Minutes of the: ***3rd meeting of Ecma TC39***

held in: ***Newton, MA, US***

on: ***15 - 18 April 2008***

Chairman: Mr. John Neumann (Adobe/Microsoft)

Vice-Chairman: Vacancy

Secretary: Dr. Istvan Sebestyen (Ecma International)

Attending*: Mr. David Boloker (IBM), Mr. Jeff Dyer (Adobe), Mr. Brendan Eich (Mozilla), Mr. Lars Hansen (Adobe), Mr. David Herman (Northeastern University), Mr. Graydon Hoare (Mozilla), Mr. Adam Peller (IBM), Mr. Chris Pine (Opera), Mr. John Resig (Mozilla), Mr. Mike Shaver (Mozilla), Mr. Scott Unterberg (Mozilla), Mr. Allen Wirfs-Brock (Microsoft), Mr. Jon Zeppieri (Boston University).

*** at Thursday and Friday meetings**

NOTE
Attendees for the first two days included representatives from Boston University, UC Irvine, Northeastern University, Adobe, and Mozilla.

By phone: Mr. Mike Cowlishaw (IBM), Mr. Waldemar Horwat (Google).

1 Opening, welcome and roll call

1.1 Introductions

The meeting was opened by **Mr. Neumann** and all delegates were welcomed to the meeting.

1.2 Host facilities

1.3 Dinner arrangements

To be announced. Hosted by Ecma-International.

1.4 Lunch

To be provided

2 Pre-meeting Technical Discussions (15 – 16)

A presentation on Trace Trees was given by **Andreas Gal**, **Michael Franz**, and **Michael Bebenita** of University of California – Irvine. Focus was on reducing trace points by evaluating code at merge points and guard points to see earliest point a tree branch can be merged, reducing the subsequent number of leaves in the tree. A second part of presentation examined bailout reduction and optimization.

There was later a discussion of plan for the week and the need to make progress on writing final text for the draft. It was agreed to work towards this objective. Meeting adjourned for the day and will pick up again on Wednesday.

The second day of meetings was a number of small groups discussing a set of different subjects and advancing the state of the text of the first draft.

3 Adoption of the agenda

Approved as modified to include subjects from WIKI agenda.

4 Approval of the minutes of March 27th & 28th, 2008

The minutes are not yet available for review and approval. They will be approved at next meeting of TC39.

5 ES 4 technical discussion

5.1 Review of draft specifications

5.2 Discussion of open design issues

- Decimal

  Proposal presented by IBM and Microsoft, followed by significant discussion. It has been recommended that this functionality be implemented to prove/disprove its viability before adding to the standard. There appears to be solution that allows ES 4 to support Decimal in a way that might allow ES 3.1 to support Decimal through the use of optional precision for rounding. A write-up will be produced and floated to see if there is support to continue. There appears to be general support with Opera, Google, IBM and Microsoft. ES 3.1 will be able to provide this support through its library and a superset of ES 4 functionality (operators).

  Lars will send out a proposal for ES 4 that can be adopted for use in ES 3.1

- Names

  Need to identify Namespace qualifiers. Problem is that there have been no responses to original email description, and Lars believes that without email responses it will be difficult to discuss. Problem is to make sure that compile-time evaluation is the same as run-time evaluation. Example follows:

```
namespace A
namespace B
use namespace A, namespace B
A namespace ns

... ns::x ...
```

- What is the meaning of `ns`?
- So far, it is `A::ns`
- But what if another namespace ns is introduced

```
B namespace ns
```

- now `ns::x` above is ambiguous
- we want to make sure this is caught early

- what is proposed is that potential namespace (and type) names are reserved in subsequently loaded programs
- this ensures the reference to `ns::x` doesn't change meaning
- units might give programmers more control over the checking of collisions

```
var v = ns
```

Further discussion will occur online. If there is a proposal to change the way it works now it has to be written up. Specification of what exists now in the RI needs to be done and will be done in the forthcoming Specification.

- Packages

Statement of proposal to eliminate Packages follows:

At the last f2f Waldemar raised pertinent questions about whether the desugaring of packages (in their current form) into namespaces would work.

