Introduction to Extensible Stylesheet Language: XSL

Lecture 1: Starting with XPath & XSLT

13 March 2013

Roman Bleier, TCD
bleierr@tcd.ie

Introduction to XSLT – Outline Week 1

- **Literature and online resources**
- **Namespaces**
- **XSL** – ***introduction to the family***

  XSLT, XPath and XSL-FO
- **XPath** ***(introduction, nodes and axes, tree representation, functions, etc.)***

  XPath Exercise
- **XSLT** ***(introduction, XSLT processor, how to work in Oxygen, stylesheet and template element)***

  XSLT Exercise 1 (to be continued in week 2)

Introduction to XSLT – Outline Week 2

- **Quick recap of week 1**

  Continue XSLT Exercise 1

- **More XSLT:**
  - **XSLT *(push vs. pull, value-of, for-each)***

    XSLT Exercise 2
  - **XSLT *(conditional statements, variables, parameters, call-template)***

    XSLT Exercise 3
- **XSLT in Versioning Machine**

Literature

- Doug Tidwell, XSLT (O'Reilly, 2nd ed. 2008).
- Jeni Tennison, Beginning XSLT 2.0: From Novice to Professional (Apress, 2nd ed. 2005).

  Shorter introductions:

  - David Hunt, Beginning XML, chapter 7 and 8 (Wiley, 2007).
  - XML Bible, chapter 17 (Wiley, 2nd ed. 2004) (a bit old!, chapter is online available).

  - A Ten-Minute Guide to XML Namespaces

Online Resources

- IBM's XSLT tutorial:
  http://www.ibm.com/developerworks/xml/tutorials/x-introxslt/

  http://www.ibm.com/developerworks/xml/tutorials/x-xpath/
- XML.com, often older contributions (!):

  Introduction: http://www.xml.com/pub/a/2000/08/holman/

  Five XSLT Basics:
  http://www.xml.com/pub/a/2003/11/26/learnXSLT.html

  What's New in XSLT 2.0:
  http://www.xml.com/pub/a/2002/04/10/xslt2.html
- w3school.com: http://www.w3schools.com/xsl/

Why Namespaces?

To distinguish elements of different standards within the XML document.

<namespace:element>

<tei:u>

In a document same names for elements might be used by various standards.

i.e.: <u>, <p>, are used in TEI and in XHTML

Namespaces are identified via an URI as attribute value on an element.

<TEI xmlns="http://www.tei-c.org/ns/1.0">

Namespaces

<TEI xmlns="http://www.tei-c.org/ns/1.0">

<teiHeader>...

TEI is the default namespace

<tei:TEI xmlns:tei="http://www.tei-c.org/ns/1.0">

<tei:teiHeader>...

TEI not default namespace and has to be referenced through prefix tei

NS is valid for the element it is attached to and its descendants.

@xmlns on the root element refers to the entire XML document.

Example: MENOTA

MENOTA within TEI document

http://menota.org/HB2_index.xml

```
<TEI xmlns="http://www.tei-c.org/ns/1.0" xmlns:me="http://www.menota.org/ns/1.0">
    <teiHeader> [23 lines]
    <text>
        <body>
            <div>
                <ab>
                    <w>
                        <me:facs>lande</me:facs>
                        <me:dipl>lande</me:dipl>
                        <me:norm>landi</me:norm>
                    </w>
                </ab>
            </div>
        </body>
    </text>
</TEI>
```

default namespace

XSL Namespaces

XSLT namespace:

xmlns:xsl="http://www.w3.org/1999/XSL/Transform"

XSL-FO namespace:

xmlns:fo="http://www.w3.org/1999/XSL/Format"

Referencing multiple namespaces:

