ECMA International

Standardizing Information and Communication Systems

E4X Meeting Notes 2003-04-24

Date	April 24, 2003 10:00 AM – 12:00 PM
Location	Conference call
	888-659-6004 access code 361330
Convener	Rok Yu (Microsoft)
Editor	John Scheider (BEA/AgileDelta)
Participants	Jeff Dyer (Compiler Company)
	John Schneider (BEA/AgileDelta)
	Michael Shenfield (RIM)
	Rok Yu (Microsoft)
	Waldemar Horwat (Netscape)

Agenda

- [05 mins] Gather and get started
- [05 mins] Review and revise agenda

Agreed to agenda with one addition:

- Discuss data model for text nodes – escaped or unescaped

Complete review of Section 9.5 Assignment Operators

- [30 mins] Complete review of Section 9.5 -- assignment operators
    - o 9.5.1 XML assignment operator
    - o 9.5.2 XMLList assignment operator

9.5.1 XML Assignment Operator

Rok and Waldemar think that this section is confusing because the format and production make it appear that the section describes a change in the assignment operator semantics, but instead, there is no change and the text gives examples and describes consequences of the definitions of [[Put]] in 6.1.1.2. The suggestion is to move the examples to the [[Put]] section, or group non-normative descriptive text into a separate overview section.

John understands the confusion and will investigate the options to decide which choice is more appropriate.

Rok notes that the examples in this section include assignment to not only to XML, but to XMLList.

John agrees that if there XML and XMLList descriptive text is separated, the examples will be separated. If they are together, the examples will be annotated to describe which semantics applies.

Action Items

- John to investigate moving 9.5.1 and 9.5.2 to corresponding [[Put]] sections and/or separate non-normative Overview section.
- John to either split out XML/XMLList cases in samples, or annotate samples to indicate which applies.

9.5.2 XMLList Assignment Operator

Comment 9.5.1 regarding confusion of text applies to this section as well.

Waldemar notes that there is inconsistent use of base and parent.

John agrees and will convert all uses to base.

Rok suggests that description in the overview regarding how XMLList behaves with a non-array index properties may be rephrased to use delegate instead of convert, as the actual normative [[Put]] algorithm doesn't convert to XML.

Waldemar and Rok think the phrase "string representation of a number" is ambiguous, and that the term "array index" can be used instead. It is described in section 15.4 in E3 defines array index

Action Items

- John to convert uses of "parent" to "base"
- John to edit process description for non-array-index properties to closer match [[Put]] algorithm
- John to replace instances of "string representation of a number" to "array index" throughout document

9.5.3 Compound Assignment (op=)

No comments

Section 10 Statements

- [20 mins] Section 10 -- statements

10.1 Namespace Statement

Waldemar notes that the URI in the syntax has no definition and should be replaced with String.

NamespaceStatement :

namespace *Qualifier* = *URI*

Waldemar suggest that instead of given syntax, we could use the string in parentheses. Advantages of this option are that it doesn't look like your assigning something to Qualifier, and it's extensible in case we wish to add more state in the future.

Everyone agrees to the change and John will make it.

Rok asks whether this section isn't really critical for E4X and we can cut it as the primary motivation for adding this was to unify the namespace concepts with those found in E4.

Everyone agrees we can nuke as E4 has already been modified to include the compile time and runtime namespace concepts of E4X.

Action Items

- John to remove this section

10.2 Use Namespace Statement

Waldemar notes that the extent of the use statement is not captured in the specification and that linking to the scope chain doesn't work because blocks don't push scopes.

Rok points out that the closest concept in E3 is the with statement, but it relies on the fact that the extent is scoped to the next statement. The use statement cannot work the same way because we use statements to be in effect until the end of the block, and not just the next statement.

Waldemar indicates there are lots of places that need to be modified if we want to change the grammar to pop the use statements.

Rok suggests we may be able to handle this by using prose to describe the desired normative behavior.

John agrees to rework this section.

Action Items

- John to rewrite semantics to account for extent of use statement.

10.3 The For-in Statement

Waldemar notes that a decision on whether to execute the for..in algorithm specified here, or the for..in algorithm specified in E3 isn't made and suggest that the algorithm be rewritten to subsume the algorithm in E3, and dispatch based on the type of object being iterated.

Waldemar notes that no wording describes what happens if the object is mutated while enumerating and recommends leaving the behavior unspecified as in E3.

Rok notes that the specification doesn't include semantics for XML.

John states there is still some discussion that needs to occur before these semantics are written because of the boundary case of an XMLList with 1 element.

Action Items

- John to rewrite algorithm based to test object being iterated and dispatch to either existing E3 algorithm or new algorithm.
- John to add text as in E3 indicating enumeration behavior is unspecified if enumerated object is mutated during enumeration.
- John to bring proposal for enumeration of XML

Discuss Issues List

- [45 mins] Discuss the following items from the Issues list (roughly listed in order of increasing difficulty)
  - 17 | Ability to test for descendents with dynamically computed property name

John proposes adding a descendant method to handle this.

Everyone accepts this proposal.

There is some confusion as to the appropriate spelling of descendant (alternative spelling descendent). Everyone agrees that spec will use whatever XPath chose.

Action Items

- John to factor out descendent operator behavior into a separate section which is called from both the syntactic invocation ("..” operator) and the method.
- John to look up spelling of descendant in XPath and apply to entire spec.
    - 19 | Ability to test for children inside filtering predicate with dynamically computed property name

John propose to have a child method to handle this. This proposal assumes there is a way to access methods/members on the context object within a filter – something still under discussion.

Everyone accepts this proposal. John will include this when OM section is written.

- 02 | Necessity of parent pointer

Waldemar and Rok are still uneasy about the parent pointer. We agree to bring the concerns to the next meeting.

Action Items

- Rok and Waldemar to think about parent pointer and reasons it shouldn't be part of E4X.
    - 09 | Comparing XML encoded strings

Waldemar describes a use case he's run into where he would like a canonical string representation for XML. He is writing scripts that may run on different platforms and saves the XML to a file managed by a source control system.

Waldemar also mentions use case of hashing XML objects.

This leads group into the longer discussion of whether order in attributes should be preserved. Existing data model is that attribute order is unspecified.

John thinks implementations will preserve and control order of attributes. We will discuss ordering again at next meeting.

To close on the issue, John asks if we could live with having a method separate from toString that guarantees a particular order for attributes – e.g. toCanonicalString.

Everyone agrees that this is a livable compromise.

Action Items

- Everyone to revisit data model to consider having data model preserver order.
    - 05 | Do XMLLists store lvalues or rvalues

- 01 | Are attributes modeled as objects or named, string valued properties
- 01 | Discuss data model for text nodes – escaped or unescaped

We skipped these agenda items because we were out of time.

Next Steps (Statements & Built-ins)

- [15 mins] Next steps: Built-in XML object

Waldemar is confused because some arguments to the member functions imply a single XML argument while and imply an XMLList.

John states that all the insertion methods can be applied to either XML or XMLList.

Waldemar notes that the examples with "x =" don't do what is expected because they just replace the contents of x. Rok concurs and points out that in the examples, the XML [[Put]] is never called.

John agrees that examples need to be rewritten so assignment is to a qualified left hand side.

Waldemar suggests innerXML be renamed setInnerXML.

Everyone agrees to this change.

Rok asks how the methods were chosen because some methods equivalent easy mechanisms for accomplishing. e.g. x.attributes() is completely the same as x.*.

John responds methods are there so users of XML DOM have familiar methods to call.