```
<xqx:module xmlns:xqx="http://www.w3.org/2005/XQueryX">
  <xqx:mainModule>
    <xqx:queryBody>
      <xqx:stringConstantExpr>
        <xqx:value>Hello!</xqx:value>
      </xqx:stringConstantExpr>
    </xqx:queryBody>
  </xqx:mainModule>
</xqx:module>
```

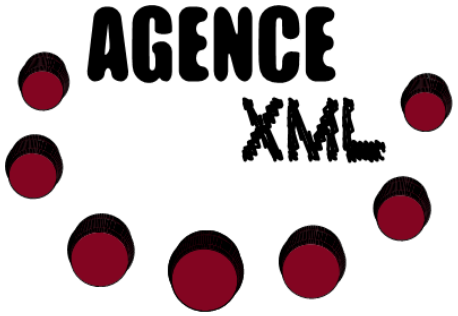
# VISIBLE XQueryX



Declarative Amsterdam 2026

Alain Couthures - agenceXML

# XML Technologies




| | | |
|---|---|---|
| XForms | => | xsltforms.xsl, xsltforms.js, xsltforms.css |
| XQuery | => | fleur.js |
| XProc | => | sbs.ps1 |
| Invisible XML | => | grammix.mjs |

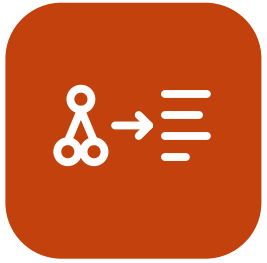
# XQueryX, an XML notation for XQuery

- First release: W3C Recommendation 23 January 2007
- Authors: Jim Melton (Oracle) & Subramanian Muralidhar (Microsoft)
- Java applet written/maintained by Scott Boag (IBM) & Michael Dyck

xqueryx.xsd



xqueryx.xsl



```
a/b ! (. + 7)
```



PARSER

```
<xqx:module xmlns:xqx="http://www.w3.org/2005/XQueryX">
  <xqx:mainModule>
    <xqx:queryBody>
      <xqx:simpleMapExpr>
        <xqx:pathExpr>
          <xqx:stepExpr>
            <xqx:xpathAxis>child</xqx:xpathAxis>
            <xqx:nameTest>a</xqx:nameTest>
          </xqx:stepExpr>
          <xqx:stepExpr>
            <xqx:xpathAxis>child</xqx:xpathAxis>
            <xqx:nameTest>b</xqx:nameTest>
          </xqx:stepExpr>
        </xqx:pathExpr>
        <xqx:pathExpr>
          <xqx:stepExpr>
            <xqx:filterExpr>
              <xqx:sequenceExpr>
                <xqx:addOp>
                  <xqx:firstOperand>
                    <xqx:contextItemExpr/>
                  </xqx:firstOperand>
                  <xqx:secondOperand>
                    <xqx:integerConstantExpr>
                      <xqx:value>7</xqx:value>
                    </xqx:integerConstantExpr>
                  </xqx:secondOperand>
                </xqx:addOp>
              </xqx:sequenceExpr>
            </xqx:filterExpr>
          </xqx:stepExpr>
        </xqx:pathExpr>
      </xqx:simpleMapExpr>
    </xqx:queryBody>
  </xqx:mainModule>
</xqx:module>
```



JS TRANSPILER

```
ctx ⇒ {
  ctx.xqx_pathExpr();
  ctx.xqx_xpathAxis_child();
  ctx.xqx_nameTest("a");
  if (ctx.item.isNotEmpty()) {
    ctx.xqx_xpathAxis_child();
    ctx.xqx_nameTest("b");
  }
  ctx.restoreContext();
  ctx.xqx_simpleMapExpr(
    ctx ⇒ {
      ctx.xqx_pathExpr();
      ctx.xqx_filterExpr(
        ctx ⇒ {
          ctx.xqx_contextItemExpr();
          ctx.fn_data_1();
          ctx.xs_numeric_1();
          ctx.xqx_integerConstantExpr('7');
          ctx.xqx_addOp();
          ctx.fn_data_1();
          ctx.xs_numeric_1();
          ctx.xqx_sequenceExpr(1);
          return ctx;
        }
      );
      ctx.restoreContext();
      return ctx;
    }
  );
  return ctx;
}
```

