EBNF Change Proposal

Scott Boag

New Test Parser

- Original Intent was to produce parser without altering the grammar source.
  - In practice this proved to difficult.
  - Was supposed to be post-CR
- Parser NOT part of CR release.
- Parsers now pass all known tests for all grammars.
- XQueryX Translator in Progress

Long Tokens

- Example: <"for" "$">
- Needed to:
  - BUG: Distinguish keywords from QNames.
    - BUT, only in “states” that have QNames!
      - BUT, “states” are not part of our specs!
  - Determine branch choices in grammar.
    - Lookahead is where you branch, *some* long tokens are what is being looked for.
- Affect EBNF structure:
  - Example: (<"empty" "greatest"> | <"empty" "least">)
    - Is better expressed as (“empty” (“greatest” | ”least”))
- ARE NOT NORMATIVE!
- Bug: Use of long tokens prevents (: (: nested :) comments :) in the white space of those tokens if matched with Regular Expressions.
  - I’ll be glad to go into deep detail why this bug is so hard to solve with existing parser construction tools!
- Short tokens make a lexical state machine even harder and trickier.
  - Though, I admit, not impossible.

Long Tokens Problem Example

- <"declare" "namespace"> Used for look ahead AND to distinguish “declare” as a keyword.
- <"at" URILiteral> Merely used to distinguish “at” as a keyword in the lexical state coming off of URILiteral.

EBNF Change 1: No Angle Brackets

- CastableExpr (<"treat" "as"> SequenceType)?
  Becomes:
  CastableExpr ("treat" "as" SequenceType)?

- Long tokens no longer have an real meaning!
- In any case, they are dependent on lexical states, which are not part of the document, and so should be removed in any case!!!
- Change is not strictly required.
  - Could leave these in, but they're no longer tested!

EBNF Change 2: minor reordering of some A | B Constructs

- VersionDecl? (MainModule | LibraryModule)
  becomes:
  VersionDecl? (LibraryModule | MainModule)

- This helps the test parser generation, because, with lookahead, MainModule must be tested last, since it contains the “fallback” QName case.
- Clearly A | B Constructs are unordered in EBNF
  - Spec only says: matches A or B but not both.
- Change not required... Could do this ordering for the parser with an attribute and leave the ordering as it was.
  - But why?

EBNF Change 3: Some local restructuring

- (<"declare" "default" "element"> |
  <"declare" "default" "function">) "namespace" URILiteral
  becomes:
  "declare" "default" ("element" | "function")
  "namespace" URILiteral
- Local ONLY, never done across production boundaries!
- More opportunities, but only did a couple obvious cases.
- Change not required: Simply Reduces Lookahead overhead for test parser!
  - But the change clearly helps the readability of the BNF.

EBNF Change 4: Superscript Lookahead Hints?

- [28] StepExpr ::= AxisStep | FilterExpr
  Becomes:
  [28] StepExpr ::= FilterExpr^{L3} | AxisStep

- Hints at likely lookahead points for parsers
- Non-normative.
- CC notation for already-documented ambiguities
  - ("/" $\text{RelativePathExpr?}^{CC}$) ... /* *xgs: leading-lone-slash* */
- Not required: they're only helpful hints.
  - Still depends on tokenization strategy: If tokenization strategy for Qname is to break it into three tokens: NCName ":" NCName, for instance, then lookahead for FunctionCall changes from L2 to L4.
  - Direct constructors don't use lookahead, but some implementations might!
  - On balance, I tend to think we should not do this.

Summary

- NO changes to what is a legal XQuery sentence are being suggested!
- No extra work for editors!
- Pre CR-Options:
  a) No change to xpath-grammar.xml source
     - In which case, we have to determine how to proceed post-CR!
  b) New xpath-grammar.xml, but EBNF must not change in even trivial ways.
  c) New xpath-grammar.xml, remove angle brackets, but no other EBNF change. (1)
  d) New xpath-grammar.xml, remove angle brackets, allow trivial choice reordering and restructure (1, 2, 3)*
  e) New xpath-grammar.xml, remove angle brackets, allow trivial choice reordering and restructure, and lookahead hints (1, 2, 3, 4)
  f) None of the above.

  * Since xpath-grammar.xml has changed for the good, and new insights have been gained about so-called long tokens, I recommend option d.

Information

- Proposed xpath-grammar.xml:
  - http://www.w3.org/XML/Group/xsl-query-specs/grammar/parser/applets-llk/xpath-grammar.xml
- The doc with the changes are available for review at:
  - http://www.w3.org/XML/Group/xsl-query-specs/grammar/llkdocs/xquery.html#id-grammar
  - http://www.w3.org/XML/Group/xsl-query-specs/grammar/llkdocs/xpath.html#id-grammar
  - http://www.w3.org/XML/Group/xsl-query-specs/grammar/llkdocs/shared.html
  - http://www.w3.org/XML/Group/xsl-query-specs/grammar/llkdocs/shared-semantics.html
- Somewhat problematic diff version for XQuery:
  - http://www.w3.org/XML/Group/xsl-query-specs/grammar/llkdocs/LLKCompare.htm
- Grammar Applets:
  - http://www.w3.org/XML/Group/xsl-query-specs/grammar/parser/applets-llk/xqueryApplet.html
  - http://www.w3.org/XML/Group/xsl-query-specs/grammar/parser/applets-llk/xpathApplet.html
  - http://www.w3.org/XML/Group/xsl-query-specs/grammar/parser/applets-llk/xquery-fulltextApplet.html
  - http://www.w3.org/XML/Group/xsl-query-specs/grammar/parser/applets-llk/xquery-updateApplet.html
  - http://www.w3.org/XML/Group/xsl-query-specs/grammar/parser/applets-llk/xquery-coreApplet.html

Backup Slides

Background

- Grammar specified in XML source produces both test parser and EBNF.
- Test parser is used to verify and experiment with the grammar.
  - And used by testing taskforce.
  - And provide starting code for casual implementers.
- Bug: Use of long tokens prevents (: comment :) in the white space of those tokens.
  - Regular Expressions can’t match nested comments!
- Grammar still being developed for Update, Fulltext.

Advantages of new xpath-grammar.xml

- Only DEFAULT lexical state, except for XQuery Direct Constructors!
- Trivial strings do not need to be named.