

From www.studio.adobe.com

Transform XML with XSL

Putting all of your data into XML presents a problem—how the heck do you get it into a form so that people can look at it? Adobe® InDesign® CS2 is certainly one answer, but there's another, and that's XSL. XSL, or Extensible Stylesheet Language, exists to transform XML into other formats.

Once upon a time, many moons ago, there was only one Web browser (Mosaic), which ran on a single type of device (a computer). HTML did a reasonably good job of displaying data (Web pages) in that browser on that device. But the Web grew. These days, we have multiple browsers (Internet Explorer, Firefox, Safari) running on multiple platforms (telephones, Palm OS® devices, Windows, the Mac OS, television sets). An HTML format that works well for one of these viewing environments probably won't work for all of the others. So Web site developers faced a problem: how could they avoid writing and maintaining multiple versions of their HTML pages?

The answer lies in the combination of XML and XSL. When you use XSL, you can store the data that makes up your Web pages as XML and transform it into HTML appropriate for viewing on whatever device and browser happens to be connecting to your Web site. If you do this, you need to write and maintain the XSL templates, but the templates change far less frequently than your Web pages.

XSL is made up of two main parts: XSL Transformations (or XSLT), which comprise the transformation language itself and XML Path Language (or XPath), a way to locate data in XML.

At this point, you're probably scratching your head and wondering just exactly what a language for transforming XML into HTML has to do with InDesign. It's this: XSL can transform XML into any text format, including plain text, PDF, PostScript, HTML, other forms of XML, and, our favorite, InDesign tagged text.

Why use XSL to transform XML before placing it in an InDesign document? Well, that depends on your workflow. If you need to import lots of tabular data from your XML files, converting to tagged text first can speed things up, because tables imported from XML appear in InDesign's default table formatting. This usually means that you'll have to select and reformat each table—a task that can be time-consuming, to say the least. If you transform the same XML file to tagged text using XSL, you can specify every attribute of the tables in the file.

Another point about tables—InDesign tagged text supports tables-within-tables; InDesign's XML import does not.

To transform an XML file with an XSL template, you'll have to use an XML parser (or an XML editor)—InDesign cannot apply an XSL template as it imports an XML file.

Here is a (very simple) XML fragment that we'll use in all of the following examples. Note that we're asking you to use your imagination a bit—if the XML files you are working with are really this small and simple, then you do not need the XSL techniques we'll talk about. But if you're looking at XML files several orders of magnitude larger and more complex—as you probably are, if you're working with XML—then you probably do.

```
<author>
  <name>
    <first>Olav</first>
    <middle>Martin</middle>
    <last>Kvern</last>
  </name>
  <address>4016 Francis Avenue North</address>
  <city>Seattle</city>
  <state>Washington</state>
  <zip>98103</zip>
</author>
```

Changing element order with XSL

InDesign's XML import frequently requires that the elements in the XML structure match the order of the appearance elements in the layout (this almost always true of text elements), which means that you might find that you need to re-order the elements in an XML file before you import it. The following is an example XSL template that can change the position of an element in our example XML file.

```
?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>
<xsl:template match="/">
<xsl:element name="author">
<xsl:element name="name">
<!--Rearrange the order of the name elements, placing
the last name first-->
<xsl:copy-of select="author/name/last"/>
<xsl:copy-of select="author/name/first"/>
<xsl:copy-of select="author/name/middle"/>
</xsl:element>
<xsl:copy-of select="author/address"/>
<xsl:copy-of select="author/city"/>
<xsl:copy-of select="author/state"/>
<xsl:copy-of select="author/zip"/>
</xsl:element>
</xsl:template>
</xsl:stylesheet>
```

Transform the example XML using the XSL template above, and you'll get the following output XML.

```
<?xml version="1.0" encoding="UTF-8"?>
<author>
  <name>
    <last>Kvern</last>
    <first>Olav</first>
    <middle>Martin</middle>
  </name>
  <address>4016 Francis Avenue North</address>
  <city>Seattle</city>
  <state>Washington</state>
  <zip>98103</zip>
</author>
```

Duplicating elements with XSL

It's fairly natural to expect that you could use one piece of XML data in multiple places in an InDesign layout—but that's not at all the way that InDesign works. Once you've imported XML, there is a one-to-one correspondence between the elements in the Structure view and their expression in the layout. If you want an element to appear multiple times, you've got to duplicate the element for each appearance on a document page. (Obviously, you can get around this in some cases by placing the XML element on a master page.)

