
 - Runs on different ports or a different IP address than production HIP, more complex setup
 - Resource issues?

HIP Architecture: Anatomy of a HIP Request



The XSL Processor

- Stylesheets are organized into six functional areas
- Each of the functional areas has a **root** stylesheet
 - [searchresponse.xsl](#) – search, view results, login
 - [patronpersonalresponse.xsl](#) – my account
 - [placerequest.xsl](#) – hold placement
 - [mylistresponse.xsl](#) – my list, manage lists
 - [sendbooklist.xsl](#) – send a booklist via email
 - [sessioninfo.xsl](#) – ???

The XSL Processor

- Root stylesheets are associated with their supporting stylesheets in the file [xsltransformer.xml](#)
 - Located in the top level XSL folder

Snippet from xsltransformer.xml

```
...  
<Root>  
  <Name>./xsl/placerequest.xsl</Name>  
  <Includes>  
    <Include type="error">error.xsl</Include>  
    <Include type="toolbar">toolbar2.xsl</Include>  
    <Include type="requestoptions">placerequestoptions.xsl</Include>  
    <Include type="footer">generalfooter.xsl</Include>  
    <Include type="timer">expiretimer.xsl</Include>  
    <Include type="bookings">bookings.xsl</Include>  
    <Include type="image">image.xsl</Include>  
    <Include type="features">onoff.xsl</Include>  
    <Include type="attributes">other.xsl</Include>  
    <Include type="string">string.xsl</Include>  
  </Includes>  
</Root>  
...
```

XSLT Editing Tips and Tricks

- Be sure to make a backup copy of a file before you edit
 - Be comfortable with your backup plan
 - Ours includes a nightly automatic XCOPY from `D:\dynix\xsl\` to `D:\backup\dynix\xsl\`
- Use a plain text editor or an editor designed for XSLT
 - MS Word or Word Pad might save the file in a non-ASCII format
- If possible, break up the customization into small edits that can be tested individually along the way

XSLT Editing Tips and Tricks

- For XSLT changes to take affect:
 - Just wait a few seconds for the XSL Processor's "File Watch" to automatically detect and incorporate the change
 - If changes don't seem to take affect, restart XSL Processor
 - Horizon Application Server (JBoss) does not need to be restarted
- Run the XSL Processor as an application rather than as a service when you are testing an edited stylesheet
 - More complete error messages
 - Fast and easy to stop and start

XSLT Editing Tips and Tricks

- Newly edited stylesheets might not be tested until they're called by HIP
 - Example: changes to **security.xsl** won't be tested until the first time a login is attempted
- HIP stylesheets contain both XSL and HTML tags
 - XSL tags are labeled as such:
 - `<xsl:attribute>` `</xsl:attribute>`
 - `<xsl:if>` `</xsl:if>`
 - HTML tags in an XSLT stylesheet look like HTML tags
 - `<table>` `</table>`
 - attributes are applied with the `<xsl:attribute>` tag

XSLT Editing Tips and Tricks

This XSLT:

```
<img>  
  <xsl:attribute name="src">/images/go.gif</xsl:attribute>  
  <xsl:attribute name="border">0</xsl:attribute>  
</img>
```

Generates this HTML:

```

```

XSLT Editing Tips and Tricks

- XSL Processor is picky when it comes to tags:
 - Tags are case sensitive
 - Bad: `<TD> ... </td>`
 - Good: `<td> ... </td>`
 - End tags are required
 - Bad: `
`
 - Good: `
</br>`
 - Empty element tags can be used:
 - Good: `
`
`<xsl:value-of select="$status"/>`

XSLT Editing Tips and Tricks

- Comment Tags

- Add comments to the stylesheet using standard HTML comment tags

```
<!-- pf added series title -->
```

- Use comment tags to deactivate SirsiDynix code

XSLT Editing Tips and Tricks

```
<!-- pf comment out unnecessary stuff
<xsl:if test="boolean(normalize-space(explanation))">
  <a>
    <xsl:attribute name="class">
      <xsl:value-of select="$css_normal_black_font1"/>
    </xsl:attribute>
    <xsl:value-of select="explanation"/>
  </a>
</xsl:if>
-->
```

