Minutes of the: Ecma TC39, ES3.1WG

held in: Phone conference

on: 19 August 2008

1 Roll call and logistics

1.1 Participants

Doug Crockford (Yahoo!), Pratap Lakshman (Microsoft), Mark Miller (Google), Adam Peller, (IBM), Sam Ruby (IBM) and Allen Wirfs-Brock (Microsoft)

2 Agenda

Decimal

3 Minutes

noSuchMethod

this one came on the discuss list - should we formalize this ? - seems interesting - not enough time to pin down details - need to consider interactions with other features, security implications, etc. - the semantics can be realized through various means (the proposal being just one of them) - needs to considered in the context of other features in “Harmony” - will need to evaluate it from a future-friendliness perspective - will need to eventually be expressed using appropriate spec formalism - not for ES3.1; can wait until “Harmony”.

Decimal

‘===’ on decimal values should check for computational indistinguishability - ok, in that case what happens in a switch case ? switch uses ===, and if we have a case 1: and a case 1.0m:, then they are equivalent, and control shall go to whichever comes earlier textually ? - how about case use EQ ? - no, leave switch alone - should we just back out EQ then ? ‘===’ already special cases Number - computational indistinguishability is too valuable a property to loose - keep EQ for now.

is it premature to introduce Decimal ? Its tentacles seem to be extending further and further - but, EQ was generalized independent of Decimal - not Decimal’s fault - we have already invested time in Decimal; lets see if we can see it through.

No compelling reason to add Hash in ES3.1 (that maps to EQ).

Mixed mode arithmetic should always be done as binary FP - limited precision (LP) vs unlimited precision (ULP) - which of these two implementations should ES3.1 have ? - what does 1/3 evaluate to in ULP ? - if there are no slippery-slope issues we should go with ULP - whatever decimal we add should be the last decimal we add - if we add LP, there will surely be a need for ULP - what are the usecases for ULP ? - crypto, for instance; need to check if ULP will address the requirements - perhaps we need to have separate types for those (like bigInterger or bigDecimal as in the case of Java).

Meeting adjourned.