

Úvod do XSL

(zdroj: <http://w3schools.com>)

XSL jazyky

XSL - Extensible Stylesheet Language.

XSL je štýlový jazyk založený na XML. XSL je vyvinutý The World Wide Web Consortium (W3C).

CSS (Cascading Style Sheets) je štýlový jazyk pre HTML

HTML **používa preddefinované značky** (tags) s ich **jasným a dohodnutým významom**. Napr. značka <table> definuje v HTML tabuľku a webový prehliadač vie ako vykonať zobrazenie tejto značky. Pridávaním štýlov pomocou CSS do HTML elementov „hovoríme“ prehliadaču ako zobrazíť daný element, napr. tabuľku, so špeciálnymi fontmi alebo farbami.

XSL (Extensible Stylesheet Language) je štýlový jazyk pre XML

XML **nepoužíva preddefinované značky** (tags), preto **ich význam nie je jasný a vopred dohodnutý**. Značka <table> by mohla v XML znamenať HTML značku, alebo nábytok – stôl, alebo niečo iné. Preto **webový prehliadač nevie ako má vykonať zobrazenie** takejto **XML značky**. XSL popisuje ako by mal byť XML dokument zobrazený.

XSL je viac ako štýlový jazyk

XSL sa skladá z troch častí:

- XSLT – jazyk pre transformovanie XML dokumentov do iných formátov, ako napr. XHTML
 - XPath – jazyk pre navigáciu v XML dokumentoch
 - XSL-FO – jazyk pre formátovanie XML dokumentov
-

Úvod do XSLT

(zdroj: <http://w3schools.com>)

XSLT je jazyk pre transformovanie XML dokumentov do XHTML dokumentov alebo do iných XML dokumentov.

XPath je jazyk pre navigovanie v XML dokumentoch.

Čo je XSLT?

- XSLT - eXtensible Stylesheet Language Transformations
- XSLT je najdôležitejšou časťou XSL
- XSLT transformuje XML dokument do iného XML dokumentu

- XSLT používa XPath pre navigovanie v XML dokumentoch
 - XSLT je W3C Recommendation (odporúčaním) od 16.11.1999
-

XSLT = XSL transformácie

XSLT je najdôležitejšou časťou XSL.

XSLT je použité na transformovanie XML dokument do iného XML dokumentu alebo iného typu dokumentu, ktorý je rozpoznaný webovým prehliadačom, napr. do HTML and XHTML dokumentu. XSLT toto vykonáva transformovaním každého XML elementu do (X)HTML elementu.

Pomocou XSLT môžeme **pridávať** alebo **odstraňovať elementy** a **atribúty do** alebo **z výstupného súboru**. Taktiež môžeme **preusporiadať** a **triediť elementy**, **vykonať ich testy** a rozhodnúť sa, **ktoré elementy skryjeme** a **ktoré zobrazíme** atď.

Obecný spôsob popisu tejto transformácie je nasledovný: **XSLT transformuje zdrojový XML strom do výsledného XML stromu**.

XSLT používa XPath

XSLT používa XPath pre hľadanie informácií v XML dokumente, čiže XPath je použitý pre navigovanie cez elementy a atribúty v XML dokumentoch.

Ako funguje transformácia?

Počas transformačného procesu XSLT používa XPath pre definovanie častí zdrojového dokumentu, ktoré by mali byť spárované s jednou alebo viacerými šablónami. Keď je nájdená zhoda, XSLT transformuje nájdenú časť zdrojového dokumentu do výsledného dokumentu.

XSLT Browsers

Väčšina webových prehliadačov má podporu pre XML a XSLT.

Mozilla Firefox podporuje XML, XSLT a XPath od verzie 3.

Internet Explorer podporuje XML, XSLT a XPath od verzie 6.

Google Chrome podporuje XML, XSLT a XPath od verzie 1.

Opera podporuje XML, XSLT a XPath od verzie 9.

Apple Safari podporuje XML a XSLT od verzie 3.

