ECMAScript meeting 21st June 1999

Present:

Mike C
Dave
Chris
Sam
Waldemar
Andrew
Herman
Dario
Jeff
Richard
Bill
Mike M

Meeting Notes

5.2 Algorithm Conventions

Herman is happy with the new wording.

5.14, etc. Error Object and exceptions

Herman asked about callable. Microsoft has shipped it but should it be in the standard?

Action: put back Error as a function (in 15.11)

Do we want to add Error.prototype.toLocaleString()? The key phrase is "implementation defined". We also want to set up a framework that users can extend.

Usually in the exception class you have errors defined by the programmer. You then need to translate the error into the correct localized message.

We have to have toLocaleString(), but it is just a question of what it does. We seem to want to punt on having the error class include localised strings. This would be ok since it is only for developers.

What about user defined exceptions? Can we add a hook, either now or later?

Resolution: we drop toLocaleString() for the Error object.

Is there a message property for Error object? Proposal we support getMessage() that return's the message property? We decide to leave the way it is now.

Returns an implementation defined result based upon some combination of name and message properties

Herman asks is it necessary to leave toString quite so vague? Yes, it allows us to add line numbers etc. Leaving it as an unspecified function offers us extra flexibility.

Perhaps the text is ***too*** specific? In its current form, it would preclude adding line numbers, asserts Mike. We decide to go with:

Returns an implementation defined string

Moving on - 15.11.6.* miscellaneous errors e.g. SyntaxError, AssignmentError, ConversionError and ArrayBoundsError. Waldemar adds RegularExpressionError

We look at the four references listed for AssignmentError. Waldemar proposes we revert the name back to ReferenceError. Mike asks should it be a separate error?

Resolution: we change the name back to ***ReferenceError*** until an alternative proposal is put forward.

We decide to make it content-agreed after we apply the agreed changes.

We now move to 5.2.

In 7.8.1 second list second item. Herman doesn't like "the program is syntactically an error". We agree to change it to "the program is syntactically incorrect". After that we move it to content-agreed.

8.6.2 Internal Properties and Methods. Change to first paragraph needs some work. Bill takes ownership.

Various changes needed to match today's decision to revert to "ReferenceError".

9.1 and 9.9 text changed to refer to ConversionError. In section 11.2.2 various changes to refer to TypeError. In section 11.2.3 and 11.8.7 ditto. Moved to content-agreed.

15.1.2.1 and nearby SyntaxError and EvalError. No objections. in 15.4.2.2 in the array constructor we renamed the error to ArrayBoundsError. Some discussion that this is perhaps confusing. You would expect an error by that name to be generated when you try to access of the end of the array.

Action: we change it to ArrayLengthError (15.11.6.4 and 15.4.2.2 and 15.4.5.1).

In summary lots of minor error renaming. The principle seems to be if in doubt, throw "TypeError". For the most part we move the changes to content-agreed. After Bill has done his bit and the other minor changes done, all will be content-agreed.

Question should ArrayLengthError be a subtype of TypeError? Waldemar thinks no.

I18N

Waldemar talks through his print-out which shows the effects of toUpper and toLower on Unicode. The lengthy tables show where round-tripping won't be possible. We then had an entertaining discussion of case conversion for Turkish and Ancient Greek.

15.9.5.? Date and Time Formatting

4 new functions. Richard has broken out the ability to get the date and time independently and in a locale dependent and independent version. Mike objects to the text "something along the lines of" to "might be". This is really just a bare minimum.

Resolution: Content-agreed

Locale sensitive case conversion

We plan to add parameters in future and hence are including some future proofing to allow this.

Locale sensitive string comparison

The idea of a strength parameter that determines the significance of various kinds of differences, e.g. case differences, hyphenation, diacritic differences. The collation strength is highly language specific, but we choose to define sort order to be locale rather than language based.

We discuss whether we really need this in the language. Richard says no one has fully implemented the Unicode collation algorithm, although Java comes close. Richard's intent here is to support comparison of strings, not words or sentences. If you want to do the latter, its up to you to break the string up first.

