

SimpleNavigationTree

Example of a navigation tree

- TARGET-AUDIENCE: ***beginner*** advanced expert
 - COCOON-RELEASES: all 2.?
 - DOCUMENT-STATUS: ***draft*** reviewed released
-

What you will get from this page

This is a simple xml xslt Example of a navigation tree

Your basic skills

xml, xslt, cocoon pipeline

Technical prerequisites

a running cocoon webapp

So, lets start

You need a xml file, a xslt file, a sitemap and some images

```

<?xml version="1.0"?>
<NAVIGATION id="0" idcount="11">
  <TOPIC id="1">
    <LINK>
      <NAME>Google (Simple)</NAME>
      <URL target="_self">http://www.google.com</URL>
    </LINK>
  </TOPIC>
  <TOPIC id="2">
    <LINK>
      <NAME>Searchengines (without link)</NAME>
    </LINK>
    <TOPIC id="3">
      <LINK>
        <NAME>Google</NAME>
        <URL target="_self">http://www.google.com</URL>
      </LINK>
    </TOPIC>
    <TOPIC id="4">
      <LINK>
        <NAME>Yahoo</NAME>
        <URL target="_self">http://www.yahoo.com</URL>
      </LINK>
    </TOPIC>
  </TOPIC>
  <TOPIC id="5">
    <LINK>
      <NAME>Searchengines (with link to google)</NAME>
      <URL target="_self">http://www.google.com</URL>
    </LINK>
    <TOPIC id="6">
      <LINK>
        <NAME>Google</NAME>
        <URL target="_self">http://www.google.com</URL>
      </LINK>
    </TOPIC>
    <TOPIC id="7">
      <LINK>
        <NAME>Yahoo</NAME>
        <URL target="_self">http://www.yahoo.com</URL>
      </LINK>
    </TOPIC>
  </TOPIC>
  <TOPIC id="8">
    <LINK>
      <NAME>PC (more deep)</NAME>
    </LINK>
    <TOPIC id="9">
      <LINK>
        <NAME>Games</NAME>
      </LINK>
      <TOPIC id="10">
        <LINK>
          <NAME>Port Royale 2</NAME>
          <URL target="_self">http://www.portroyale2.de</URL>
        </LINK>
      </TOPIC>
      <TOPIC id="11">
        <LINK>
          <NAME>Civilization 3</NAME>
          <URL target="_self">http://www.civ3.com</URL>
        </LINK>
      </TOPIC>
    </TOPIC>
  </TOPIC>
</NAVIGATION>

```

```

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:param name="old"/>
  <xsl:param name="new"/>
  <xsl:variable name="newResult">
    <xsl:choose>
      <!-- IF new in old, then close -->
      <xsl:when test="contains($old,$new)">
        <xsl:value-of select="concat(substring-before($old,$new),substring-after($old,$new))"/>
      </xsl:when>
      <!-- open -->
      <xsl:otherwise>
        <xsl:value-of select="concat($old,$new)"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <xsl:template match="/">
    <html>
      <head>
        <title>Tree</title>
      </head>
      <body>
        <p>
          <xsl:apply-templates select="/NAVIGATION/TOPIC"/>
        </p>
      </body>
    </html>
  </xsl:template>
  <xsl:template match="TOPIC">
    <xsl:if test="./descendant-or-self::TOPIC[not(LINK/URL='') or not(LINK[URL])]">
      <table border="0">
        <tr>
          <td valign="top">
            <xsl:choose>
              <xsl:when test="contains($newResult,concat(':',./@id,':')) and (child::TOPIC)">
                <a name="{@id}"/>
                <a>
                  <xsl:attribute name="href"><xsl:value-of select="concat('page.html?', 'old=', $newResult, '&#x0000;new=', @id, ':# ', @id)"/></xsl:attribute>
                  
                </a>
              </xsl:when>
              <xsl:when test="not(child::TOPIC)">
                
              </xsl:when>
              <xsl:otherwise>
                <a name="{@id}"/>
                <a>
                  <xsl:attribute name="href"><xsl:value-of select="concat('page.html?', 'old=', $newResult, '&#x0000;new=', @id, ':# ', @id)"/></xsl:attribute>
                  
                </a>
              </xsl:otherwise>
            </xsl:choose>
          </td>
          <td>
            <xsl:choose>
              <xsl:when test="./LINK/URL">
                <a>
                  <xsl:attribute name="href"><xsl:value-of select="./LINK/URL"/></xsl:attribute>
                  <xsl:attribute name="target"><xsl:value-of select="./LINK/URL/@target"/></xsl:attribute>
                  <xsl:value-of select="./LINK/NAME"/>
                </a>
              </xsl:when>
              <xsl:otherwise>
                <xsl:value-of select="./LINK/NAME"/>
              </xsl:otherwise>
            </xsl:choose>
          </td>
        </tr>
      </table>
    </xsl:if>
  </xsl:template>

```

```

        <tr>
        <td>
            </td>
        <td>
            <xsl:if test="contains($newResult,concat(':',./@id,':')) and (child::TOPIC)">
                <xsl:apply-templates select="TOPIC"/>
            </xsl:if>
        </td>
        </tr>
    </table>
</xsl:if>
</xsl:template>
</xsl:stylesheet>

```

content.xslt

```

<?xml version="1.0"?>

<map:sitemap xmlns:map="http://apache.org/cocoon/sitemap/1.0">
<!-- ===== Components ===== -->

    <map:components>

        <map:generators default="file"/>
        <map:transformers default="xslt">
            <map:transformer logger="sitemap.transformer.xslt" name="xslt" pool-grow="2" pool-max="32" pool-min="
8" src="org.apache.cocoon.transformation.TraxTransformer">
                <use-request-parameters>true</use-request-parameters>
                <use-browser-capabilities-db>false</use-browser-capabilities-db>
                <use-deli>false</use-deli>
            </map:transformer>
        </map:transformers>
        <map:readers default="resource"/>
        <map:serializers default="html"/>
        <map:matchers default="wildcard"/>
        <map:selectors default="browser"/>
    </map:components>

<!-- ===== Pipelines ===== -->

    <map:pipelines>
        <map:pipeline>

            <map:match pattern="page.html">
                <map:generate src="content.xml"/>
                <map:transform src="content.xsl"/>
                <map:serialize type="html"/>
            </map:match>
            <map:match pattern="images/*.gif">
                <map:read mime-type="image/gif" src="images/{1}.gif" />
            </map:match>
            <map:match pattern="images/*.jpg">
                <map:read mime-type="image/jpg" src="images/{1}.jpg" />
            </map:match>
        </map:pipeline>
    </map:pipelines>
</map:sitemap>

<!-- end of file -->

```

sitemap.xmap

So now call it with page.html

Links to other information sources

page metadata

- AUTHOR:[BR](#)
- AUTHOR-CONTACT: author@site.com[BR](#)
- REVIEWED-BY:[BR](#)
- REVIEWER-CONTACT:[BR](#)