XSLT - Transformácia

Vysvetlíme ju na prípadovej štúdii: Ako transformovať XML dokument do XHTML dokumentu pomocou XSLT.

Chceme transformovať nasledujúci XML dokument **books.xml** do XHTML dokumentu:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<bookstore>
```

```
<book category="COOKING">
  <title lang="en">Everyday Russian</title>
  <author>Sergey Chelemendik</author>
  <year>2005</year>
  <price>32.00</price>
</book>
```

```
<book category="WEB">
  <title lang="en">HTTP servers</title>
  <author>James Millan</author>
  <year>2003</year>
  <price>39.99</price>
</book>
```

```
<book category="CHILDREN">
  <title lang="en">Nice Day</title>
  <author>J. K. Brown</author>
  <year>2005</year>
  <price>28.99</price>
</book>
```

```
<book category="WEB">
  <title lang="en">Learning XML</title>
  <author>Frank T. Cole</author>
  <year>2003</year>
  <price>37.95</price>
</book>
```

```
<book category="WEB">
  <title lang="en">Learning JavaScript</title>
  <author>Dave Stewart</author>
  <year>2003</year>
  <price>29.85</price>
</book>
```

```
<book category="KLASIKA">
  <title lang="sk">Reci a prednasky</title>
  <author>Ludovit Stur</author>
  <year>2003</year>
  <price>10.50</price>
</book>
```

```
<book category="KLASIKA">
  <title lang="sk">Mor ho!</title>
  <author>Samo Chalupka</author>
  <year>1998</year>
  <price>9.50</price>
</book>
```

```
<book category="KLASIKA">
  <title lang="sk">Marina</title>
  <author>Andrej Sladkovic</author>
  <year>1998</year>
  <price>9.60</price>
</book>
```

```
</bookstore>
```

Vytvorenie XSL štýlového listu (predlohy)

Pre daný XML súbor *books.xml* môžeme vytvoriť nasledovnú XSL štýlovú predlohu ***books1.xsl*** s transformačnou šablónou:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
  <h2>My Books Collection</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
      <th>Title</th>
      <th>Author</th>
    </tr>
    <tr>
      <td><xsl:value-of select="bookstore/book/title"/></td>
      <td><xsl:value-of select="bookstore/book/author"/></td>
    </tr>
  </table>
  </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```

Koreňový element, ktorý deklaruje dokument ako XSL štýlovú predlohu je `<xsl:stylesheet>` alebo `<xsl:transform>`.

Korektný spôsob deklarovania XSL štýlovej predlohy podľa the W3C XSLT Recommendation je:

```
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

alebo:

```
<xsl:transform version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

Aby sme získali prístup k XSLT elementom, atribútom a vlastnostiam, musíme deklarovať priestor názvov XSLT na vrchu *.xml* dokumentu.

Priestor názvov `xmlns:xsl="http://www.w3.org/1999/XSL/Transform"` ukazuje na priestor názvov W3C XSLT.

Do súboru *books.xml* doplníme referenciu na štýlový súbor *books1.xsl*, pričom táto referencia bude druhým riadkom súboru *books.xml*, ktorý bude v nasledujúcom tvare

```
<?xml-stylesheet type="text/xsl" href="books1.xsl"?>
```

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="books1.xsl"?>

<bookstore>

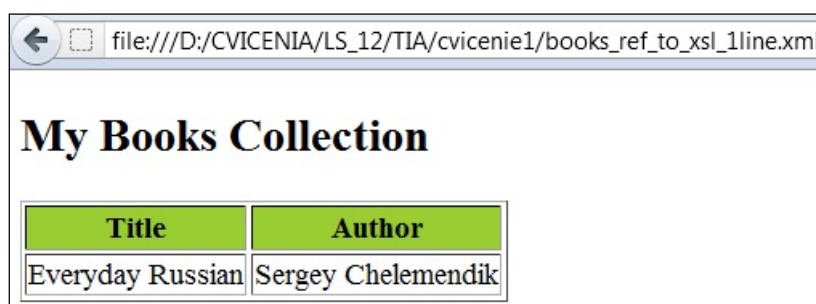
  <book category="COOKING">
    <title lang="en">Everyday Russian</title>
    <author>Sergey Chelemendik</author>
    <year>2005</year>
    <price>32.00</price>
  </book>

  .
  .
  .

</bookstore>