Our layout requires (for whatever reason) that the author's last name appear twice. How can we duplicate the last name field? Try the following XSL template.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >
<xsl:output method="xml" version="1.0" encoding="UTF-8"
indent="yes"/>
<xsl:template match="/">
<xsl:element name="author">
<!--Create a copy of the last name element
with a different element name-->
<xsl:element name="last_name"><xsl:value-of select="author/name/
last"/></xsl:element>
<xsl:copy-of select="author/name"/>
<xsl:copy-of select="author/address"/>
<xsl:copy-of select="author/city"/>
<xsl:copy-of select="author/state"/>
<xsl:copy-of select="author/zip"/>
</xsl:element>
</xsl:template>
</xsl:stylesheet>
```

When you transform the example XML file with the XSL template above, you'll get the following output XML.

```
<?xml version="1.0" encoding="UTF-8"?>
<author>
  <last_name>Kvern</last_name>
  <name>
    <first>Olav</first>
    <middle>Martin</middle>
    <last>Kvern</last>
  </name>
  <address>4016 Francis Avenue North</address>
  <city>Seattle</city>
  <state>Washington</state>
  <zip>98103</zip>
</author>
```

Transforming XML to tagged text

As we mentioned earlier, transforming XML into tagged text for import can offer some significant advantages for some workflows and publications. If you don't care about maintaining the XML structure in your InDesign documents, or if your use of XML involves adding text data from XML elements to other text, you might want to consider this approach.

```
<?xml version="1.0"?>
<xsl:stylesheet version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="text"/>
<xsl:template match="author">&lt;ASCII-WIN&gt;&#10;&#13;
&lt;Version:4&gt;&lt;FeatureSet:InDesign-Roman&gt;&lt;ColorTable:=&l
t;Black:COLOR:CMYK:Process:0,0,0,1&gt;&gt;
&lt;DefineParaStyle:heading&gt;
&lt;DefineParaStyle:body_text&gt;
&lt;DefineCharStyle:name&gt;
&lt;ParaStyle:heading&gt;<xsl:value-of select="name/first"/
>&#32;<xsl:value-of select="name/middle"/>&#32;<xsl:value-of
select="name/last"/>
&lt;ParaStyle:body_text&gt;Once upon a time, there was an
author named &lt;CharStyle:name&gt;<xsl:value-of select="name/
first"/>&#32;<xsl:value-of select="name/middle"/>&#32;<xsl:value-
of select="name/last"/>&lt;CharStyle:&gt; who lived in a strange
little house at <xsl:value-of select="address"/> in <xsl:value-of
select="city"/>, <xsl:value-of select="state"/>.
</xsl:template>
</xsl:stylesheet>
```

When you process the XML example file using the above XSL template, you'll get the following tagged text output.

```
<ASCII-WIN>
<Version:4><FeatureSet:InDesign-Roman><ColorTable:=<Black:COLOR:
CMYK:Process:0,0,0,1>>
<DefineParaStyle:heading>
<DefineParaStyle:body_text>
<DefineCharStyle:name>
<ParaStyle:heading>Olav Martin Kvern
<ParaStyle:body_text>Once upon a time, there was an author named
<CharStyle:name>Olav Martin Kvern<CharStyle:> who lived in a strange
little house at 4016 Francis Avenue North in Seattle, Washington.
```

Excerpted from "Real World Adobe InDesign CS2" by Olav Martin Kvern and David Blatner © 2006. Used with the permission of Pearson Education, Inc. and Peachpit. To buy this book, visit www.adobeypress.com.

Find more tools, tips, and resources for Adobe software in Resource Center at www.studio.adobe.com.



Adobe, the Adobe logo, and InDesign are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. Windows is either a registered trademark or trademark of Microsoft Corporation in the United States and/or other countries. Mac OS is a trademark of Apple Computer, Inc., registered in the United States and other countries. Palm OS is a registered trademark of Palm, Inc.

© 2006 Adobe Systems Incorporated. All rights reserved.