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
 version="2.0" xmlns:tei="http://www.tei-c.org/ns/1.0">
```

What is XSL?

Extensible Stylesheet Language (XSL)

"A family of recommendations for defining XML document transformation and presentation..."

W3C recommendation: http://www.w3.org/Style/XSL/

Three Parts:

XSLT: transformation language

XSL-FO: formatting language

XPath is used to address nodes and axes (i.e. elements, attributes, groups of elements) in the source document.

XSL Transform and Formatter

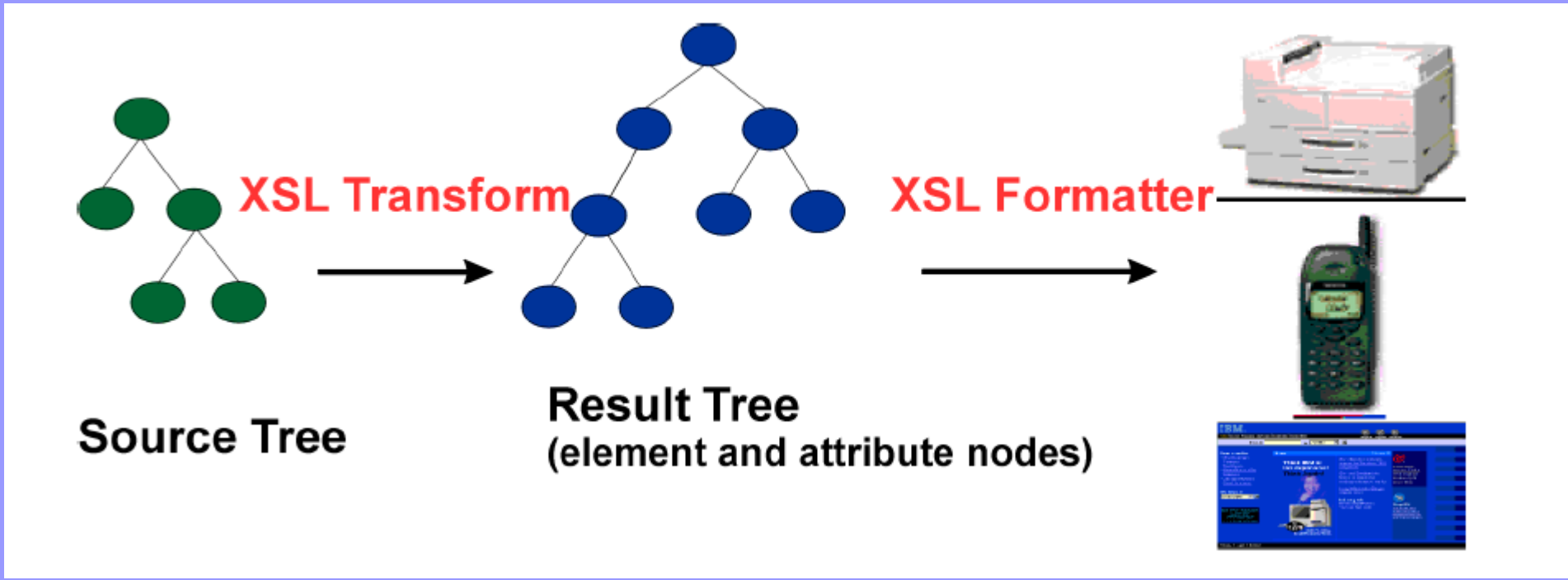


Image from the w3c recommendations: http://www.w3.org/TR/xsl/

W3C Recommendation History

- XSL 1.0 (2001), XSL 1.1 (2006).
- XSLT 1.0 (1999), XSLT 2.0 (2007).
- XSL-FO 1.0 (2001), XSL-FO 1.1 (2006).
- XPath 1.0 (1999), XPath 2.0 (2007, revised recommendation from 2010).

- Working draft XSLT 3.0 and XPath 3.0:

  http://www.w3.org/TR/xslt-30/

  http://www.w3.org/TR/xpath-30/

Introduction to XPath

Data model provides a tree representation of the XML document

Language for hierarchical addressing of nodes and axes in the XML tree

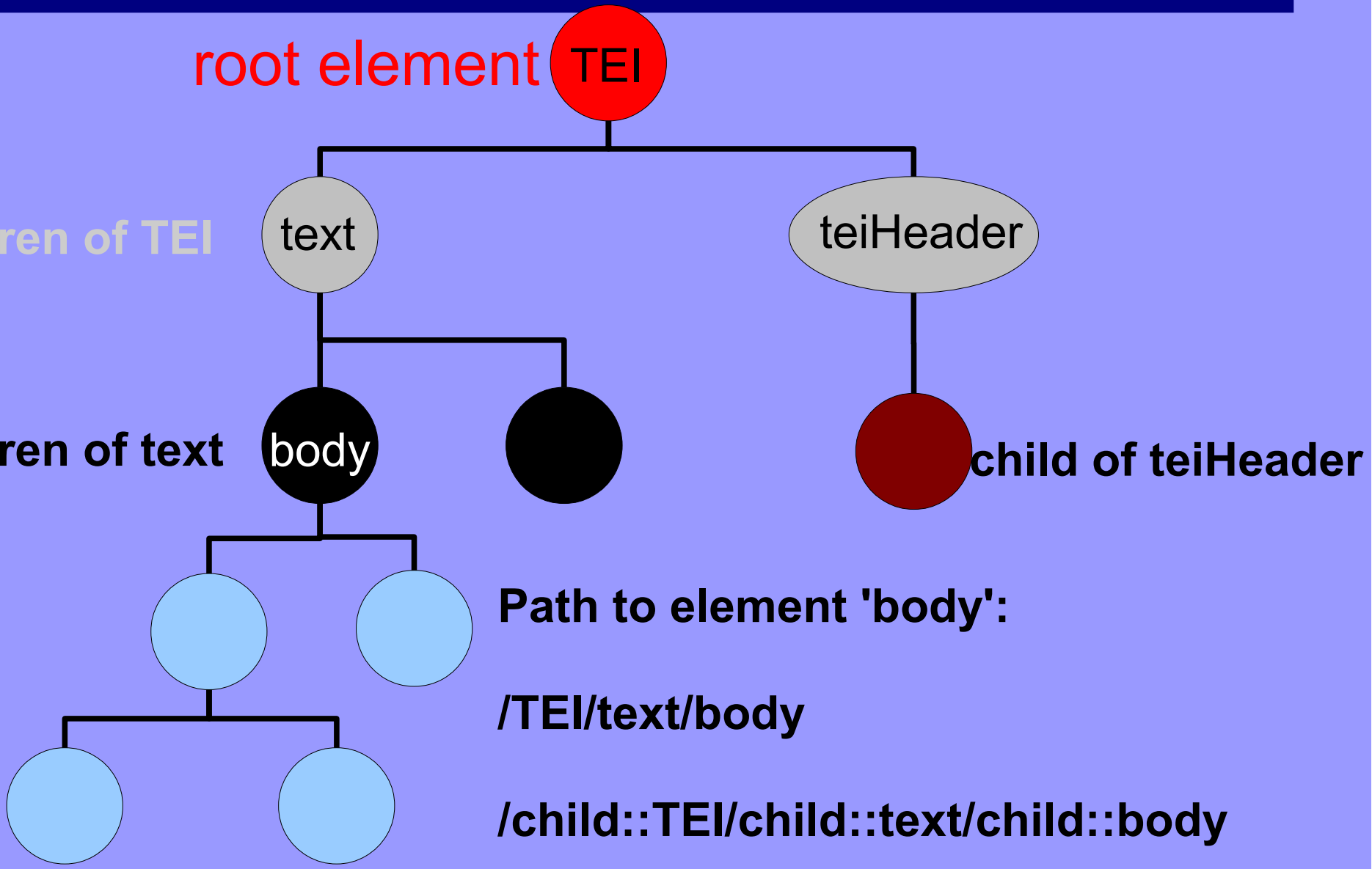
Current W3C recommendation XPath 2.0:

http://www.w3.org/TR/xpath20/

XPath with TEI: talk by James Cumming from 2006, slides on TEI website:

http://www.tei-c.org/Talks/OUCS/2006-02/talk-xpath.pdf

XPath - XML tree



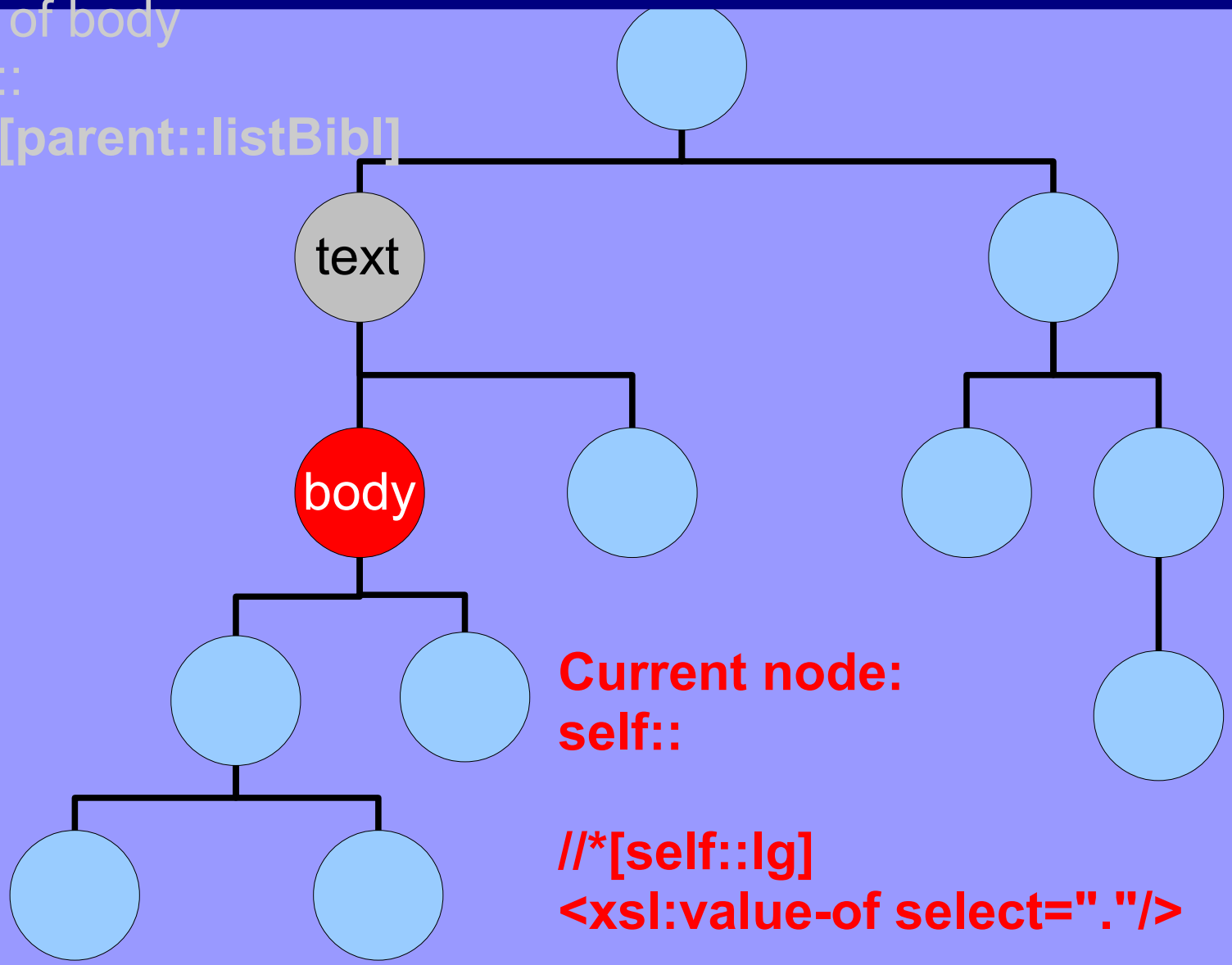
Path to element 'body':

/TEI/text/body

/child::TEI/child::text/child::body

self and parent axes

parent of body
parent::