```

Výsledok takejto transformácie XML dokument do XHTML dokumentu pomocou XSLT zobrazený v okne prehliadača:



XSLT <xsl:template> element

XSL štýlová predloha sa skladá z jednej alebo viacerých sád pravidiel, ktoré sa nazývajú šablóny. Šablóna obsahuje pravidlá pre aplikovanie, keď je špecifikovaný uzol nájdený.

<xsl:template> element sa používa na zostavenie šablón.

Porovnávaný (hľadaný) atribút sa používa na asociovanie šablóny s XML elementom. Porovnávaný atribút môže byť tiež použitý na definovanie šablóny pre celý XML dokument. Hodnotou porovnávaného atribútu je XPath výraz. V našom príklade atribút **match="/"** definuje celý XML document. Inak povedané, atribút **match="/"** asociuje šablónu s koreňom XML dokumentu.

XSLT <xsl:value-of> element

<xsl:value-of> element sa používa na extrahovanie hodnoty vybratého uzla a je pridané do výstupného prúdu transformácie.

2 príklady použitia `<xsl:value-of>` elementu z nášho **books1.xsl** súboru uvedeného vyššie:

```
<xsl:value-of select="bookstore/book/title"/>
```

```
<xsl:value-of select="bookstore/book/author"/>
```

select atribúty týchto 2 elementov obsahujú XPath výrazy určené pre nájdenie elementov *title* a *author* v stromovej štruktúre XML dokumentu uloženého v **books.xml** súbore (zobrazený vyššie).

XSLT `<xsl:for-each>` element

`<xsl:for-each>` element nám umožňuje vykonať „cyklovanie“ v XSLT kóde. Pomocou tohto elementu môže byť vybratý každý XML element špecifikovanej uzlovej sady.

Príklad použitia `<xsl:for-each>` elementu:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
    <h2>My Book Collection</h2>
    <table border="1">
      <tr bgcolor="#9acd32">
        <th>Title</th>
        <th> Author </th>
        <th>Year</th>
      </tr>
      <xsl:for-each select="bookstore/book[year=2003]"> <!-- moze byt aj v tvare '2003' -->
      <tr>
        <td><xsl:value-of select="title"/></td>
        <td><xsl:value-of select="author"/></td>
        <td><xsl:value-of select="year"/></td>
      </tr>
      </xsl:for-each>
    </table>
  </body>
</html>
</xsl:template>

</xsl:stylesheet>
```

Výsledok takejto transformácie XML dokument do XHTML dokumentu pomocou XSLT zobrazený v okne prehliadača:

file:///D:/CVICENIA/LS_12/TIA/cvicenie1/books_ref_to_xsl_filtering_years.xml		
My Book Collection		
Title	Author	Year
HTTP servers	James Millan	2003
Learning XML	Frank T. Cole	2003
Learning JavaScript	Dave Stewart	2003
Reci a prednasky	Ludovit Stur	2003

Filtrovane výstupu

Výstup z XML súboru môžeme filtrovať pridaním filtrovacieho kritéria do **select** atribútu v `<xsl:for-each>` elemente.

