ECMA

Standardizing Information and Communication Systems

ECMA/TC37/97/10

From: Wiltamuth S.

>Sent: mardi, 11. février 1997 22:30

>To: e-tc39

>Subject: Notes from the 2/11 conference call>

>

>

>Next meetings

>--

>2/14 in-person at Netscape 1-6 pm, organized by Clayton Lewis. Clayton to provide details.

>

>

>Review status of contributions

>--

>Contributions that have been made already but haven't reviewed in detail:

>* Guy's ToInt32, ToUInt32 proposal. Shon and Guy prefer Mod2^32 option because this allows intermingling arithmetic and bitwise operators. In the rounding step, round toward 0 rather than applying floor. Other than that, what is in the .7 version of the doc is what we want.

>* Guy's more rigorous definition of the Number type. This section is affected by the ToInt32 and ToUInt32 decision.

>* Shon's revised scoping section (Chapter 6).

>* Scott's high-level proposal with regard to versioning * Clayton's proposal for numeric literal semantics. E.g., the

>0x111111111111111111111111111 issue. Shon's coment -- the # of digits is wrong.

>

>

>Randy's to-do list:

>* Working on object model section. Will transfer to Guy by Friday.

>* Proposal for extending the range of dates. Will have for Friday.

>

>Brendan's to-do list:

>* ToString(Number)

>* Arguments object and activation object.

>

>

>Stepping through the revisions from last time

>--

>For the most part, I did not take notes on this part of the call.

>Mostly minor comments.

>

>A few particular items:

>* Lengthy discussion on the condiitonal operator. We will do

>   LogicalORExpression ? Expression : AssignmentExpression as it is in the doc today. There was a conflict between prior art in JavaScript/JScript and similarity with Java, C, C++. We went with compatibility for this case.

>

>

>Added to the proposed extensions list

>---

>* A precise numeric type -- some form of decimal rather than floating point. Useful for money.

>

>

>Resolved issues

>---

>* typeof(x) and typeof x. As currently spec'd, these two expresions are different. This seems bad. Agreed that we should fix this so that they are the same. The remainging discussion is wrt how to handle this in the doc.   The grouping operator will no longer do GetValue. The result of a parenthesized expression maybe an lvalue. This causes a change in 7.1.4.

>

>* 7.4 Zeroes. We will keep signed zeroes and add a function that allows a user to determine if the result of an operation is finite. NaN's will not be considered finite.

>

>

>Open issues (these are unmodified from my last set of notes)

>---

>* 7.8 Equality operators. Consider comparing (null == undefined).

>The spec has this expression has false. Shon suggests true because of compatibility with existing pages. MS has run across cases of this in web pages. No resolution -- waiting on input from Brendan.

>

>* 7.10 Meaning of && and ||: do they work like PERL or like Java, C and C++? Clayton wants PERL-like. Shon wants Java/C/C++-like. Shon points out that no implementation correctly implements the PERL semantics.

>Brendan to provide examples showing the usefulness of the PERL semantics.

>

>* Order of evaluation for assignment. Randy and Shon now agree with the left-to-right logic. Brendan needs to think about this.

>   // Example 0

>   x = y = z = 1

>

>   // Example 1

>   var x = 1

```
>	var o = new Object()
>	with (o)
>		x = o.x = 2
>	print(x)  // Is x 1 or 2?  It's 2 if we do left-to-right.
```

>This is what is in the spec today.

>

```
>	// Example 2
>	o[i] = i++
```

>

>* Lifetime of Activation Record Object (has scope chain). Waiting on Brendan.

>

>* Should the arguments object include local variables? Waiting on Brendan.

>

>* Should the Activation object and the arguments object be the same? If not, then two arguments share the same members.

>

>* Need to specify the semantics of numeric literals. Agreed that this is legal:

```
>	var x =
>0x1111111111111111111111111111111111111111111111111111111111111
```

>and that if there are enough 1's, it is just equivalent to infinity.

>This isn't really specified at all today. So this is a work item now

>--

>we have a direction.

>

>---

>* 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?

>

>* Versioning. Scott contributed a high-level proposal, which we need to review.

>

>* What should division by zero result in? Shon thinks that division by zero should generate NaN rather than signed infinity, as specified in the doc today, and that we should not have the concept of positive 0 and negative 0. IEEE-64 does have these concepts. We are approximately evenly split on this issue. We agreed to think about this for one (and only one) more week. The next time we meet, we will settle this issue, even if settling it requires flipping a coin to determine the result.