ECMA/TC39/98/7

ECMA

Standardizing Information and Communication Systems

Letter ballot results for DIS 16262

The next steps

Attached you find the ballot results for DIS 16262. The DIS has passed successfully. Congratulations ! 13 P-members and several other countries have voted in favour, there are 6 abstentions, and 3 NO votes. At least one NO vote can be easily accommodated and changed.

Comments have been received from the following countries:

– Denmark
– France
– Japan
– Netherlands
– USA
– ECMA: see document TC39/98/5.

The comments have been scanned (apologies for the poor quality).

The comments have to be resolved in the ballot resolution meeting which will be held back-to-back with the TC39 meeting in the week of 25th June 1998: see also SC22 N 2707 from Bob Mathis. Two deliverables have to be prepared:

1) A disposition of comments report, listing how each of the comments has been resolved.

2) The so-called ‘Final DIS text’: this is the ‘camera-ready’ copy of ISO/IEC 16262, the international standard.

The ECMA Secretariat will work closely together with the editor to have these deliverables as soon as possible. It may be useful, in order to keep the ECMA Standard fully aligned with ISO/IEC 16262, to publish a second edition of ECMA-262 (not to be confused with what is currently called now and then the 2nd edition - under development - of ECMA-262).

As a suggestion, it may be useful to identify all comments received (e.g. DK1, DK3, DK3, etc., NL1, NL2, etc., US1, US2, etc.) and merge these, in the same sequence as the paragraphs in the DIS, into an intermediate document to which all dispositions can be added, resulting in the ‘Disposition of Comments’ report.

TABLE OF REPLIES / 1998-04-21 1 TABLEAU DES REPONSES

JTC l/SC 22
ISO/IEC DIS 16262

VOTING BEGAN ON/DEBUT DU VOTE:1997-10-09
TIME LIMIT FOR REPLY/DELAI:1998-04-09

TITLE: ECMAScript: A general purpose. cross-platform programming language

TITRE: ECMAscript: un langage de programmation "cross-platform" à usage généra1

MEMBER BODY/COMITE MEMBRE		APPROVAL/APPROBATION	DISAPPROVAL/DESAPPROBATION	ABSTENTIOh
Australia (SAA)	P			X
Austria ION)	P	X		
Belgiun: (IBN)	P	X		
Brazil (ABNT)	P	X		
Canada (SCC)	P			X
China (CSBTS)	P	X		
Czech Republic (CSNI)	P	X		
Denmark (DS)	P		x	
Egypt (EOS)	P			
Finland (SFS)	P	X		
France (AFNOR)	P		X	
Germany (DIN)	P			X
H ungary (MSZT)	o	x		
Ireland (NSAI)		x		
Italy (UNI)	0			X

MEMBER BODY/COMITE MEMBRE		APPROVAL/APPROBATIO N	DISAPPROVAL/DBSAPPROBATION	ABSTENTION
** Japan (JISC)	P		x	
Kenya (KEBS)***		X		
Korea, Republic of (KNITQ)	o	x		
Netherlands (NNI)	P	X		
New Zealand (SNZ)	O	X		
Norway (NSF)	P	X		
Portugal (IPQ)	0			X
Romania (IRS)	P	X		
Russian Federation (GOS T R)	P			
Slovenia (SMIS)	P	X		
Swede n (SIS)	0			X
Switzerland (SNV)	N	X		
Ukraine (DSTU)	P	X		
United Kingdom (BSI)	P	X		
USA (ANSI)	s	X		
T O T A L		18	3	6

* = Comments/Commentaires
** = P-member having abstained and therefore not counted in the vote / Membre (P) s'abstenant de voter; n'est donc pas compté dans le vote
***= member body suspended and therefore not counted in the vote / Comité membre suspendu: n'est donc pas compté dans le vote

Organisations sending comments: ECMA

?-MEMBERS VOTING :	13 IN FAVOUR OUT OF	16 = 81.25%	REQUIREMENT >= 66.66%
MEMBRE (P) VOTANT :	EN FAVEUR SUR		CRITERE

MEMBER BODIES VOTING:	3 NEGATIVE VOTES OUT OF	21 = 14.29%	REQUIREMENT <= 25%
COMITE SMEMBRE VOTANT:	VOTES NEGATIFS SUR		CRITERE

THIS DRAFT IS THEREFORE UNDER BALLOT
in accordance with the ISO/IEC Directives. Part 1. sub-clause 2.6.3.

CE PROJET EST DONC EN COURS DE VOTE
selon les Directives ISO/CEI. Partie I. paragraphe 2.6.3

DS
DANSK STANDARD

Danish vote on DIS 16262 ballot, ECMASCRIPT

The Danish vote is “no” with the following comments.

