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

held in: ***Phone conference***

on: ***24th March 2006***

Attendees

- Francis Cheng, Adobe Systems
- Jeff Dyer, Adobe Systems
- Brendan Eich, Mozilla Foundation
- Gary Grossman, Adobe Systems
- Lars Hansen, Opera Software
- Dave Herman, Northeastern University
- Graydon Hoare, Mozilla Foundation
- Dan Smith, Adobe Systems
- Edwin Smith, Adobe Systems

Agenda

- Type parameters
- Switch class
- Block expressions
- Make Wiki public?
- Multiple compilation units
- Bug fixes
- Dave's progress on spec language
- Member lookup clarification

Notes

Type Parameters

- Reviewing proposal type_parameters
- Committed feature, but syntax details remain. We have some syntactic proposals, but we don't have anything we agree on.
- Brendan: Already doing this for containers and iterators, and if we do that we should do it generally, but not in the complicated way that Java did it, rather only with invariant type params.
- Graydon: This is not type inference or overloading, so we're not doing anything terribly significant.
- Brendan: Dave's main concern was lack of lexical scope, or presence of dynamic scope when you use old features. Lars reminded us of what other languages do where programmers have to declare when they are using dynamic features.
- Dave's type soundness issues: whether they are statically known (the set of classes). You can only create dynamic classes with eval(). Can you have a class decl inside a function body?
- Jeff: Not in AS3.
- Dave: The main concern is the lack of definition. we just haven't defined it. We just don't know what the type system is.
- Jeff: So how do we make progress? We need to figure out how to solve this.

- Brendan: How do we start small and grow?
- Dave: it would be useful for Cormac and me to flush out a more formal definition and run it by the AS3 folks to see if this is what they had in mind. We should be painfully precise.
- Brendan: Brought up issue of Initial value for non-nullable types. So for Number (0), Boolean(false), etc.
- Dave: There is some precedent, so I need to do some research. This isn't the biggest concern. I'm more concerned about the class structure, etc.
- Brendan: Lars brought up modula-3 and C# where you declare an "unsafe" region – we could call it "dynamic".

Switch class

- Reviewing proposal switch_class
- Graydon is less enthusiastic about the switch class proposal, and will answer Lars's questions on the wiki.

Block expressions

- Reviewing proposal block_expressions
- Lars has convinced Dave of the right middle ground.
- Dave: the idea of let has issues (esp. if proper tail recursion happens). Lars has a counter-proposal. Dave and Brendan like it.
- Jeff has concerns because "let" sounds like it comes from functional languages (like ML). Just seems weird that there's 2 ways to declare a var.
- Brendan: I hear you, but we need something new to maintain compatibility. Building on var may not clarify. If we're going to do something new, prefer to do something that looks different. We could use "local", don't like "block". But what does "local" mean?
- Jeff: This kind of deprecates var.
- Brendan: Yes it does, but having var have local scope with hoisting was a mistake. I had originally been hoping to simplify the scope system before I was pulled away to work on the browser.
- Dave: another point is that no one likes adding new features to a language, but we don't have that luxury. Mature languages often must live with feature bloat.
- Jeff: well if I look at it as a replacement for var, I feel a little better. Do let def's shadow?
- Dave: they should shadow.
- Jeff: like Java?
- Lars: yeah, in Java it's an error.
- Dave: it works better for Macros if you shadow, too.
- Action Item: Lars will clean up the proposal.

Make Wiki public

- Brendan: hoping to get to a point soon where we can make the wiki open to the public.
- Jeff: probably when things get more stable.
- Brendan: we'll probably need sign-off from Ecma as well.

Multiple Compilation Units

- Reviewed Lars's proposal multiple_compilation_units
- Lars has started this and would like to get some feedback from Adobe folks. Need some terminology clarification. Problem is that the 3rd edition grammar uses the term "program".