```
<xsl:for-each select="bookstore/book[year=2003]">
```

Povolené filtrovacie operátory sú:

- **=** (zhodný)
- **!=** (nezhodný)
- **<** menší ako
- **>** väčší ako

XSLT <xsl:sort> element

`<xsl:sort>` element sa používa na triedenie výstupu. Tento element sa pridáva dovnútra `<xsl:for-each>` elementu v XSL súbore.

Príklad použitia `<xsl:sort>` elementu (hodnota **select** atribútu indikuje, ktorý XML element bude triedený a hodnota **order** atribútu indikuje vzostupnosť (ascending) alebo zostupnosť (descending) usporiadania výstupu podľa hodnoty triedeného elementu):

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
  <h2>My Book Collection</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
      <th>Title</th>
      <th>Author</th>
```

```

<th>Price (sorted descending)</th>
</tr>
<xsl:for-each select="bookstore/book[price>30]"> <!-- moze byt aj v tvare '30' -->
<xsl:sort select="price" order="descending"/> <!-- element <xsl:sort/> usporiada vyfiltrovane riadky
podla hodnoty "price" zostupne (descending) -->
<tr>
<td><xsl:value-of select="title"/></td>
<td><xsl:value-of select="author"/></td>
<td><xsl:value-of select="price"/></td>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>

</xsl:stylesheet>

```

Výsledok takejto transformácie XML dokument do XHTML dokumentu pomocou XSLT s filtrovaním a triedením výstupu zobrazený v okne prehliadača:

file:///D:/CVICENIA/LS_12/TIA/cvicenie1/du/du_with_sort_price_descending/books_ref_to_xsl_filtering_prices.xml

My Book Collection

Title	Author	Price (sorted descending)
HTTP servers	James Millan	39.99
Learning XML	Frank T. Cole	37.95
Everyday Russian	Sergey Chelemendik	32.00

XSLT <xsl:if> element

<xsl:if> element sa používa na vloženie podmienkového testu, ktorý vyhodnotí (testuje) výraz oproti vybranému obsahu XML súboru.

Tento element sa pridáva dovnútra <xsl:for-each> elementu v XSL súbore.

Syntax <xsl:if> elementu

```

<xsl:if test="výraz">
...vykoná sa nejaký výstup, ak je výraz pravdivý ...
</xsl:if>

```

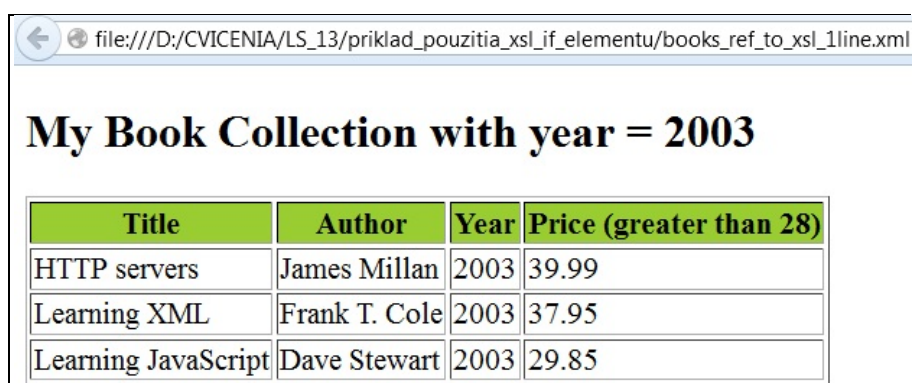

Príklad použitia `<xsl:if>` elementu (atribút **test** tohto elementu obsahuje výraz, ktorý bude vyhodnotený):

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
    <h2>My Book Collection with year = 2003</h2>
    <table border="1">
      <tr bgcolor="#9acd32">
        <th>Title</th>
        <th> Author </th>
        <th>Year</th>
        <th>Price (greater than 28)</th>
      </tr>
      <xsl:for-each select="bookstore/book[year=2003]"> <!-- moze byt aj v tvare '2003' -->
        <xsl:if test="price > 28"> <!-- zobrazime knihy s rokom vydania 2003, ktore maju price > 28 -->
          <tr>
            <td><xsl:value-of select="title"/></td>
            <td><xsl:value-of select="author"/></td>
            <td><xsl:value-of select="year"/></td>
            <td><xsl:value-of select="price"/></td>
          </tr>
        </xsl:if>
      </xsl:for-each>
    </table>
  </body>
</html>
</xsl:template>