- Watch out for comment tags within comment tags

XSLT Editing Tips and Tricks

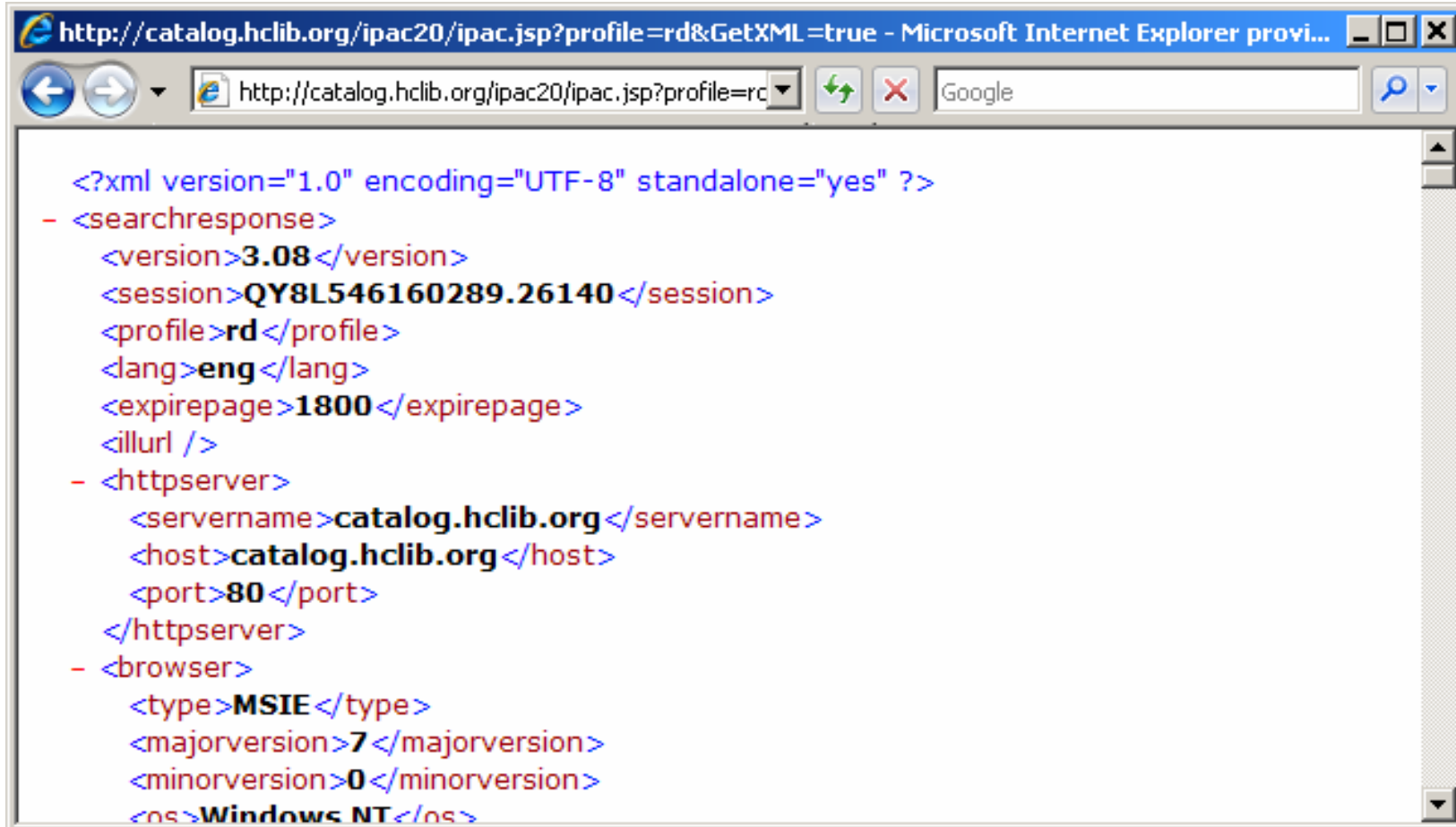
- Following the trail from the HTML source to the corresponding spot in the stylesheets can be tricky
- Never a direct route
 - Root and support stylesheets
 - Variables
 - Templates
 - Cascading Style Sheets
 - Flow control tags

XSLT Editing Tips and Tricks

Root and support stylesheets

- The XSL responsible for the HTML on any given page comes from multiple stylesheets
- Find out which root file by adding "&GetXML=true" to the end of the URL
 - Case sensitive
 - Remove "#focus" from the end of the URL
 - Example:
<http://catalog.hclib.org/ipac20/ipac.jsp?profile=rd#focus>
<http://catalog.hclib.org/ipac20/ipac.jsp?profile=rd&GetXML=true>

XSLT Editing Tips and Tricks



The screenshot shows a Microsoft Internet Explorer browser window. The address bar contains the URL: `http://catalog.hclib.org/ipac20/ipac.jsp?profile=rd&GetXML=true`. The search bar contains the text "Google". The main content area displays XML data with the following structure:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
- <searchresponse>
  <version>3.08</version>
  <session>QY8L546160289.26140</session>
  <profile>rd</profile>
  <lang>eng</lang>
  <expirepage>1800</expirepage>
  <llurl />
- <httpserver>
  <servername>catalog.hclib.org</servername>
  <host>catalog.hclib.org</host>
  <port>80</port>
</httpserver>
- <browser>
  <type>MSIE</type>
  <majorversion>7</majorversion>
  <minorversion>0</minorversion>
  <os>Windows NT</os>
```

XSLT Editing Tips and Tricks

Variables

– HTML

Send suggestions to