# Invisible XML

- Declarative grammar to create XML elements and attributes from text
- Operators for alternatives and repeats
- Ambiguity reporting on successful parsing

```
expr: open, -arith, @close, -";".
@open: "(".
close: ")".
arith: left, op, ^right>second.
left>first: operand.
-right: operand.
-operand: name; -number.
@name: ["a"-"z"].
@number: ["0"-"9"].
-op: sign.
@sign>operator: "+"; "-".
```

# XQuery to XML with Invisible XML from EBNF



**jωiXML processor**

*A SaxonJS/JavaScript processor of Invisible XML*
*Workbench Version 1.7.1 Processor Version 1.2.1 (TypeScript)*
*Compiled 2026 May 18 @ 14:18 using Saxon-EE 12.4;*
*Running under SaxonJS 2.6*

```
 {
 Reduced tree version of https://raw.githubusercontent.com/qt4cg/qtspecs/master/specifications/grammar-40/xpath-gram
    Generated at 2024 Sep 24 @ 14:48
}
ixml version "1.1+".
 {Grammar for XQuery 4.0
Generated from https://raw.githubusercontent.com/qt4cg/qtspecs/master/specifications/grammar-40/xpath-grammar.xml
at 2024 Sep 24 @ 14:48
}
 {Module altered to permit leading whitespace}
                    Module: s, VersionDecl?, (LibraryModule; MainModule).
                VersionDecl: -"xquery", RS, ("encoding", RS, StringLiteral; "version", RS, StringLiteral, ("encodi
                 MainModule: Prolog, QueryBody.
              LibraryModule: ModuleDecl, Prolog.
                 ModuleDecl: -"module", RS, -"namespace", s, NCName, -"=", s, URILiteral, Separator.
                     Prolog: ((DefaultNamespaceDecl; Setter; NamespaceDecl; Import), Separator)*, ((ContextValueDe
                  Separator: -";", s.
                    -Setter: BoundarySpaceDecl |
```

# XQuery EBNF `/* xgc: reserved-function-names */`

Syntax for node tests is the same as the syntax for function calls

```
attribute
comment
document-node
element
namespace-node
node
processing-instruction
schema-attribute
schema-element
text
```

# XQuery EBNF `/* xgc: leading-lone-slash */`

The XQuery parser must consider the longest possible path expression

```
/*/a
```

```
<xqx:pathExpr>
  <xqx:rootExpr/>
  <xqx:stepExpr>
    <xqx:xpathAxis>child</xqx:xpathAxis>
    <xqx:Wildcard/>
  </xqx:stepExpr>
  <xqx:stepExpr>
    <xqx:xpathAxis>child</xqx:xpathAxis>
    <xqx:nameTest>a</xqx:nameTest>
  </xqx:stepExpr>
</xqx:pathExpr>
```

```
map{a:b:c}
```

```
<xqx:mapConstructor>
  <xqx:mapConstructorEntry>
    <xqx:mapKeyExpr>
      <xqx:pathExpr>
        <xqx:stepExpr>
          <xqx:xpathAxis>child</xqx:xpathAxis>
          <xqx:nameTest xqx:prefix="a">b</xqx:nameTest>
        </xqx:stepExpr>
      </xqx:pathExpr>
    </xqx:mapKeyExpr>
    <xqx:mapValueExpr>
      <xqx:pathExpr>
        <xqx:stepExpr>
          <xqx:xpathAxis>child</xqx:xpathAxis>
          <xqx:nameTest>c</xqx:nameTest>
        </xqx:stepExpr>
      </xqx:pathExpr>
    </xqx:mapValueExpr>
  </xqx:mapConstructorEntry>
</xqx:mapConstructor>
```

