Minutes of the: ***Ecma TC39, ES3.1WG***

held in: ***Phone conference***

on: ***16 December 2008***

1 Roll call and logistics

1.1 Participants

Doug Crockford (Yahoo!), Pratap Lakshman (Microsoft) and Allen Wirfs-Brock (Microsoft)

2 Agenda

Triage trac ticket 441.

3 Minutes

441 - Why do we need to standardize the object literal form when we are standardizing the reflexive form? - the use of ‘get’ and ‘getter’, and ‘set’ and ‘setter’ could be confusing - agreed that there is a consistency issue here - I would prefer if only the reflexive form in standardized, and revert the grammar changes in 11.1.5; the object literal form can continue to be supported in browsers as an extension; why extend object initializers for this? - but 3 out of 4 browsers already support it! - I am not sure about this, but at the time we came up with the ‘3 out of 4’ rule, was Opera already supporting it; I ask because I ran the test on the latest version 9.63 for Windows - that does not matter now; we should standardize what is already supported - in any case, it is good to have rich object initialize syntax - can you change ‘getter’ to ‘get’ and ‘setter’ to ‘set’ now for IE? - but we already shipped this in the present form (using ‘getter’/’setter’) in Beta2! - see if it can be changed - Ok, but there is only a slim chance, though.

439 - If we are going with ticket 441, we can as well discuss this - what should be the value of the name properties for getter/setter functions defined in object literals? - how about just “p” where the name of the property is “p” - but, the name property could be used in debuggers, and naming it “get p” would be useful there instead of just “p” - make sure that the name property could match the recursive binding - but you cannot use the string to do any binding anyway - agreed, lets go with “get p” with exactly one space between “get” and “p”.

428 - Sharing semantics of arguments (Brendan and Maciej have objections) - what does it gain strict mode to not have sharing semantics? - if the semantics were to be different in strict mode and non-strict mode, then each set of semantics needs to be described - but, what are the semantics in non-strict mode? That does not seem clear - consider 2 closures; one referencing arguments and the other referencing a named parameter … are they still shared? - what does it mean to “share a value” in ES3 anyway? - lets also get Mark on the call to discuss this further.

440 - Why is the ‘name’ property enumerable? All other predefined properties of Function (length, prototype) are non-enumerable - typically, built-in properties are non-enumerable - agreed, we should change it to be non-enumerable.

438 - We have made an addition to section 16 that implementations “must” report, at scan time, violation of strict mode restrictions whose detection does not require program execution - could this be subject to interpretation? - should we change the “must” to a “may”? - well, I have called it out clearly in the revision history at the end of the document I uploaded, but no one has got back on this yet - ok, update the ticket explicitly mentioning this particular addition.

Meeting adjourned.