1. The standard **needs** to be aligned with IS0 and IEC standards in the area, These include:

on page 1, clause 3. references:

ANSI X3.159 programming language C, should read ISO/IEC 9899:1996 including AM1 and TCOR1.

ANSI/IEEE 1754 should maybe be ‘754”.

Unicode consortium unicode standard 2.0 should be replaced by :
ISO/IEC 10646-1 :I998 including TCOR 1 and AM 1-9 plus
ISO/IEC DIS 1451 International sorting order, and
ISO/IEC DIS 14552 Speoiflcatione for cultural conventions
Then there is no need to refer the non-de-jure Unicode specification.

The java specification is not used normatively, and can be moved to a bibliography section.

The specific statements needed of RFC1738 can be incorporated directly in the standard, it is about encoding in characters of control characters.

We could not ascertain the nonnatlve usefulnes of the Ungar and Smith reference, it can most likely be moved to a bibliography section.

Then the normatlve references is only de jure standards

2. The followlng references should be added:

ISO/IEC 646 (instead of ASCII)
ISO/IEC 6429 - for control characters.
ISO/IEC DIS 15897 -for reference to locales/fdcc-sets

3. All references to “unicode” string og characters should be changed to “UCS” strings or characters.

This relates at least to clauses 4,3.16, 4.3.175.1.4 6 7 7.7.4
8.4 11.8.5 11.9.3 15.1.2.4 15.5.3.2 15.5.4.5
It is necessary to specify that this means UCS-4, or possibly UTF-8

4. clause 6: change ASCII to ISO/IEC 6464 IRV. Four hexadecimal digits
are too little to represent UCS characters of ISO/IEC 10646-2
(planes outside BMP).

5. Clause 7: strictly speaking control characters are defined in
ISO/IEC 6429.

6. 7.5: DOLLAR SIGN should not be in the identifier list, according
to recommendations in TR 10176. 7.5 should refer to the "i18n"
specification of ISO/IEC 14652 for definitions of letters and
digits.

7. in 7.7.4 UnicodeEscapeSequence should be renamed UcsEscapeSequence.
Care should be taken **that all** UCS **characters (31-bit)** can be handled.
eg in UcsEscapeSequencee. HexEscapeSequence. and OctalEscapeSequence.

8. clause 9.8.1 the Gay 1990 algorithm needs to be spelled out completely,
for portability.

9. clause 11.9.3 should refer ti ISO/IEC 14651 for the complex
sorting. We propose that ECMAScript does include a more
complex string comparison conforming to ISO/IEC 14651.

10. clause 15.1.2.4: RFC1738 should be spelled out, it is not
very complicated and thus a non-de jure reference can be
removed. "ASCII" should be replaced by ISO/IEC 646 IRV.
escape() and unescape() should be applicable to 31-bit
UCS values.

11. clause 15.5.3.2: UCS characters are 31 bit, not 16 bit,
The right function to call is ToUint32().

12, clause 15.4.4.5. String.prototype.charCodeAt() shall
return a integer less than 2"31.

13. clause 15.5.4.11 and 15.5.4.12 Needs to refer to
ISO/IEC 14652 specifications in **the** "i18n" fdcc-set
for upper and **lowercase** equivalences. instead of Unicode values.

14. clause 15.9.1.4: month numbers hould be numbered from 1 to 12.
This is analogeous to date number in 15.9.1.5 and conforming

to ISO 8601.

16. clause 15.9.1.6: week day numbers should be numbered from 1 to 7 as argued in comment 14.

16. clause 15.9.1.8: refer to ISO/IEC 14652 for a possible reference to information on daylight savings.

17. clause 15.9.1.12 and 15.9.2: please note that year is currently a four-diglt Integer.

18. clause 15.9.3.1-7: the result rules for year<99 makes It hard to talk about things at the time of the birth of Jesus Christ - it should be removed. The easines it gives for dates in this century are not so useful just a couple of years from now. This is not a foolproof rule.

19. clause 15.9.5.39: refer to ISO/IEC 14652 for specifications of date formatting. and ISO/lEC 15897 for references to different locales.

Solving the above comments satisfactory will revert the Danish ‘no” vote to a “‘yes” vote

AFNOR 1998 -04- 0 9

TITLE: **AFNOR Ballot Comments on ISO/IEC DIS 16262 - ECMAScript**

SOURCE: AFNOR

AFNOR votes NO to ISO/IEC **DIS 16262 due** to **a major editorial comment on the** French Title
AFNOR will reverse its vote if its comment is **adopted.**

ITEM **1**
Qualifier: major editorial
Reference document title
Rationale:
French title inaccuracy
Proposed change
Change the French title for the following :
ECMAScript : un language de programmation multiplate-forme à usage général

Japan's **Comments on** ISO/IEC DIS 16262

Title: Information technology --ECMAScript : A general-purpose, cross-platform programming language

The National Body **of** Japan disapproves ISO/IEC DIS 16262 for the reasons below. If the comments are satisfactorily resolved, it will change its vote to approval.