Bug fixes

- Reviewed proposal bug_fixes

- Main topic of discussion was Function.length (a.k.a. Function.arity)
- Ed will consolidate all spec bugs about function.length. Should tell you number of formal parameters. The number of declared (req and opt) parameters.
- Brendan: So length should be minimum expected. Is it useful?
- Dave: if it consistently tells you the min. number, at least you can detect possible error conditions.
- Ed: part of the problem is that Ed3 doesn't throw if you pass in fewer than the min. args.
- Brendan and Dave: favor meaning equal to min args. What about backward compat. issue of changing length?
- Lars: not sure it's used that much, so probably not a big issue.
- Ed: so consensus is min? That would mean it's always 0 because Ed3 doesn't throw for arg count mismatch. One way out is to make Ed3 behave such that it assigns default values for required args. Ed3 code calling function with req args, so Ed3 will pass default values. In Ed4, it will be an error.
- Dave: So we will be able to dist. between ed3 and ed4 code?
- Ed and Brendan: we'll have to.
- Dave: I like it, but we now have combinatorial explosion of lang, old Ed3, ed4 typed, ed4 untyped.
- Brendan: maybe this is more a distinction between unchecked and checked.
- Dave: so you have a typed function with args, you have untyped code that calls it. If its ed4, error, but ed3 code wouldn't. So that's dist. between ed3 and ed4.
- Brendan: you're right. If you're talking about old callers supplying missing arguments, that's diff from checked versus unchecked.
- Ed: In AS3 once you get into a class, everything is checked.
- Brendan: someone should check the clarification issue
- Graydon: that's the code mixing topic(code_mixing).
- Brendan: I agree with Ed that if we can redefine arity to mean formal required parameters.
- Ed: Ex: Number.toString.length will be 1 because radix is optional.
- Action Item: Ed will try to enumerate all cases where the length value will change.
- Ed: Another issue to consider is that all core objects are instances of class Class instead of Function. Function has a length parameter, which implies that Class should have length parameter that gives number of constructor args. In AS3, the Class type has an instance var length.
- Dave: what if you have a static var named length.
- Ed: Exactly, that's the problem.
- Dave: if you want to share static in same object as class constructor, either say it is non-overridable or it has the same kind, if you're inheriting through prototype you would have shadowing, or you can say you can't override something that's there.
- Brendan: what about namespaces? Could we use a built-in const namespace, or would that not resolve ambiguity? I did this in hashcode, which will be a new global function. Someone somewhere probably has a function named hashcode. We don't want to make it non-overridable and break that page. If we use a private namespace, we might be able to avoid that.
- Ed: that would be solved if we put global identifiers in front of imported ones. So if they're on the same object, namespaces get you close to a solution.
- Dave: what does it mean if you have a special namespace on an object, and you try to get to that property using a string index?
- Ed: It's an implicit prefix plus an explicit runtime string.
- Dave: namespaces are intended to be static. Anything on the left of :: should be a compile-time construct.
- Ed: there's a pair, lhs side can be compile-time or runtime computed namespace.
- Jeff: E4X has this problem because qualifiers are always resolved at runtime. So "use namespace" requires compile-time.
- Dave: So in some cases, it can be lexical binding, it's for this particular case of E4X that we need to dynamically compute. You can always statically detect whether someone is using dynamic binding. The compiler can always know what's static vs. dynamic.
- Ed: is there a way to introduce new names without conflict? Have an ed4 namespace, put that on all new functions. Define that implicitly and it doesn't solve the ambiguity problem, but it gets really close. We want to use this for early binding. You always see the fixed one before the dynamic one. There's a look up order, always look at fixed before dynamic.
- Brendan: the other thing is to make the built-ins, when named properly, early bound.

Dave's progress on spec language

Has prototype semantics, a lamda calculus, using Stratego. It lets you spec your own syntax, so I can have free reign over how it looks, but it is still executable. Feel excited about this. I was starting to think about pretty notations with nice fonts and pictographs, but Graydon said, “look, we’re programmers, put it in something we can understand.” The way forward is to make the semantics that, at its core, takes from operational semantics but has the feel of a programming language. Should have a demo soon. Hoping we can use this to prototype or notate the ideas were talking about. Dave’s hoping to have something next week.

Member lookup clarification

- Brendan brought up early binding and disambiguation without naming conflicts for built-in versus user identifiers. Should we write something up about this?
- Ed: it would be better if we could intro new namespace. If we just use shadowing, there’s no way to get to identifiers that have been shadowed. What if we combine shadowing and namespaces? Jeff, if we try to add ordering, would this somehow screw up the versioning scenario?
- Brendan: did this come up with one of Jeff’s proposals?
- Jeff: sort of, mine was shadowing with namespaces: namespace_shadowing
- Ed: here’s a problem. Base class has fixed property, subclass has dynamic prop of same name. which one do you get?
- Action Item: JD and Ed will work on separate member lookup clarification topic. We’ll morph Jeff’s namespace shadowing proposal.