</xsl:stylesheet>
```

Výsledok takejto transformácie XML dokument do XHTML dokumentu pomocou XSLT s použitím `<xsl:if>` elementu vo vnútri `<xsl:for-each>` elementu zobrazený v okne prehliadača:



Title	Author	Year	Price (greater than 28)
HTTP servers	James Millan	2003	39.99
Learning XML	Frank T. Cole	2003	37.95
Learning JavaScript	Dave Stewart	2003	29.85

XSLT `<xsl:choose>` element

`<xsl:choose>` element sa používa spolu s `<xsl:when>` a `<xsl:otherwise>` elementmi na vyjadrenie viacnásobného podmieneného vetvenia.

Syntax <xsl:choose> elementu

```
<xsl:choose>
  <xsl:when test="výraz">
    ... nejaký výstup ...
  </xsl:when>
  <xsl:otherwise>
    ... nejaký výstup ....
  </xsl:otherwise>
</xsl:choose>
```

Príklad použitia <xsl:choose> elementu spolu s <xsl:when> a <xsl:otherwise> elementmi:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
    <body style="font-family:Arial;font-size:12pt;background-color:AliceBlue">

      <h2>My Book Collection with prices
      <br/>between 10 and 39 marked by yellow</h2>

      <table> <!-- bez zadania atributu 'border' nebude tabulka ohranicena -->

        <tr style="background-color:teal;color:white">
          <th>Title</th>
          <th>Author</th>
          <th>Price</th>
        </tr>

        <!-- pre kazdy element 'book', ktory je detskym elementom elementu 'bookstore' -->
        <xsl:for-each select="bookstore/book">
          <tr style="font-size:11pt">
            <td><xsl:value-of select="title"/></td>
            <td><xsl:value-of select="author"/></td>

            <xsl:choose> <!-- a ktory ma v detskom elem. 'price' hodn. vacsiu ako 10 a sucasne mensiu ako 39, -->
              <!-- vyber nasledujucu farbu pozadia a styl textu v bunke tabulky, -->
              <xsl:when test="price > 10 and price < 39">
                <td bgcolor="#FFFF00"> <!-- hexadecimalny kod zltej farby -->
                  <span style="font-style:italic"><b><xsl:value-of select="price"/></b></span></td>
                </xsl:when>
                <xsl:otherwise> <!-- inak vyber nasledujucu farbu a styl textu v bunke tabulky -->
                  <td><span style="font-style:italic"><b><xsl:value-of select="price"/></b></span></td>
                </xsl:otherwise>
              </xsl:choose>
            </td>
          </tr>
        </xsl:for-each>

      </table>

    </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```

Výsledok takejto transformácie XML dokument do XHTML dokumentu pomocou XSLT s použitím `<xsl:choose>` elementu spolu s `<xsl:when>` a `<xsl:otherwise>` elementmi zobrazený v okne prehliadača:



Title	Author	Price
Everyday Russian	Sergey Chelemendik	32.00
HTTP servers	James Millan	39.99
Nice Day	J. K. Brown	28.99
Learning XML	Frank T. Cole	37.95
Learning JavaScript	Dave Stewart	29.85
Reci a prednasky	Ludovit Stur	10.50
Mor ho!	Samo Chalupka	9.50
Marina	Andrej Sladkovic	9.60

V texte vyššie je vedený popis vybratých elementov XSLT

Zoznam XSLT elementov (z <http://w3schools.com>)

XSLT elementy z the W3C Recommendation (XSLT Version 1.0).

Vysvetlivka k poslednému stĺpcu tabuľky:

- **FF**: najstašia verzia Mozilla Firefox-u, ktorá podporuje uvedenú značku (tag)
- **IE**: najstašia verzia Internet Explorer-a, ktorá podporuje uvedenú značku (tag)

Pozn.: Elementy podporované IE 5 môžu mať neštandardné správanie, pretože IE 5 bol vydaný skôr, než sa stal XSLT oficiálnym W3C Recommendation.

Zoznam XSLT elementov (z <http://w3schools.com> s popisom v angličtine):

Element	Description	IE	FF
apply-imports	Applies a template rule from an imported style sheet	6.0	1.0
apply-templates	Applies a template rule to the current element or to the current element's child nodes	5.0	1.0
attribute	Adds an attribute	5.0	1.0
attribute-set	Defines a named set of attributes	6.0	1.0