```
<a class="normalBlackFont1"  
  href="mailto:jack.blount@sirsidynix.com">  
  jack.blount@sirsidynix.com  
</a>
```

– Variable assignment

```
<xsl:variable name="emailsuggestions">  
  jack.blount@sirsidynix.com  
</xsl:variable>
```

XSLT Editing Tips and Tricks

– Variable called by the XSL code

Send suggestions to

```
<a>  
  <xsl:attribute name="class">  
    <xsl:value-of select="$css_normal_black_font1"/>  
  </xsl:attribute>  
  <xsl:attribute name="href">  
    mailto:<xsl:value-of select="$emailsuggestions"/>  
  </xsl:attribute>  
  <xsl:value-of select="$emailsuggestions"/>  
</a>
```

Templates

- Allows XSLT code to be packaged and reused
- Define a template

```
<xsl:template name="footer">  
  <center>  
    ...  
  </center>  
</xsl:template>
```

- Call up a template

```
...  
<xsl:call-template name="footer"/>  
</form>  
</body>  
</html>
```

XSLT Editing Tips and Tricks

Another template example: `addSessionToURL`

```
<!-- put the current session ID in to the URL -->  
<xsl:template name="addSessionToURL">  
  <xsl:choose>  
    <!-- if sending an email get new session -->  
    <xsl:when test="//sendingemail = '$true'">  
      <xsl:text>session=new</xsl:text>  
    </xsl:when>  
    <!-- if not sending an email preserve session ID -->  
    <xsl:otherwise>  
      <xsl:text>session=</xsl:text>  
      <xsl:value-of select="/searchresponse/session"/>  
    </xsl:otherwise>  
  </xsl:choose>  
</xsl:template>
```

XSLT Editing Tips and Tricks

Output of

```
<xsl:call-template name="addSessionToURL"/>
```

- As part of an email: `session=new`
- All other situations: `session=106W6250V8X87.3594`

Cascading Stylesheets (CSS)

- In HIP, defines style elements for fonts, buttons, etc.
 - Color
 - Size
 - Text decoration: bold, underline, etc.
- Assigned in the XSLT by the 'class' attribute

```
<a>
```

```
  <xsl:attribute name="class">
```

```
    <xsl:value-of select="$css_normal_black_font1"/>
```

```
  </xsl:attribute>
```

```
  <b><xsl:value-of select="$full_bib_call_num"/></b>
```

```
</a>
```

XSLT Editing Tips and Tricks

Flow control tags

– `<xsl:for-each ... >`

- Repeat the code within the tag for each item in the dataset

```
<xsl:for-each select="//blockdata/block">  
  <xsl:call-template name="blocktable"/>  
</xsl:for-each>
```

– `<xsl:if ... >`

- Only use the code within the tag if the condition is true

```
<xsl:if test="boolean(//itemsout)">  
  ...  
</xsl:if>
```

XSLT Editing Tips and Tricks

Flow control tags (continued)

- `<xsl:choose>` `<xsl:when ... >` `<xsl:otherwise>`
 - Choose which code to use from multiple alternatives

```
<xsl:choose>
  <xsl:when test="//profile = 'elibrary'">
    call your local library
  </xsl:when>
  <xsl:when test="//profile = 'staff'">
    call the Help Desk
  </xsl:when>
  <xsl:otherwise>
    ask for assistance at the service desk
  </xsl:otherwise>
</xsl:choose>
```

Editors

- XML/XSLT Editor
 - Google "xslt editor"
- Text Editor
 - Notepad
 - UltraEdit
 - Find/Replace across multiple files
 - Line number indicator, "Go to line" utility
 - Compare files

Managing Upgrades with Customized Stylesheets

- During upgrades, stylesheets may be replaced with new versions
 - Customizations are NOT retained
- Before the upgrade, make sure you have backup copies of all the stylesheets that have been customized along with the original versions

Managing Upgrades with Customized Stylesheets

- After the upgrade, compare the new version of the stylesheet to the old *original uncustomized version*.
 - Date/time modified
 - Size
 - Side by side comparison
- If there are no differences between the old original version and the new version, your customized version needs no changes

Managing Upgrades with Customized Stylesheets

- If there are differences between the old and new stylesheets, find out the nature of the changes
 - If the changes are relatively minor, incorporate them into your old customized stylesheet
 - If the change is relatively major compared to your customizations, incorporate your customizations into the new version of the stylesheet



18th Annual CODI Conference
Pittsburgh Pennsylvania

NOVEMBER 2007

Customizing HIP 2.x/3.x: A Beginner's Guide to Editing XSLT Stylesheets

Phil Feilmeyer

Hennepin County (MN) Library

pfeilmeyer@hclib.org

<http://www.hclib.org>