# XQuery EBNF `/* xgc: occurrence-indicators */`

The occurrence indicator binds to the closest type

```
local:f() instance of function() as xs:integer*
```

```
<xqx:instanceOfExpr>
  <xqx:argExpr>
    <xqx:functionCallExpr>
      <xqx:functionName xqx:prefix="local">f</xqx:functionName>
    </xqx:functionCallExpr>
  </xqx:argExpr>
  <xqx:sequenceType>
    <xqx:typedFunctionTest>
      <xqx:sequenceType>
        <xqx:atomicType xqx:prefix="xs">integer</xqx:atomicType>
        <xqx:occurrenceIndicator>*</xqx:occurrenceIndicator>
      </xqx:sequenceType>
    </xqx:typedFunctionTest>
  </xqx:sequenceType>
</xqx:instanceOfExpr>
```

# XQueryX structure

Permissive schema for an empty element or no element when there are no children

```
<a/>
```

```
<xqx:elementConstructor>
  <xqx:tagName>a</xqx:tagName>
</xqx:elementConstructor>
```

```
<xqx:elementConstructor>
  <xqx:tagName>a</xqx:tagName>
  <xqx:elementContent/>
</xqx:elementConstructor>
```

# XQueryX structure

Some extra intermediate elements

1+2

```
<xqx:addOp>
  <xqx:firstOperand>
    <xqx:integerConstantExpr>
      <xqx:value>1</xqx:value>
    </xqx:integerConstantExpr>
  </xqx:firstOperand>
  <xqx:secondOperand>
    <xqx:integerConstantExpr>
      <xqx:value>2</xqx:value>
    </xqx:integerConstantExpr>
  </xqx:secondOperand>
</xqx:addOp>
```

# XQueryX structure

Reduced expression tree

```
(1-2)*3
```

```
<xqx:multiplyOp>
  <xqx:firstOperand>
    <xqx:subtractOp>
      <xqx:firstOperand>
        <xqx:integerConstantExpr>
          <xqx:value>1</xqx:value>
        </xqx:integerConstantExpr>
      </xqx:firstOperand>
      <xqx:secondOperand>
        <xqx:integerConstantExpr>
          <xqx:value>2</xqx:value>
        </xqx:integerConstantExpr>
      </xqx:secondOperand>
    </xqx:subtractOp>
  </xqx:firstOperand>
  <xqx:secondOperand>
    <xqx:integerConstantExpr>
      <xqx:value>3</xqx:value>
    </xqx:integerConstantExpr>
  </xqx:secondOperand>
</xqx:multiplyOp>
```

# Whitespace issues

Whitespace implicitly allowed between EBNF terms

```
-s: (whitespace; comment)*.
-RS: (whitespace; comment)+.
-S: whitespace+.
-whitespace: -[" "; #9; #a; #d].
```

- Consecutive whitespace non-terminals create ambiguities
- Required whitespace before and after operators?

# Limitations of a context-free parser

Checking that element constructors are well-formed: comparing the end tag with the start tag

```
<a> <b>   </a>
```

# Limitations of a context-free parser

Processing boundary space according to the prolog

```
  <a> <b/>  </a>
```

```
<xqx:elementConstructor>
  <xqx:tagName>a</xqx:tagName>
  <xqx:elementContent>
    <xqx:elementConstructor>
      <xqx:tagName>b</xqx:tagName>
    </xqx:elementConstructor>
  </xqx:elementContent>
</xqx:elementConstructor>
```

```
declare boundary-space preserve;
  <a> <b/>  </a>
```

```
<xqx:elementConstructor>
  <xqx:tagName>a</xqx:tagName>
  <xqx:elementContent>
    <xqx:stringConstantExpr>
      <xqx:value> </xqx:value>
    </xqx:stringConstantExpr>
    <xqx:elementConstructor>
      <xqx:tagName>b</xqx:tagName>
    </xqx:elementConstructor>
    <xqx:stringConstantExpr>
      <xqx:value>  </xqx:value>
    </xqx:stringConstantExpr>
  </xqx:elementContent>
</xqx:elementConstructor>
```

