More XML
XPATH, XSLT

CS 431 – February 22, 2006

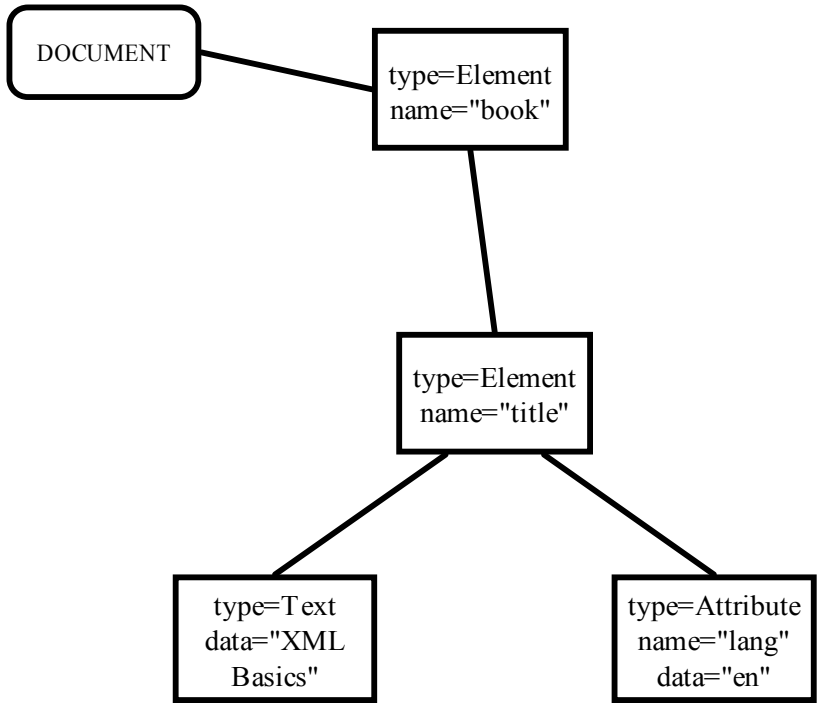
Carl Lagoze – Cornell University

XPath

- Language for addressing parts of an XML document
  - XSLT
  - Xpointer
- Tree model similar to DOM
- W3C Recommendation (1999)
  - http://www.w3.org/TR/xpath