//head[parent::listBibl]



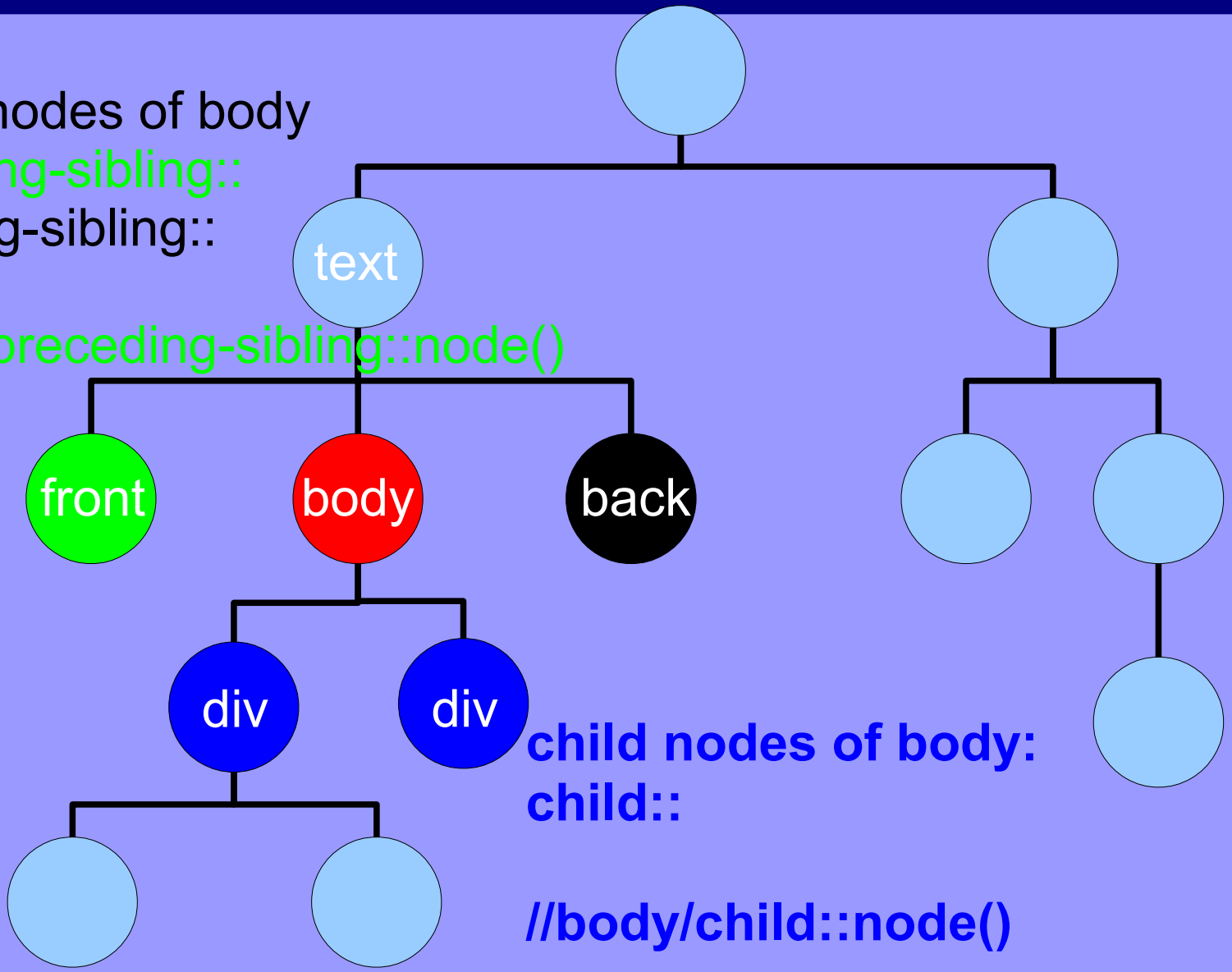
Current node:
self::

//*[self::lg]
<xsl:value-of select="."/>

sibling and child axes

sibling nodes of body
preceding-sibling::
following-sibling::

//lg/l[2]/preceding-sibling::node()

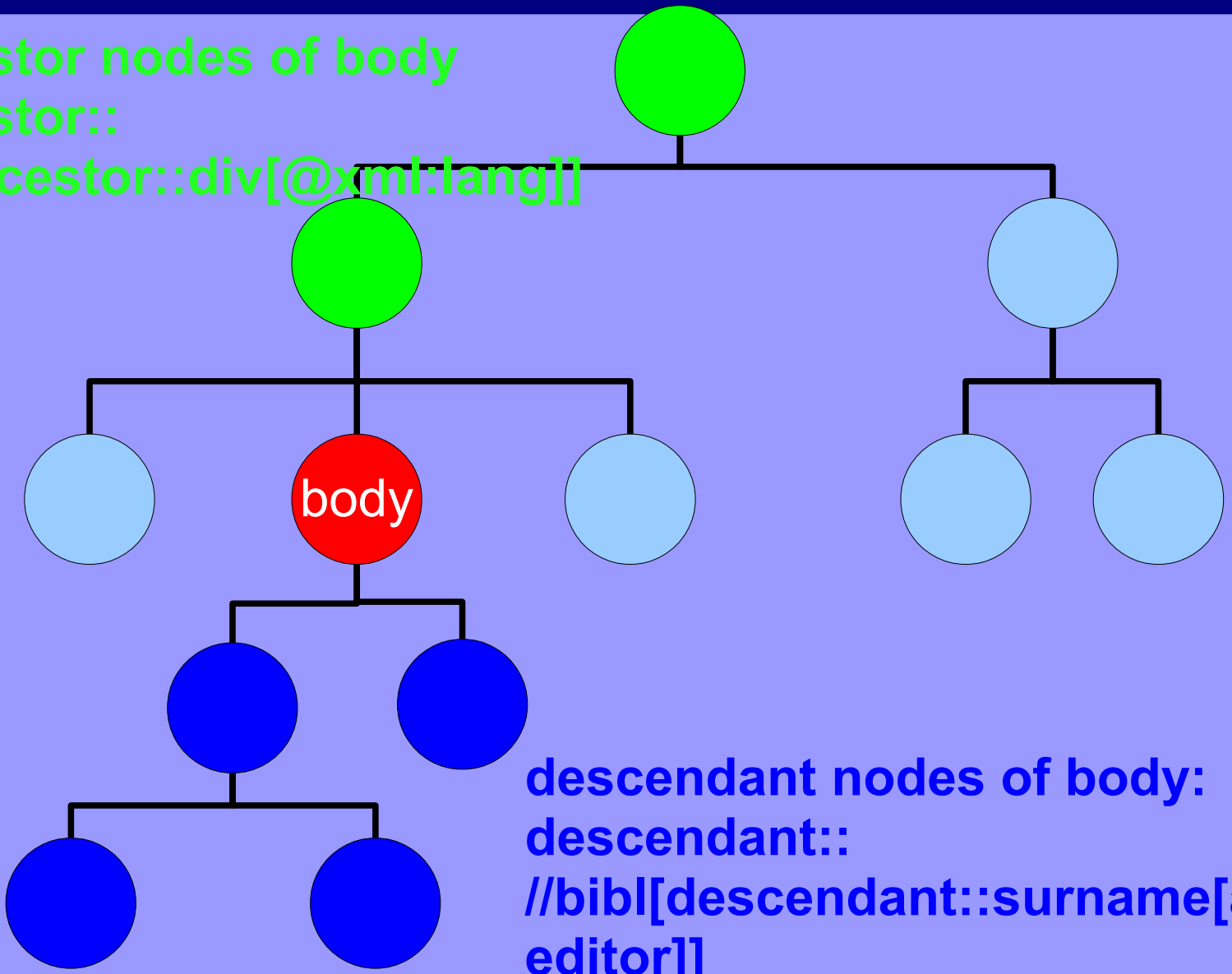


child nodes of body:
child::

//body/child::node()

ancestor and descendant axes

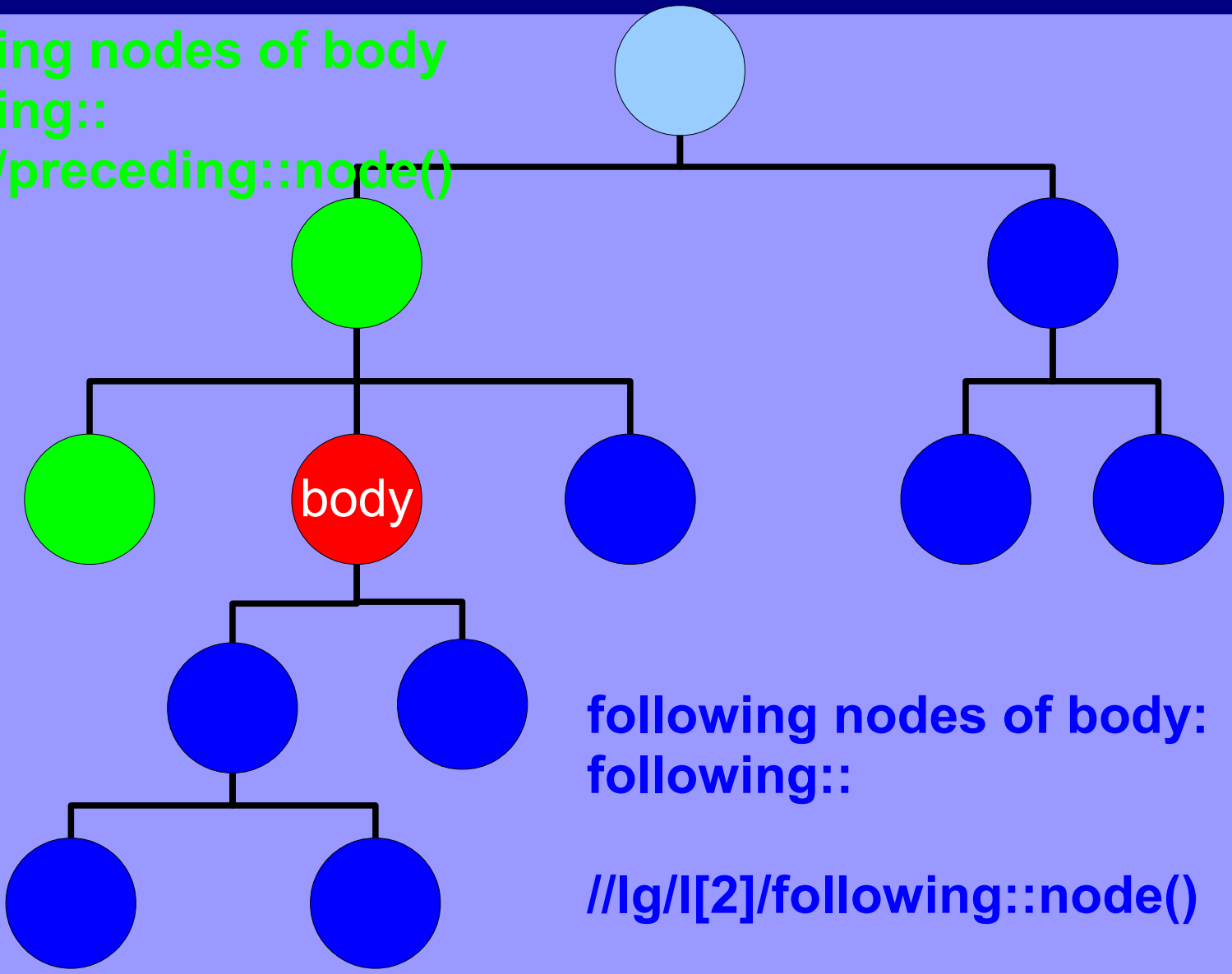
ancestor nodes of body
ancestor::
//*[ancestor::div[@xml:lang]]



descendant nodes of body:
descendant::
//bibl[descendant::surname[ancestor::
editor]]

preceding and following axes

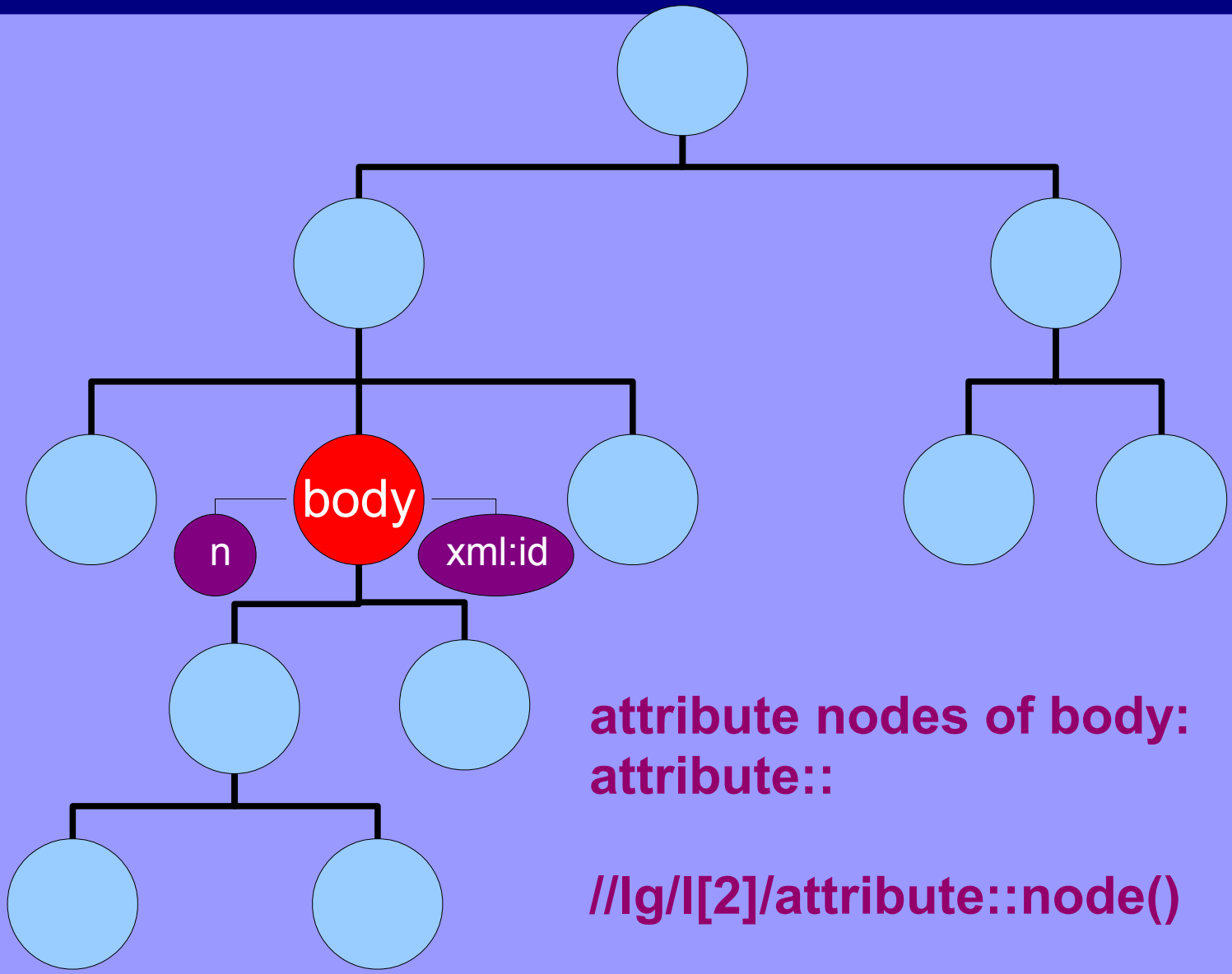
preceding nodes of body
preceding::
//lg/l[2]/preceding::node()



following nodes of body:
following::

//lg/l[2]/following::node()

attributes axis



attribute nodes of body:
attribute::

//lg/l[2]/attribute::node()

XPath – Attributes Axis

Attributes are separate nodes

To select attributes of current node:

attribute:: or @

Select all attributes of a node:

element/@*

Select attribute with a particular name (i.e. type):

element/@type

Search for an attribute with a particular value – returns true or false:

element/@type="stanza"

XPath - Shorthand Notation, Wildcard, Predicates

Expression	Example	Description
/	/TEI/teiHeader	**Start from document node and proceed down in the hierarchy: Selects teiHeader starting from the root element TEI**
//	//p	**Anywhere in the hierarchy: Selects any p element in the document**
.		**Self-selector: Selects current node**
