
XPath – Introduction

- XPath is a central part of both XSLT and XQuery
- XPath 1.0 is the official standard, work on XPath 2.0 is in progress
- <http://www.w3.org/TR/xpath>

Examples

- `child::baboon` selects the `baboon` element children of the context node
- `child::*` selects all the element children of the context node
- `child::text()` selects all text node children of the context node
- `descendant::baboon[attribute::colour='red']` selects all the `baboon` element descendants of the context node which have an attribute `colour` with a value of `red`
- `/child::doc/child::chapter[position()=2]` selects the second `chapter` of the `doc` document element

Context

- Evaluation occurs with respect to a *context* consisting of:
 - a context node
 - a context position and context size (integers)
 - a set of variable bindings
 - a function library
 - a set of namespace declarations
- Expression evaluation yields an *object*, which has one of the types: *node-set*, *boolean*, *number*, *string*

Location Paths

- A location path can be *absolute* (starts with a '/') or *relative*
- A location path consists of one or more *steps*, each separated by a slash '/'

Location Steps

Step ::= AxisName '::' NodeTest Predicate*

NodeTest ::= NameTest | NodeType '(' Literal? ')'

- An axis is chosen resulting in a node-set, which is filtered through a *NodeTest* and optional *Predicates*.
- Available axes: ancestor, ancestor-or-self, attribute, child, descendant, descendant-or-self, following, following-sibling, namespace, parent, preceding, preceding-sibling, self
- An axis can be a *forward* axis or a *reverse* axis with respect to document order
- NodeTests:
 - A *NameTest* is true if the node has the specified name
 - *NodeTypes*: comment, text, processing-instruction, node

Predicates

Predicate ::= '[' Expression ']'

- For each node in the node-set, the predicate *expression* is evaluated with that node as the context node
- The result of the expression is converted to a *boolean*
- The meaning of the predicate depends on the chosen axis. Example: `[position()=2]` has different meaning depending on whether a forward axis or reverse axis is used.

Expressions

$\text{PrimaryExpr} ::= \text{VariableReference} \mid \text{'(' Expr ')'} \mid \text{Literal} \mid \text{Number} \mid \text{FunctionCall}$

- *LocationPath* and *PrimaryExpr* are the most important expressions
- Variables and functions are looked up in the evaluation context

Core Function Library

- number **position()** returns the *context position* from the evaluation context
- number **count(node-set)** returns the number of nodes in the argument node-set
- node-set **id(object)** selects an element by its unique ID (attribute declared as type ID in the DTD)
- string **string(object?)** converts an object to a string
- boolean **boolean(object)** converts an object to a boolean
- number **number(object?)** converts an object to a boolean

XPath 2.0

- More datatypes, XML Schema support, strongly-typed
- Sequences instead of node-sets
- Querying using `for`-statements (similar to `select` in SQL)
- 80% of XQuery

XPath and DVM

- How do we find the parent of a node? Since nodes can be shared, each node can have multiple parents.
- A node-set is returned, but how do we know *where* in the document each of these nodes are located? This is important if, for example, the result is to be used in XSLT for transforming the selected nodes.