Remember to think in terms of DOM trees

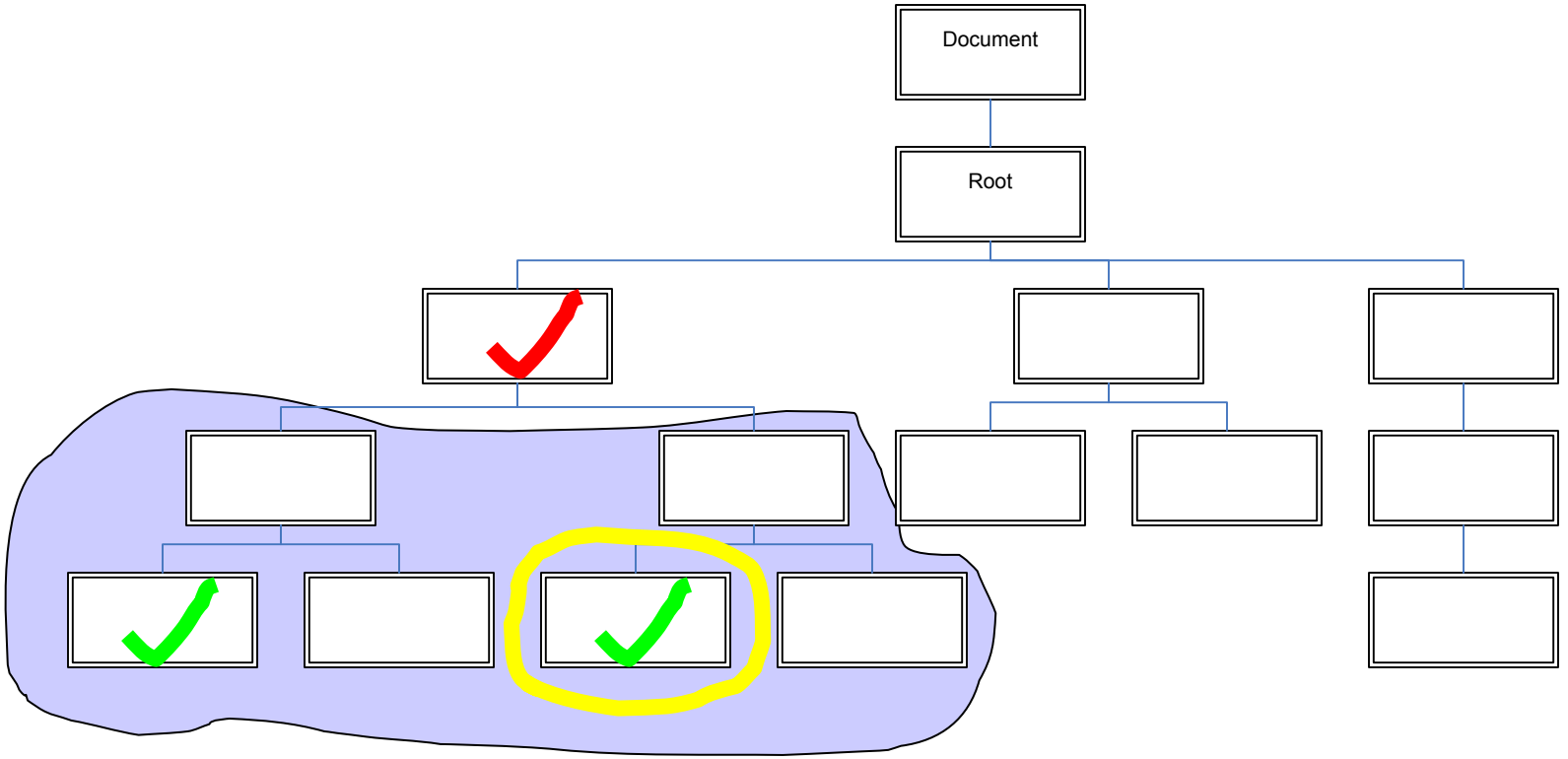
```
<?xml version="1.0"
    encoding="UTF-8"?>
<book>
    <title lang='"en"'>"XML
    Basics"</title>
</book>
```



Xpath Concepts

- Context Node
  - current node in XML document that is basis of path evaluation
  - Default to root (remember that root is "Document")
- Location Steps – selection from context node
  - Axis – sub-tree(s) selection from context node
  - Node Test – select specific elements or node type(s)
  - Predicates – predicate for filtering after axis and node tests

Context, Axis, Node Test, Predicate



Location Path Specification

- /step/step/…. – absolute from document root
- step/step …. – relative from context
- //step/step – anywhere in document tree

- where step is: axis::node-test[predicate]

axis::node-test[predicate]

- child:: all children of context
- descendant:: all children, grandchildren, …
- parent:: parent of context
- ancestor:: all nodes on path to root from context

axis::node-test[predicate]

- Element name: e.g. "Book"
  - make sure to pay attention to namespaces!!!!
- Wildcard: *
- *Type*(): where type is "node", "text", etc.
  - *Remember in DOM that everything is a node*

axis::node-test[predicate]

- Boolean and comparative operators
- Types
  - Numbers
  - Strings
  - node-sets (the set of nodes selected)
- Functions
  - Examples
    - *boolean* starts-with(*string, string*)
    - *number* count(*node-set*)
    - *number* position()

xpath examples

- http://www.cs.cornell.edu/courses/CS431/2006sp/examples/xpath/base.xml
- /child::source/child::AAA
  - or /source/AAA since child is default axis
- /child::source/child::*[position()=2]
  - or /source/*[2]
- /child::source/child::AAA[position()=2]/attribute::id
  - or /source/AAA[2]/@id
- /child::source/child::AAA/@*
  - or /source/AAA/@*
- /child::source/child::AAA[contains(. ,'a1')]
  - /source/AAA[contains(. ,'a1')]