..	//div/..	**Parent-selector: Selects parent node**
@	//div/@type	**Attribute-selector: Selects type attributes on divs**
*	//div/@*	**Wildcard: Selects all attributes on divs**
[ ]	//div/p[@type]	**Predicates: Selects all p elements, but only if they have a type attribute**

XPath - Example Expressions

- **/TEI**
- **/TEI/text**
- **/TEI/text/body/div/child::head**
- **/TEI/text/body/div/head/hi/@xml:lang**
- **//head/hi/parent::node()**
- **/TEI/text/body/div/lg[1]/l[3]**
- **//div/lg[@n=1]/child::node()**

XPath Functions

Function	Example
not()	//div/lg[1]/l[not(@n=1)] Returns: boolean
position()	//div/lg[1]/l[position()>=3] Returns: number
contains()	contains(//lg[1]/l[5], "near") Returns: boolean
last()	//div/lg/l[last()] Returns: list of one or more nodes
substring()	substring(//lg[1]/@type, 3) substring(//lg[1]/@type, 3, 2) Returns: string
substring-before() substring-after()	substring(//lg[1]/@type,4) Returns: string
starts-with() ends-with()	ends-with(//lg[1]/l[1], "all") Returns: boolean

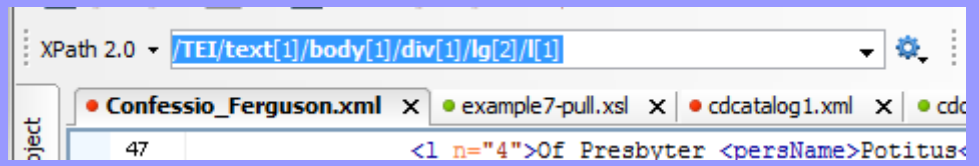
XPath - Operators

Operator	Description	Example	Return
|	Computes two node-sets	//lg | //p	a node-set with all <lg> and <p> elements
+ - *	Addition, Subtraction, Multiplication	//l[@n=3+4]	all <l> with @n=”7”
div	Division	//l[@n=9 div 3]	all <l> with @n value of 3
!= =	Not equal, Equal	//lg[@type="stanza"]	all <lg> with an attribute of type stanza
< >	Less than, Greater than	//l[@n>3]	all <l> with an @n value bigger than 3
<= >=	Less than or equal to, Greater than or equal to	//l[@n>=3]	all <l> with an @n value bigger than or equal to 3