Perhaps we could add locale and strength in version 3? The strength is important for searching but not for sorting. The number one application is sorting lists for display in a list widget. For this we can do without the strength. Bill proposes that when we add strength we do so for a new function.

Resolution: we drop strength but reserve for future the possibility of adding the locale parameter. We also allow for returning the strength.

Regular Expressions: 5.5.4.10 split and 15.1 Global Object

We run through the changes Waldemar has made to the grammar.

7.6 is missing === from the list of punctuators. The paragraph on ambiguity has been dropped in favor of adding the lookahead constraints. Section 12.4 prohibits expression statements from starting with a left curly brace, which could otherwise occur in Object literals.

Jeff will adopt Waldemar's text for 15.1.2.1 items 4, 5 and 6.

Richard asks about the motivation for the empty separator string. Answer: its modelled after perl. Waldemar draws an example on the whiteboard to explain some of the tricky details for empty strings. Waldemar explains that there is no-backtracking in the algorithm. Herman asks for some more time to check the details.

Mike rules that the earlier sections are content agreed, but 15.5.4.10 is function agreed, and Herman and Dario will proof read it and report back.

Bill has some comments on improving the clarity of introductory paragraphs. Mike asks about suggests adding a reference to Perl 5. Bill will deal with this.

Waldemar explains some subtle differences from Perl, e.g. with ?= (logical and for regular expressions) where Perl doesn't restore the match position. Microsoft uses an external module for regular expressions which has to satisfy the perl compatibility tests.

One of our guiding principles was compatibility with Perl, even to the point of duplicating perl's bugs. There is no standard for perl, but this "bug" is one of those in the compatibility tests. Another yardstick is the O'Reilly book on regular expressions as we can expect users to spot any departures.

Unless we can find any evidence that perl will change, we should stick with perl compatibility.

Action: Andrew to talk with Dick Hardt about direction perl is evolving so that any decision we take isn't wrong footed by work on perl.

Action: Dario to chase down discrepancy between perl implementations with regards to section 15.10.2.5

15.10.2.8 content agreed except for the perl bug issue.

Mike resolves that the document can be merged into the base docoument (i.e. passed to Bill) with pending info from Andrew and Dario.

15.1.2.8 encodeURI() and 15.1.2.9 decodeURI()

Dario adapted the definitions of escape and unescape. Does the specification as written make sense? Bill agrees to make editorial changes to put the grammar into the same presentation as elsewhere in the spec, and to fix any name collisions at the same time.

In 1.2.1 step 14 the result should be 12 not 18. encodeURI() doesn't throw any exceptions. decodeURI() can throw an InvalidEscapedURI exception. Mike points out that the name isn't consistent with other errors. We agree to "URIError".

Some discussion about large Unicode values. The algorithm as currently defined doesn't deal with surrogate pairs.

Action: Chris to provide improved text for step 13

Resolution: function agreed, Dario to pass spec to Bill for inclusion in the

base document.

11.1, 11.1.5, 11.1.6 Array and Object initializers

Proposal: making style change - drop array literal head production, and move left bracket down. Left recursive allowing a further cleanup.

Drop "array literal" since its not really a literal. After that, pass text to Bill for incorporation into the base draft. Needs a high level reference - action on Bill.

15.4.4.10 Array.prototype.sort

Waldemar circulated his revised text. Some mirth over the definition. Bill suggests we call it implementation defined behavior rather than saying "the result is implementation defined". With this change it is resolved that this is content agreed.

7.1 Unicode Format control characters

Richard says he doesn't care about this. Action: Richard to look into what this was about and to mail Mike C.

Discussion topics

We ended the meeting at 4pm and hence postponed the discussion on numeric formatting and changes to 1. Scope.

Next Meeting

Netscape Building ??, Mountain View, July 12-13th 1999. 9:30 start for both days. The agenda will include locale sensitive string comparision. String.split. One or two small items. A question on perl5. Final text for encodeURI and decodeURI. Walk-through of unaltered sections.

Bill requests people to send him their new text by the end of this week, so that he has a chance of getting the full draft ready for people to download a week before we meet again. Bill prefers to accept material in Word format.

A 2 hour I18N meeting is planned for the morning of 22nd June.