call-template	Calls a named template	6.0	1.0

choose	Used in conjunction with <when> and <otherwise> to express multiple conditional tests	5.0	1.0
comment	Creates a comment node in the result tree	5.0	1.0
copy	Creates a copy of the current node (without child nodes and attributes)	5.0	1.0
copy-of	Creates a copy of the current node (with child nodes and attributes)	6.0	1.0
decimal-format	Defines the characters and symbols to be used when converting numbers into strings, with the format-number() function	6.0	1.0
element	Creates an element node in the output document	5.0	1.0
fallback	Specifies an alternate code to run if the processor does not support an XSLT element	6.0	
for-each	Loops through each node in a specified node set	5.0	1.0
if	Contains a template that will be applied only if a specified condition is true	5.0	1.0
import	Imports the contents of one style sheet into another. Note: An imported style sheet has lower precedence than the importing style sheet	6.0	1.0
include	Includes the contents of one style sheet into another. Note: An included style sheet has the same precedence as the including style sheet	6.0	1.0
key	Declares a named key that can be used in the style sheet with the key() function	6.0	1.0
message	Writes a message to the output (used to report errors)	6.0	1.0
namespace-alias	Replaces a namespace in the style sheet to a different namespace in the output	6.0	
number	Determines the integer position of the current node and formats a number	6.0	1.0
otherwise	Specifies a default action for the <choose> element	5.0	1.0
output	Defines the format of the output document	6.0	1.0
param	Declares a local or global parameter	6.0	1.0
preserve-space	Defines the elements for which white space should be preserved	6.0	1.0
processing-instruction	Writes a processing instruction to the output	5.0	1.0
sort	Sorts the output	6.0	1.0
strip-space	Defines the elements for which white space should be removed	6.0	1.0
stylesheet	Defines the root element of a style sheet	5.0	1.0
template	Rules to apply when a specified node is matched	5.0	1.0
text	Writes literal text to the output	5.0	1.0
transform	Defines the root element of a style sheet	6.0	1.0
value-of	Extracts the value of a selected node	5.0	1.0

variable	Declares a local or global variable	6.0	1.0
when	Specifies an action for the <choose> element	5.0	1.0
with-param	Defines the value of a parameter to be passed into a template	6.0	1.0

Zoznam XSLT funkcií

(z <http://w3schools.com>)

XQuery 1.0, XPath 2.0 a XSLT 2.0 zdieľajú rovnakú knižnicu funkcií.

XSLT zahŕňa viac ako 100 vstavaných funkcií. Tu sú uvedené funkcie pre spracovanie hodnôt reťazcov, číselných hodnôt, pre porovnávanie dátumov a časov, pre manipuláciu s uzlom a QName manipuláciu, sekvenčnú manipuláciu, pre spracovanie Boolean hodnôt atď.

Defaultový prefix pre priestor názvov funkcií je fn:, jeho URI je <http://www.w3.org/2005/xpath-functions>

Funkcie sú často volané s prefixom fn:, ako napr. fn:string(). Odkedy je fn: defaultový prefix priestor názvov funkcií, funkcie nemusia mať prefix, keď sú volané.

Zoznam vybratých vstavaných XSLT funkcií (z <http://w3schools.com> s popisom v angličtine):

Name	Description
current()	Returns the current node
document()	Used to access the nodes in an external XML document
element-available()	Tests whether the element specified is supported by the XSLT processor
format-number()	Converts a number into a string
function-available()	Tests whether the function specified is supported by the XSLT processor
generate-id()	Returns a string value that uniquely identifies a specified node
key()	Returns a node-set using the index specified by an <xsl:key> element
system-property()	Returns the value of the system properties
unparsed-entity-uri()	Returns the URI of an unparsed entity