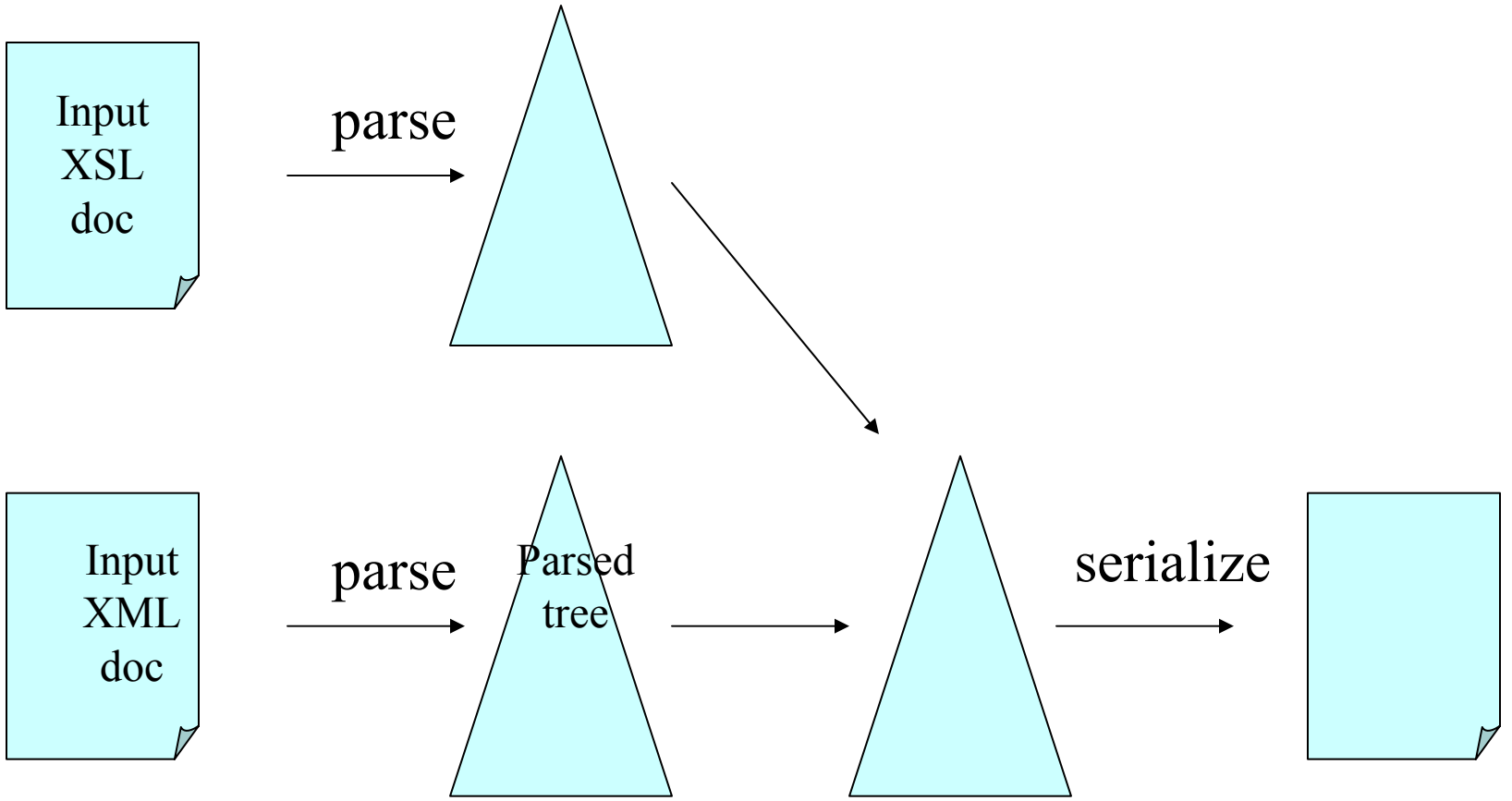
XML Transformations (XSLT)