or	or	//l[@n="2" or @n="7"]	All <l> with @n value of 2 **or** of 7
and	and	//lg[@n="2" and @type="stanza"]	all <lg> with @n value of 2 **and** a @type value of “stanza”
mod	Modulus (division remainder)	//l[@n=12 mod 5]	All <l> with an @n value of 2

XPath - Exercise

Getting used to Oxygen:

- Open Confessio_Ferguson.xml in Oxygen
- Familiarize yourself with the Oxygen XPath toolbar and the result window



Try the XPath examples from the previous slides (Example Expressions, XPath – Functions and XPath – Operators)

Observe how Oxygen highlights the nodes that are addressed with your XPath expression.

Exercises – Source Text

SIR SAMUEL FERGUSON—*On the Patrician Documents.*

THE "CONFESSIO" OF SAINT PATRICK.

[The passages ending with (A) are from the Armagh Codex; those ending with (B) are from the Bodleian.]

(*Before A. D. 500.*)

I, Patrick, sinner, most unlearned of all
The Faithful, and of many most despised,
Had, for my father, Deacon Calphurn, son
Of Presbyter Potitus, of a place
Called Bannow of Tabernia, near whereto
He owned his country dwelling; and 'twas there
I suffered capture, then not full sixteen.
I knew not the true God; and, led away
Into captivity, with thousands more,
Was brought to Ireland—fate too well deserved.

Ferguson, Samuel, "On the Patrician Documents", *Transactions of the Royal Irish Academy,* 27, Polite Literature and Antiquities (1877-1886 [Read, Nov. 10, 1884 ; Jan. 12, 26; Feb. 9, 1885]), 71.

Exercises - XML File

