TC39 TG1 E4 Meeting Notes 2003-05-22

Date	May 22, 2003 10:00 AM – 12:00 PM
Location	Conference call 888-659-6004 access code 361330
Convener	Rok Yu (Microsoft)
Editor	Waldemar Horwat (Netscape)
Participants	Jeff Dyer (Macromedia) John Schneider (BEA/AgileDelta) Rok Yu (Microsoft) Waldemar Horwat (Netscape) Markus Scherer (IBM)

Announcement

Jeff has recently been hired by Macromedia and will be representing them at the TC39 TG1 meetings.

Update on Unicode Support

Waldemar states he is having difficulty adding support for supplementary Unicode characters. He has encountered issues involving lexing, case conversions, case insensitve matches, and regular expression matching.

Issue 1: Adding supplementary code point support to lexer

Waldemar is having difficulty modifying the lexer to add support for characters in the supplementary range because the lexer is hard wired to use lists of LISP character type which are essentially 16 bit code units.

Waldemar is approaching this problem by defining a semantic domain that is a union of code units and code points in the supplementary range.

Action Items

- Markus will look at algorithms and suggest implementation options.

Issue 2: Behavior of supplementary code points in case insensitive match and case conversions

Waldemar states that in E3, general rule was to avoid case mappings where a single code point gets mapped into more than one code point.

Markus states that Unicode defines the concept of simple case mapping and full case mapping. The former is context insensitive and never expands a single code point to multiple code points, the latter may map from one character to multiple characters and be context sensitive – i.e. convert to different character depending on whether its at the end of the word, etc. Markus states that in Unicode 4.0 all supplementary characters there is no full case mapping for supplementary code points - only a simple case mapping. He adds that in IBM's implementation of regexp, they are only supporting simple case mapping and this should be sufficient for E4.

Waldemar agrees that E3 is essentially simple case mapping and will do the same in E4. He adds that in E3, a special case was that ASCII characters would never map to non-ASCII characters.

Markus states that this is legitimate to continue with the one exception being for Turkish.

Rok suggests E4 keep E3 behavior w/ prose to describe behavior in supplementary character range.

The working group agrees to this proposal.

Issue 3: Handling code points in surrogate pairs ranges in regular expressions

Waldemar states that adding support for supplementary characters will require us to decide on how to deal with code points in the surrogate pair ranges. He suggests three possible options.

1. Don't support supplementary characters – regular expressions continue to behave exactly as in E3 and matches are against 16 bit code units.
2. Partial solution – do a pre-pass on the pattern and text to combine surrogate pairs into equivalent supplementary code points. Some regular expressions which used to match code points in surrogate pair ranges will stop working. e.g. /\uDC01[\uDD02-\uDD10]/
3. Do full processing to match surrogate pairs against corresponding supplementary characters in all cases. Waldemar doesn't know of any feasible way to implement this in time remaining.

Rok and John believe there's insufficient time to implement the regular expression support for supplementary code points and vote to keep regular expression behavior identical to E3. Waldemar thinks he can do either option 1 or 2. Markus abstains.

The working group agrees to punt this work and keep regular expression support identical to that in E3. In particular \U will be forbidden in regular expression patterns.

Behavior of ToNumber

Waldemar asks the question should ToNumber ever produce a long or ulong, or should it only every return IEEE doubles? In addition, what happens when values are coerced directly to long or ulong?

Rok states that in Microsoft's implementation, ToNumber always returns double and direct coercions to long and ulong conver directly and don't indirect via double.

The working group agrees to this behavior

Prototype Based Objects

Waldemar has removed distinction between prototype based objects and class based objects. Objects support expando behavior if the dynamic attribute is set. The root class Object has the bit set. Classes inherit from their base class except for the special case of Object.

Waldemar asks whether prototype objects should have the dynamic attribute set (i.e. whether they are sealed). Waldemar proposes to keep this implementation dependent.

Rok notes that in Microsoft's implementation, there are cases where they are sealed (server) and cases where they aren't (client).

The working group agrees to keep the setting of the dynamic attribute on built in prototype objects implementation dependent.

Waldemar asks whether methods should be immutable.

Rok states that it's pretty common for users to overwrite some methods. As an example, users often replace toLocaleString to give back better results then what is supplied in the implementation.

Action Items

- Members to come up with list of methods that should be mutable

Documentation Reviews

Waldemar led the working group through a high level review of sections 12.5 through 12.21.

John asks whether the grammar in 12.5 Object Literals supports the nesting of object literals.

Waldemar states yes – nested literals are derived via the expansion through AssignmentExpression. There's quite a few productions that must be applied, but it eventually allows for ObjectLiteral.

Waldemar advises the working group to pay close attention to the following:

- PostFixExpressions for the precedence of new expressions and attribute expressions.
- Algorithm for instanceof
- Equality and relational operators. These are inherited from E3 with the modification that it allows char('c') == "c".

Waldemar states all other semantics should be the same as E3 with exception of bitwise operations which have been extended to do 64 bit operations on long/ulong values.

Next Meeting

For next face to face meeting, we will:

- Review issues that Waldemar hits from writing the algorithms for the built in object model.
- Final sign off of sections 12.1 through 12.4
- Detailed walk through of remainder of 12 and13
- High level over view of sections 14, 15, and 10.