The answer appears to be, for the time being, no. One important problem, for example, is that packages appear to be scoped entities but are not. This fragment of code:

```
package P {
  public var x
}
```

means exactly the same thing as this fragment:

```
package Q {
}
public var x
```

and that leads you into trouble when you start using import. Consider a third package R:

```
package R {
  external var x
}
```

Now import R into P and Q and reference x:

```
package P {
  import R.*;
  public var x;
  ... x ...
}

package Q {
  import R.*;
  ... x ...
}
public var x
```

Now the _intent_ of the reference in Q is clearly meant to be to R.x, and the (outer) public::x should not be allowed to change that intent (consider that public::x is introduced in some later file, at a later time) while the most reasonable behavior for P is that x references public::x (it's in the same block as the reference, therefore "closer" or more obvious -- take your pick). The desired behavior is not the same in the two cases, yet the two fragments mean the same thing.

It's possible to construct even more fun examples with lexically nested import statements.

It may be possible to fix this by introducing arcane restrictions (consider flagging the reference to x in P as ambiguous) but not, probably, as a pure desugaring to namespaces, and even then it's not clear how useful the package abstraction would be.

As a consequence Jeff and I think we should remove packages from ES4 for the time being, instead of embarking on a design exercise without the benefit of practice. Concretely, we propose the following:

- the 'package', 'import', and 'external' keywords and forms are removed from the language along with the dotted-name syntax for packages

- the notion of top-level blocks (which was only required to model the desugaring) is removed from the language

- "use default namespace" is allowed anywhere at the top level of the program and at the top level of class blocks (and its effect extends to the end of the program or class or until a new "use default namespace" pragma is seen)

- "use namespace" is allowed anywhere at the top level of any block (including program and class blocks) and its effect extends to the end of the program or block

- the keyword 'internal' persists, but its value is now an unforgeable namespace that is private to the compilation unit (script, file, etc).

The net effect is simplification and actually better integrity, since the file-private unforgeable namespace allows programs truly to hide definitions in a file. (Our "local package" mechanism provided something similar but that proposal appears to have fallen by the wayside.)

The mechanisms proposed above are sufficient to emulate simple packages a la Java. Consider an evaluator package that has a statement evaluator in one file and an expression evaluator in another:

```
// file 1
internal namespace PRIVATE = "eval private"
internal namespace PUBLIC = "eval public"
```

```
use namespace PRIVATE, namespace PUBLIC

PUBLIC function evaluate(t)
  evalStmt(t)

PRIVATE function evalStmt(t)
  ... evalExpr(t) ...

// file 2

internal namespace PRIVATE = "eval private"
internal namespace PUBLIC = "eval public"
use namespace PRIVATE, namespace PUBLIC

PRIVATE function evalExpr(t)
  ... evalStmt(t) ...

// file 3 (client)

namespace EVAL = "eval public"
use namespace EVAL              // import EVAL.*

print(evaluate(read()))
```

Perhaps, with some experience, a lightweight syntax could be added to ES4 to make that kind of evaluation simpler, but that's not what we're proposing at this time.

End proposal.

Feedback on the web has been positive to this proposal. There is a small group of programmers on the web who like Packages so there may be some backlash from that group.

There were no objections to removing Packages from RI and Specification.

- other

- Review of features specs

Comment was made that most have been well written, but of late they have been poorly written and semi-complete. It has been the intent to break things down into easily reviewed pieces. Some features proposed for ES 3.1 need to be proposed for ES 4 since 3.1 is a subset of 4.

- Review of spec schedule

Basic schedule would call for mid-May availability of first draft document. Final review draft in July with posting to Ecma Website, and sent to SC 22 for their information and review. Final draft will be reviewed in late September. ES 3.1 is trying for the same schedule, but meeting this schedule is a function of having sufficient resources to do the job. There are some issues related to presentation within the document that are still under discussion.

Wiki page:

http://wiki.ecmascript.org/doku.php?id=meetings:es4wg_apr_17_2008

Group broke up into smaller groups to discuss specific issues. There was an ES 3.1 group in addition to the two or three ES 4 groups.

6 ES 3.1 liaison/technical issues

ES 3.1 was tasked with reviewing their work and ensures that all features being planned are also a part of the ES 4 work. This loop will be closed in the next couple of weeks.

7 Any other business

None

8 Date and place of the next meeting(s)

May 29-30 in San Francisco, CA (Full TC).

9 Closure

Meeting closed around 3:30 PM on Friday.