- Origins: separate rendering from data
  - Roots in CSS
- W3C Recommendation
  - http://www.w3.org/TR/xslt
- Generalized notion of transformation for:
  - Multiple renderings
  - Structural transformation between different languages
  - Dynamic documents
- XSLT – rule-based (declarative) language for transformations

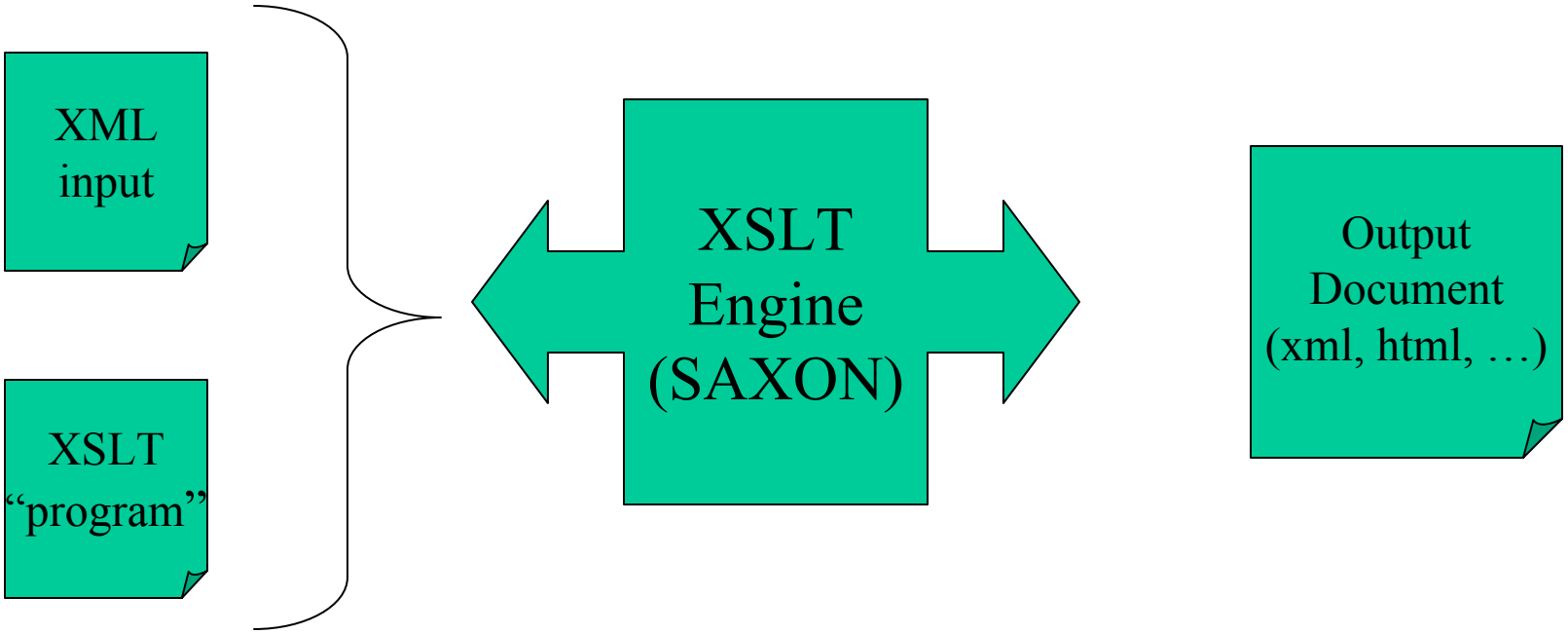
XSLT Capabilities

- Produce any type of document
  - xHTML, XML, PDF…
- Generate constant text
- Filter out content
- Change tree ordering
- Duplicate nodes
- Sort nodes
- Any computational task (XSLT is "turing complete")
  - extra credit if you write an OS in XSLT