```
<head>The <hi>Confessio</hi> of Saint Patrick</head>
<lg  n="1" type="stanza">
<l  n="1">I, Patrick, sinner, most unlearned of all</l>
<l  n="2">The Faithful, and of many most despised,</l>
<l  n="3">Had, for my father, Deacon Calphurn, son</l>
<l  n="4">Of Presbyter Potitus, of a place</l>
<l  n="5">Called Bannow of Tabernia, near whereto</l>
<l  n="6">He owned his country dwelling; and 'twas there</l>
<l  n="7">I suffered capture, then not full sixteen.</l>
</lg>
<lg  n="2">
<l  n="1">I knew not the true God; and, led away</l>
<l  n="2">Into captivity, with thousands more,</l>
<l  n="3">Was brought to Ireland—fate too well deserved.</l>
</lg>
```

Exercises – XML File

```
<listBibl>
      <head>Bibliography</head>

       <bibl xml:id="OLoughlin2000" type="Book">
          <author>
                <name>
                      <forename>Thomas</forename>
                      <surname>O'Loughlin</surname>
                </name>
          </author>
          <date when="2000">2000</date>
          <title level="m">
                Celtic Theology: Humanity, World and God in Early Irish Writings
           </title>
      </bibl>

</listBibl>
```

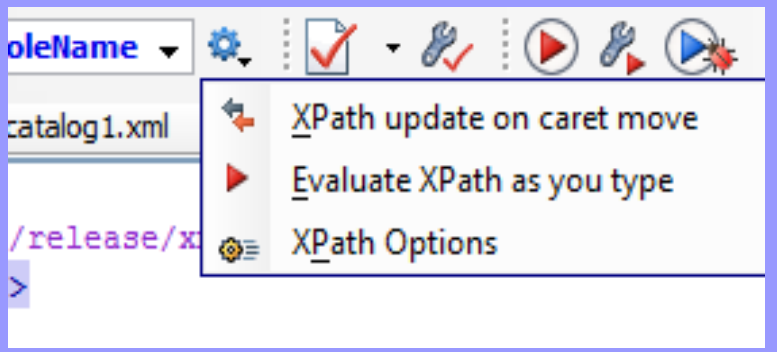
XPath Helper in Oxygen

Context menu:

Right-click on element – Context menu – **Copy XPath**

Right-click on element – Context menu – **Select**

XPath toolbar – Settings



Windows / Show view:

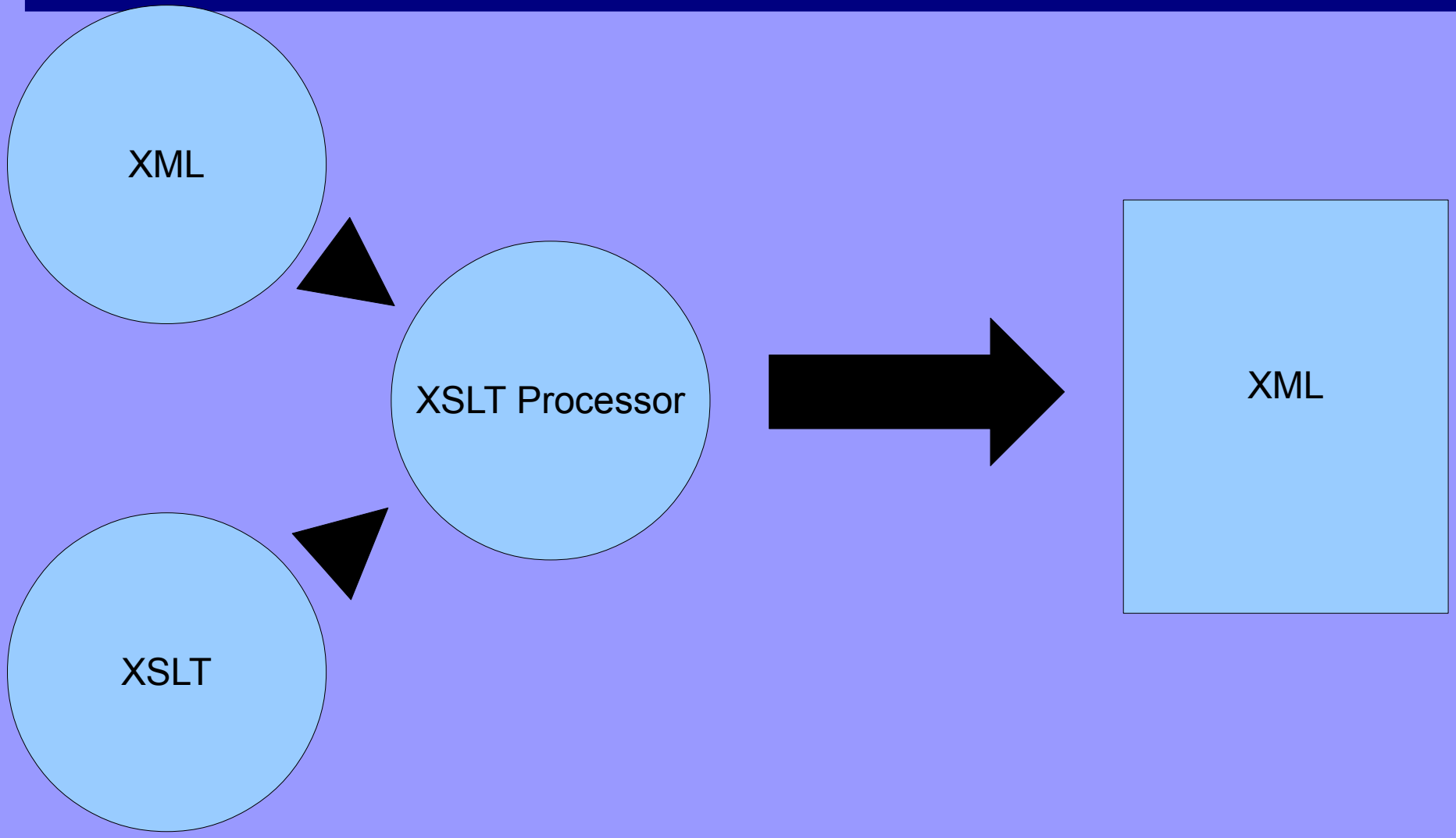
Outline, XPath/XQuery Builder

- Oxygen Website:
  http://www.oxygenxml.com/doc/ug-editor/topics/xpath-console.html
  http://www.oxygenxml.com/xml_editor/xslt.html

Introduction to XSLT

- e**X**tensible **S**tylesheet **L**anguage **T**ransformation
- Set of elements to transform XML content
- Transformation of a XML document into other XML document(s) (i.e. TEI to XHTML)
- XSLT document is XML:
  - well-formed, namespace, <xsl:stylesheet> root element
- The file ending is .xsl
- XPath expressions are used to select elements or axes for processing

How does a Transformation work?



XSLT Tree Transformation

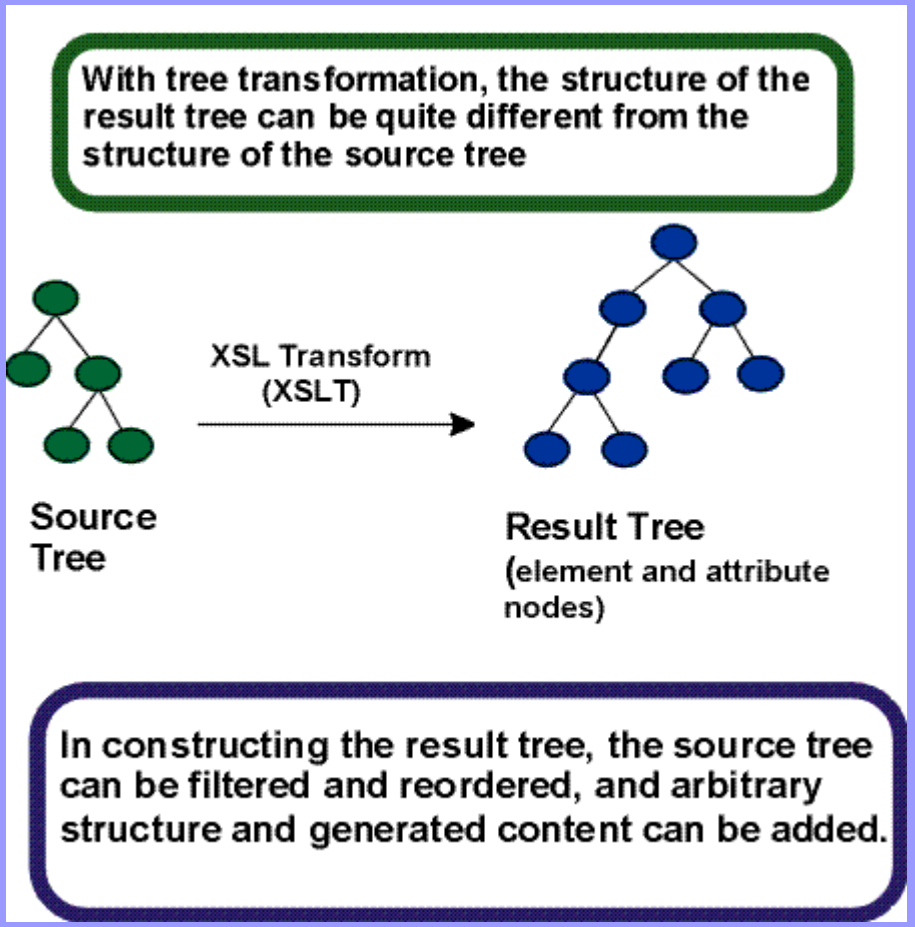
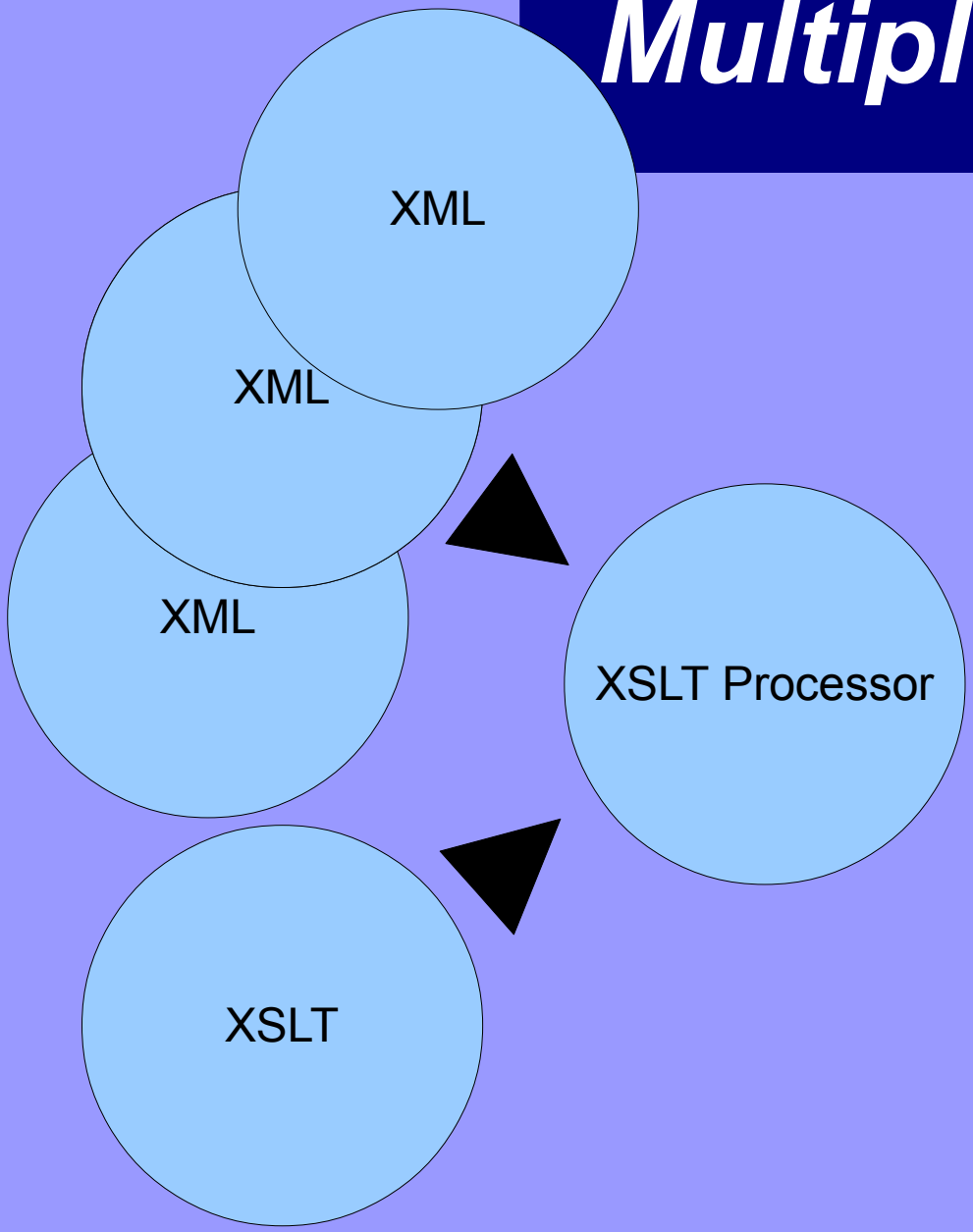


