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

held in: ***Phone conference***

on: ***18 December 2008***

1 Roll call and logistics

1.1 Participants

Pratap Lakshman (Microsoft), Mark Miller (Google) and Allen Wirfs-Brock (Microsoft)

2 Agenda

Not circulated ahead of time.

3 Minutes

Sharing semantics of "arguments"

Sharing semantics of arguments - if the semantics were to be different in strict mode and non-strict mode, then each set of semantics needs to be described - the implementation support is not common across all browsers - consider the example of two closures; one referencing "arguments" and the other referencing a named parameter … what should happen after the function returns? - the linkage is broken in the case of IE and FF; but is retained in the case of Safari, Opera and Chrome - interesting … IE and FF cover a majority of the browser share … should we enshrine their implementation as the de-facto standard? - but it appears that their interpretation of the specification in incorrect.

Why does strict mode not want to have sharing semantics? - this is so that the language is more sane - but then Brendan and Maciej have objections (refer ticket 428) - not clear why they are objecting - there are clear implementation costs to the change we are proposing - ok, so there are 4 options for this particular scenario:

(a) do nothing.

(b) call out the behaviour as explicitly unspecified.

(c) standardize the IE/FF behaviour.

(d) standardize the Safari/Opera/Chrome behaviour.

Mark and Allen prefer options (d) and (b) in that order, pratapL prefers (b) for ES3.1 and fix it for Harmony - but the expectation is that Harmony will build upon ES3.1 strict - this section still needs some work.

We need to more clearly separate the discussion of strict arguments and non-strict arguments - the above four choices are re non-strict arguments - for strict arguments, there is no sharing (as stated in the Kona draft and agreed at Kona); however, there still remains an important choice:

(x) Have strict arguments be a frozen genuine array, as stated in the Kona draft

(y) Have strict arguments be a frozen, non-joined array-like object, as agreed at the Kona meeting.

If we're reconsidering arguments, as we seem to be, we favour (x).

How are you adding properties to the "arguments" object ? - using [[DefineOwnProperty]] - good, don't use [[Put]] for that purpose - agreed; need to fix it in Array initialiser, though.

Scan time reporting of strict mode violations

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”? - Mark strongly favours mandatory early reporting of strict mode violations when possible - ok, we will leave it as-is for now and revisit only if there is objection.

Array.prototype.sort

What does it mean to sort an array with accessor properties - is the comparison function meant to be a consistent comparison function? - what should happen if we sort an array having NaN’s - there are requirements of reflexivity, symmetry and transitivity on the function - the algorithm performs an implementation-dependent sequence of calls to the [[Get]], [[Put]] and [[Delete]] methods on the object - but why [[Delete]], that should be removed - by removing [[Delete]] are we precluding the use of certain implementation optimizations - can imagine an implementation that gets all the values from the array, deletes them from the array, sorts them offline, and then puts them back in sorted order - lets ask on the discuss list if anyone actually uses this facility - if no one is using it, we must remove the use of [[Delete]] - ok, and make it an error to sort an array with accessor properties; step 5 in the algorithm needs to become step 4c.

Meeting adjourned.