ECMA/TC39/97/12

>----------
>From: Wiltamuth S.
>Sent: samedi, 1. mars 1997 07:18
>To: e-tc39
>Subject: Notes from the 2/28 working group meeting
>
>
>
>In attendance
>---
>* Guy from Sun
>* Brendan from Netscape
>* Clayton from Netscape
>* Shon from Microsoft
>* Scott from Microsoft
>* Randy from Borland
>* Richard Gabriel from Sun
>* Michael from IBM
>
>Agenda:
>* Discussion of changes from last time
>* Object model discussion (functions, prototypes, etc)
>* Discussion of open issues
>
>
>Next meetings, etc.
>---
>3/7 Teleconference at 11 am - 1 pm West Coast time. Scott will arrange
>this.
>3/11 Guy will have a draft of the document for distribution. This is
>the doc for the 3/18 meeting.
>3/14 Normal working group meeting, 1 pm - 6 pm JavaSoft again. Guy
>will arrange this.
>
>
>

>Discussion of changes from last time
>---
>3.3.3 "+" is missing from the table. It appears that "_" accidentally
>took its place.
>
>4.4 The value -1073 should be -1074. Also, for denormals, the right
>value is 2^52 rather than 2^53.
>
>5.7 In this area:
>   If m is an integer less than XX, then it is represented as that
>integer
>   value in decimal form with no leading zeroes and no decimal point.
>We decided on XX = 10^21 rather than 10^16.
>
>Similarly, for small numbers we decided on a bound of 10^(-6)
>
>
>7.2 Shon noted that the grammar as it currently stands does not allow
>   x = new new y
>and proposed a new grammar. Everyone agreed on this, so we adopted
>this
>proposal.
>
>7.2.2 Step 4. We should invoke the IEEE standard here. Same thing for
>the prefix operators.
>
>7.3.1. Step 7 should be Return Result(6) rather than Return Result(7).
>
>7.12.1 and 7.12.2. The "UnaryExpression"'s used should really be
>"PostFixExpression"'s.
>
>
>A.1 This is Guy's proposal for "Break and continue label stacks."
>Everyone liked this mechanism for describing the language, so Guy is
>going to go ahead and use this throughout the specification.
>
>A.4. Escape sequences in string literals. We will not allow new lines
>or carriage returns within a string, even if preceded with a backslash.
>

>A.5. Break, Continue, Return in wrong place. We will treat these as
>syntactic (meaning compilation) errors rather than semantic (meaning
>runtime) errors. Thi sis not a general rule. There are errors that
>are
>catchable at compilation time that we specify as runtime errors. Since
>we specify the latest time at which an error can occur, it is possible
>for an implementation to decide to err at compilation time instead.
>This means that it is possible for a program to run correctly on one
>implementation and not run at all on another. This coul dhappen if the
>program contains an error in code that will never actually be executed.
>
>
>
>General comments
>--
>5.5 depends on an undefined % operator.. We should just spell this out
>instead of depending on this operator.
>
>7.3.7. Typo -- the tilda (~) should be an exlamation point (!)
>instead.
>
>7.4.3 This statement:
>   "If the dividend is a zero and the divisor is finite, the result is
>zero."
>needs to handle the negative 0 case.
>
>7.5.3. Need to handle the negatve zero case again. Including -0 + -0.
>
>8.5. For this case:
>   for (var VariableDeclarationList ; Expressionopt ; Expressionopt)
>Statement
>two questions:
>*   Do we allow an initializer for the var being defined?
>*   Can a full variable declaration list be used?
>
>
>8.5 For this case:
>   for (Expressionopt ; Expressionopt ; Expressionopt) Statement
>We should allow PostFixExpression.

>
>
>
>Object model discussion
>--
>I didn't take significant notes on this part of the meeting because for
>the most part we were agreeing on and explaining a conceptual model
>rather than trying to settle issues. My limited notes are provided
>below.
>
>Hints.
>* Hints are really hints, not requirements.
>* Discussion of what happens when you get the default value of an
>object
>and an object is returned.
>
>Hints and addition. As spec'd, we hint number and if at least one arm
>of the addition returns a string, then string concatenation is done.
>Instead, we can supply no hint in this case. For internal objects this
>will not cause any changes. External objects can return their
>preferred
>type.
>
>
>
>Open issues
>--
>* Versioning.
>
>* Non-1970 dates. We will discuss this as an issue. We will not
>discuss a date type. Waiting for a proposal from Randy. Shon raised
>an
>interesting sub-issue -- that you can do SetYear(96). Does this mean
>96
>A.D. or 1996?
>
>