Image from the w3c recommendations: http://www.w3.org/TR/xsl/

Multiple XML Documents



Three different XML document processed with the same XSLT

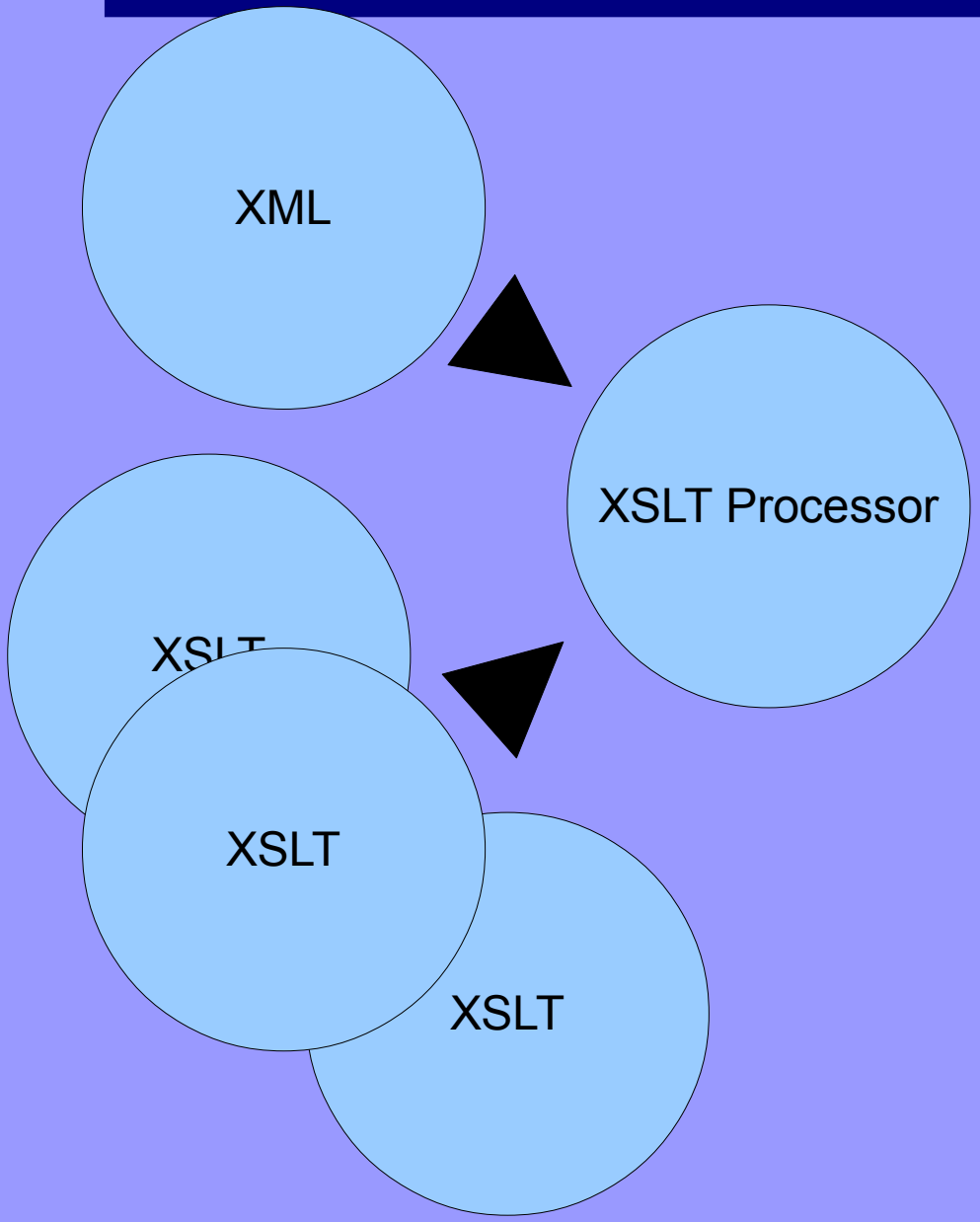
XML

XML

XML

Result tree of the three documents is the same

Multiple XSLT Documents



Three versions of the same XML document

XML

XML

XML

Result tree of the three documents can be different

XSLT Processor

“The basic operation can be summarized as follows: when a node in the source matches a rule's pattern, the content of that rule is created in the result tree. Once you grasp this basic operation, the overall XSLT processing model is easy to understand. Given a source tree and a stylesheet, the XSLT processor carries out the transformation described by rules in the stylesheet by following a sequence of steps...”

http://oreilly.com/catalog/orxmlapp/chapter/ch07.html

Where does the transformation happen?

- **Client side**: XML document and XSLT are both served to the client (Web browser).
- **Server side**: The server applies an XSLT to an XML document to transform it to HTML (or other format). The transformed document is than send to the client.
- A **third software**, for instance Oxygen, performs the transformation the result document (i.e. HTML) is than put on the Server. Server and client deal only with HTML.

Example: XSLT in Browser

Newer browser should support at least XSLT 1.0

http://www.w3schools.com/xsl/xsl_browsers.asp

Reality is sometimes a bit different and some browser might do unexpected things – TESTING!!!

How does it work? Include the reference to the XSLT file in the XML source document:

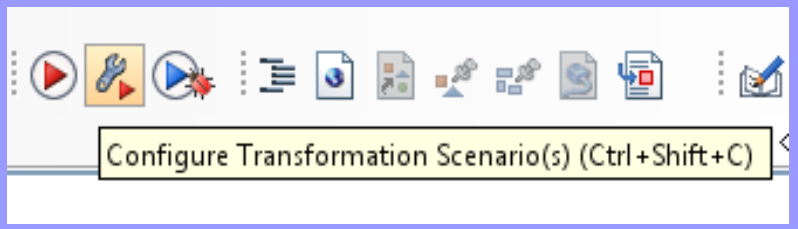
<?xml-stylesheet type="text/xsl" href="fileName.xsl"?>

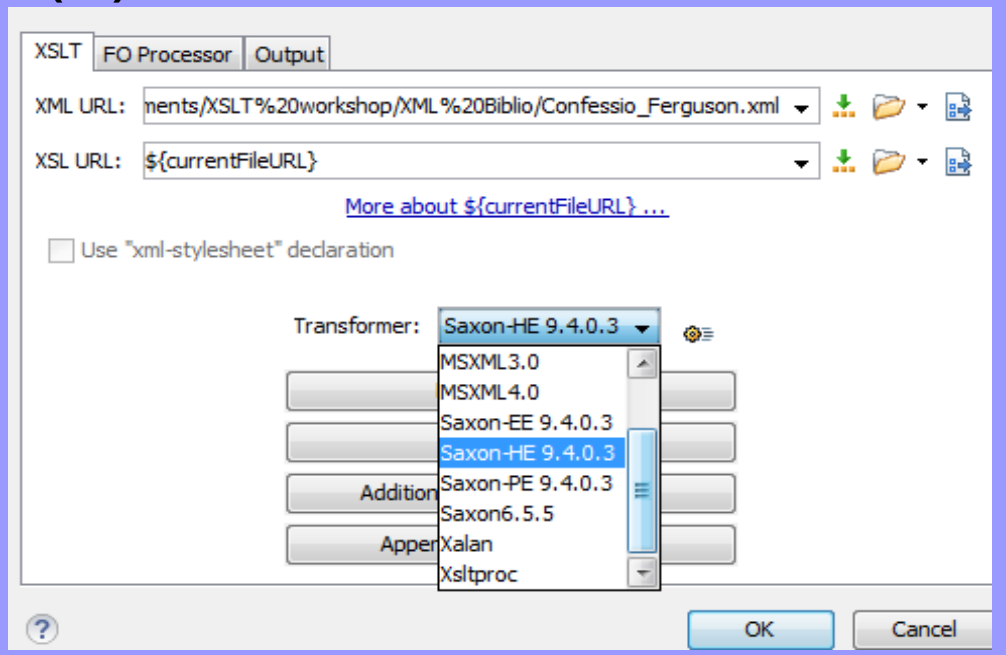
Another way is to use Javascript:

http://www.w3schools.com/xsl/xsl_client.asp

XSLT Processor in Oxygen

- Top menu: Document-Transformation-Configure transformation scenario(s) – New or Edit scenario
- Or over toolbar





- **Oxygen standard processors:**
  - Saxon and Xalan (open source, Java)
  - More info on Oxygen website.

The XSLT Root Element

- Root element*, two required attributes:

  @version

  @xmlns

