Minutes of the: ***Ecma TC39-TG1***

held in: ***Phone conference***

on: ***26th April 2006***

Attendees

- Jeff Dyer, Adobe Systems
- Ed Smith, Adobe Systems
- Brendan Eich, Mozilla Foundation
- Graydon Hoare, Mozilla Foundation
- Dave Herman, Northeastern University
- Lars Thomas Hansen, Opera Software
- Francis Cheng, Adobe Systems

Agenda

Note new meeting day of week for phone conferences.

- type system
- other hot topics

Discussion

- AS3 experimenting with Class <: Function

- Type system
  - Review for people who were not at the face-to-face last week
  - Dave: type T = (A, B, C) – what if type A = function(int):int, B = function(String):String?
  - Lars: unions flatten, so they are less expressive than switch class already
  - Dave: still, unions can't be pulled apart in all cases
  - Graydon: switch type (t:T) { case (a:A) {...} case (b:B) {...} }
  - Graydon: memoizing helps
  - Dave: runtime-only discrimination is very strange:
    - function(Object):specialInt <: A, so that matches A not B in T
    - so which match wins? closest, first in some order, ...?
  - Agreement this needs closer inspection, but it has advantages for structural typing
  - Dave mentions OCaml tagging all arms of its sums
  - Graydon: we're doing that but tags come "for free" (modulo memoization)
  - Branding vs. nominal and structural rules
  - Lars: does Class <: Function make for trouble if we use branded structural types for nominal types?
  - Jeff: could model types as nominal, naming structural types by their shape
  - Dave: soundness at risk if static type and dynamic type system boundaries are blurred
    - Brendan/Graydon: not a risk here, runtime reflection not required by branded structural proposal
    - brandnames can be statically checked

  - More thought required
  - Lars: current union type proposal may be ill-conceived

- Lars updated operators proposal, looks good

- Lars made a Unicode counterproposal, seems inevitable
  - Brendan floated not stripping ZWJ/ZWNJ from strings and regexps

- Let block proposal – remove the FunctionBody special body

- Nullability: default value, definite analysis in constructor bodies, and all that
  - Ed: try-catch-finally makes for hardship beyond constructor case
  - Default values help this
  - Dave: recursive types? Lars: no recursive non-nullable types!
    - (It's actually possible to have recursive non-nullable types provided that it is possible to create a self-referencing node to act as a sentinel – a fixed point.)
  - Ed points out non-nullable use-case for fields is as strong as for args
    - so non-nullability is well-motivated, we believe
    - do this by hand in other langs: init non-null in ctor, don't check in methods over life of object
    - can we do better using the type system?
  - Dave: does super + default values make a loophole? Seems so
  - Seems UninitializedError is **not** a cop-out that's as bad as NPE
    - it comes early
    - it points to faulting party
    - unlike a null that flows off into space and is dereferenced later
  - Ed: how about non-nullability helps static analysis, but null can flow through
    - and runtime checking still required – just for bogus constructor cases?
    - Graydon, others: see the practical benefit, but would prefer the type system handled this
  - Ed: what if we made non-nullable fields have to be specified as arguments, to require init?
    - Dave: pure functional style
    - Brendan: people do this (Crockford) – environments as objects
    - Can we make non-nullable fields require special form init in constructor?
  - Ed: minor correction
    - default value for Number is NaN, not 0
    - int default value is 0
  - Ed confirmed that we meant to eliminate `dynamic` as field qualifier (in builtin classes recent changes)
    - We did (yay!)