# Namespaces in Grammix

```
xqx:module xmlns:xqx=http://www.w3.org/2005/XQueryX
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance":
   xsi:schemaLocation, s, (xqx:versionDecl, s)?,
   (xqx:libraryModule; xqx:mainModule), s.
@xsi:schemaLocation:
   +"http://www.w3.org/2005/XQueryX http://www.w3.org/2005/XQueryX/xqueryx.xsd".
xqx:versionDecl:
   -"xquery", RS,
   (-"encoding", s, ^stringValue>xqx:encoding;
    -"version", s, ^stringValue>xqx:version,
   (s, -"encoding", s, ^stringValue>xqx:encoding)?),
   separator.
xqx:mainModule:
   (xqx:prolog, s)?, ^expr>xqx:queryBody.
xqx:libraryModule:
   xqx:moduleDecl, (s, xqx:prolog)?.
```

# Subtraction operator in Grammix

```
xqx:attributeList:
   S,(xqx:attributeConstructor ¬ xqx:namespaceDeclaration;
      xqx:namespaceDeclaration)++S.


xqx:multiplyOp:
   (^multiplicativeOp>xqx:firstOperand ¬ "/"), s,
   -"*", s, ^unionOp>xqx:secondOperand.


xqx:functionName:
   EQName ¬ ("attribute"; "comment"; "document-node"; "element";
             "namespace-node"; "node"; "schema-attribute";
             "schema-element"; "processing-instruction"; "text").
```

# XQuery postprocessing in Grammix

```
-decimalEntityRef>{char(parse-integer(.))}:
    -"&#", ["0"-"9"]+, -";".
-hexadecimalEntityRef>{char(parse-integer(., 16))}:
    -"&#x", ["0"-"9";"A"-"F";"a"-"f"]+, -";".



xqx:elementConstructor:
   -"<", ^QName>xqx:tagName, xqx:attributeList?, S?,
   (-"/>"; -">", (xqx:elementContent; boundaryOnlyContent)?,
   endTag>{if(deep-equal(preceding-sibling::xqx:tagName, .))
           then ()
           else error(xs:QName('err:XQST0118'))}).
-endTag: -"</", ^QName>xqx:tagName, S?, -">".
```

# XQuery postprocessing in Grammix

```
-boundaryOnlyContent>{
   if(ancestor::xqx:mainModule/xqx:prolog/xqx:boundarySpaceDecl eq 'preserve')
   then <xqx:elementContent>
           <xqx:stringConstantExpr>
              <xqx:value>{string(.)}</xqx:value>
           </xqx:stringConstantExpr>
        </xqx:elementContent>
   else ()
   }: boundaryWhitespace.
```

# Extended Invisible XML grammar in Grammix

```
          rule: naming, (RS, ns)*, -["=:"], s, -alts, -".".
       -naming: (mark, s)?, name, s, (-">", alias, s; -">{", postprocessing, -"}", s)?.
postprocessing: type, xqx:module.
         @type: +"application/xqueryx+xml".
         @name: namestart, namefollower*, (":", namestart, namefollower*)?.
    -namestart: ["_"; L].
 -namefollower: namestart; ["-.#·‿⁀"; Nd; Mn].
        @alias: name.
           ns: -"xmlns", (-":", prefix)?, s, -"=", s, uri.
       @prefix: namestart, namefollower*.
          @uri: string; -'"'"; -"'".
```

# Parsing XQuery 3.1 with Grammix

- ~ 300 Invisible XML rules
- ~ 2,000 elementary Earley rules
- XQueryX grammar is de facto added to the iXML grammar
- HTML entities resolved with rules
- 99.4% success against the XQueryX part of the XQuery 3.1 Test Suite (XML Schema validation and smart node comparisons)

# Conclusion

- An iXML grammar for XQuery-XQueryX now exists!
- Most ambiguities disappeared without XQuery postprocessing
- XQueryX becomes more accessible for all uses!
- Invisible XML for another well-known XML notation?
- Postprocessing for Invisible XML's own grammar?

# Next steps!

- Grammix in Rust with libxml2
- XPath 3.1/4.0 functions as a Rust library for libxml2 and libxslt
- XSLTX and XProcX as XSLT 1.0 stylesheets for Rust-extended libxslt
- A text notation for XSLT and XProc?
- Rust libxml3 including XQuery
- Rust libxslt3, libxproc3