```
<xsl:stylesheet version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:tei="http://www.tei-c.org/ns/1.0">

    <!-- further XSLT elements -->

    </xsl:stylesheet> <!-- closing tag -->
```

*One of the two root element (xsl:stylesheet and xsl:transform) has to be used. There is not difference between them.

The template element

Associates particular output with particular input

Are child elements of xsl:stylesheet

May contain: text, elements of another standard, XSLT instructions, commentary

@match or @name is required

```
<xsl:template match="/">
    <html>
        <head></head>
        <body>Hello World!</body>
    </html>
</xsl:template>
```

Apply-templates

```
<xsl:apply-templates select="//tei:lg/tei:l">
</xsl:apply-templates>
```

Can be matched (through @select, @match) to a template element

A Template:

- can be applied more than once
- may itself contain “apply-templates” calls

The processed node is treated as a tree!

Build-in Templates

By default the build-in templates are used.

Simple example:

```
<xsl:stylesheet xmlns:xsl="...">
    <xsl:template match="/">
        <xsl:apply-templates></xsl:apply-
    templates>
        </xsl:template>
</xsl:stylesheet>
```

This template will process the XML content according to the following rules...

Build in template

For elements and document nodes:

```
<xsl:template match="*|/">
    <xsl:apply-templates/>
</xsl:template>
```

For comment and processing-instruction nodes:

```
<xsl:template match="processing-instruction()|comment()"/>
```

Text and attribute nodes:

```
<xsl:template match="text()|@*">
    <xsl:value-of select="."/>
</xsl:template>
```

XSLT Exercise 1

- Use Confessio_Ferguson.xml
- Create a new XSLT file, name it and safe it in the same folder as Confessio_Ferguson.xml
  - First invoke the build-in templates as shown before
  - Observer how the XML content is displayed in the Browser
- Transform the XML into a valid HTML web page:
  - Display only the body content
  - The title (tei:body/tei:div/tei:head) should become <h2>
  - The lines should be displayed as paragraphs: tei:l should become HTML <p>
  - Display the line number @n at the beginning of the line
  - Name the created HTML file **text.html**