XSLT Processing Model



XSLT "engine"


XML
input
XSLT
"program"
XSLT
Engine
(SAXON)
Output
Document
(xml, html, …)

Stylesheet Document or Program

- XML document rooted in <stylesheet> element
- Body is set of templates or rules
  - match attribute specifies xpath of elements in source tree
  - Body of template specifies contribution of source elements to result tree

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
   <xsl:template match="AAA"> [4 lines]
   <xsl:template match="BBB"> [4 lines]
   <xsl:template match="CCC"> [4 lines]
   <xsl:template match="DDD"> [4 lines]
</xsl:stylesheet>
```

Associating an XML document with a transform

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="http://www.cs.cornell.edu/Courses/cs502/2002SP/Demos/xslt/simple.xsl"?>
<para>
  this is
  <emphasis>
    big
  </emphasis>
  text
</para>
```

XSL Execution Model

- Templates represent a set of rules
- Rule matching is done within current tree context
- Rules are not executed in order
- Default behavior is depth-first walk of tree, outputting element values

- http://www.cs.cornell.edu/lagoze/courses/CS431/2005sp/Examples/Lecture10/base.xml
- http://www.cs.cornell.edu/lagoze/courses/CS431/2005sp/Examples/Lecture10/null.xsl

Template Form

```
<xsl:template match="para">
    <p>
        <xsl:apply-templates/>
    </p>
</xsl:template>
```

- Sequential execution within template
- Elements from xsl namespace are transform instructions
- Match attribute value is xpath expression setting rule for execution of body
- Non-xsl namespace elements are literals.
- <xsl:apply-templates>
  - set context to next tree step
  - re-evaluate rules

Result Tree Creation

- Literals – any element not in xsl namespace
- <xsl:text> - send content directly to output (retain whitespaces)
- <xsl:value-of> - expression processing
- <xsl:copy> and <xsl:copyof> - Copy current node or selected nodes into result tree
- <xsl:element> - instantiate an element
- <xsl:attribute> - instantiate an attribute

A simple example

- XML base file
  - http://www.cs.cornell.edu/lagoze/courses/CS431/2005sp/Examples/Lecture10/simple.xml
- XSLT file
  - http://www.cs.cornell.edu/lagoze/courses/CS431/2005sp/Examples/Lecture10/simple.xsl

Modifying rule set and context

- Mode setting
  - <xsl:apply-templates mode="this">
  - <xsl:template match="foo" mode="this">
  - <xsl:template match="foo" mode="that">
- Context setting
  - <xsl:apply-templates select="//bar">
  - Modifies default depth-first behavior
- Conflict resolution rules

- http://www.cs.cornell.edu/lagoze/courses/CS431/2005sp/Examples/Lecture10/elements.xsl
- http://www.cs.cornell.edu/lagoze/courses/CS431/2005sp/Examples/Lecture10/elements2.xsl

XSLT Procedural Programming

- Sequential programming style
- Basics
  - for-each – loop through a set of elements
  - call-template – like a standard procedure call

For-each programming example

- XML base file
  - http://www.cs.cornell.edu/lagoze/courses/CS431/2005sp/Examples/Lecture10/foreach.xml
- XSLT file
  - http://www.cs.cornell.edu/lagoze/courses/CS431/2005sp/Examples/Lecture10/foreach.xsl

Call-template programming example

- XML base file
  - http://www.cs.cornell.edu/lagoze/courses/CS431/2005sp/Examples/Lecture10/call.xml
- XSLT file
  - http://www.cs.cornell.edu/lagoze/courses/CS431/2005sp/Examples/Lecture10/call.xsl

Various other programming constructs

- Conditionals
- Variables (declaration and use)
- Some type conversion
- Sorting