Major technical comments

1. Conformance

The conformance section should **be rewritten** in order to clarify what are included in the requirements to conform to the standard and what are excluded. The following items 1.1 trough **1.3 are** problems.

1.1 Implementation limits and implementation defined matters

The ECMA-262 does not specify implementation limits nor implementation defined items, therefore it is considered that conforming implementation of the ECMAScript must meet with everything described in the ECMA-262.

Besides, the clause 7.5 says "An identifier is a character sequence of unlimited length" without apecifying any implementation limits of number of characters and numbers of identifiers. It implies that every **conforming** implementation must support identifiers that consists of millions of characters and accept millions of identifiers in a program. Japanese National Body (JNB) believe that the requirement would be too tough for any implementation.

Therefore, JNB would suggest that the ECMA-262 should have "implementation limits" clause and specifies minimum requirements for program portability in the clause, then provide a clause," implementation defined matter" and list the items that a conforming implementation can define.

For example, specify minimum requirements of length of an identifier as 256 in the implementation limits clause and specify the ***number*** of character allowed for identifiers as implementation defined matter, so that such an implementation becomes conformance to thia standard that takes first 1024 characters of the identifier as meaningful and ignore the remaining characters.

The length of identifiers, the number **of** identifiers in a program, and the length of a line should be implementation defined.

The clause 7.2 introduces the concept Line terminators. But the means of line termination is file system dependent, e.g. FIXED type dataset of IBM System 390 does not have any line terminator character. So, the means of line termination should also be implementation defined, as far as the scope of this standard is general purpose.

1.2 Direct reference to documents outside of ISO/IEC standard

The ECMA-262 directly refers to technical contents of document/specifications outside **of** de jure standards. The bad examples are of reference to Unicode book and a **RFC.**

Since those documents are out of control of standard body, once the documents are revised, a once-conforming implementation of this standard may suddenly become non-conforming.

Therefore, only international de jure standards, i.e: ISO/IEC, can be referred to by normative part of this standard. For example, reference to Unicode book should be replaced with the reference to the ISO/IEC 10646, If there is no existing ISO/IEC standard that is equivalent with the technical contents of what referred to by this standard, a clause or a normative annex should **be provided in** this standard, then specify the
technical contents in the clause or annex.

1.3 "Discussion" clause

ECMA-262 has clauses named "discussions." According to the IS0 directives part 3. these clauses ehould be normative portion of this standard and the contents in the clauses are included *in the* requirements for conformity.

However, my impression on these is different. They sounds more like private notes or memoranda.

If the contents of the discussion clauses does not have requirements *for conformity, the* clause should be NOTESor it should be clarified that the discussion clauses are informative portion of this standard in the conformance clause.

2. Data representation in a datatype

Programminglanguage standard that does not have binary/object level portability as its objectives should not specify data representation of a datatype. in order not to restrict freedom of implementation.
In order programming language standards to be independent from any encoding *technology,* the datatype should be specified by repertoire of data that the datatype **can** *contains.*

In this sense, the String type should be specified as the set of all finite ordered sequence of zero or more character datatype, then should have a definition of character type as that the repertoire of the character data type shall be whole/entire repertoire of 1SO/IEC 10646,

Note that the ISO/IEC 10646 has the concept **of** subset, so if this standard allows an implementation that support a subset **of** ISO/IEC 10646, the minimal subset should be specified by this standard and actual repertoire of character should becomes implementation defined.

The same thing can be applied for the Number type. Usually, a datatype for numeric data is specified by limits of the value, e.g., -128 through 127.

If this standard need to have a wide range of exact integer values, e.g., -2^40 through +2^40 to **assure the exact calculation** of Date values **in milliseconds,** this standard should specify so, instead of referring to IEEE 754 **and** concluding the integer value range.

Also. if this standard need some severe requirement on the precision of real (floating) **values, tbis** standard should specify so by giving necessary minimum requirements.
Many **programming** languages have tried to make their specifications as, "cross-platform' as possible from the users' point of view. especially for the **purpose** that numerical algorithms can be programmed in "cross-platform" way.
They specify minimum ***requirements (for all*** implementation) and introduce constant names such as MAX_VALUE, MIN_VALUE etc.
to make platform defined values available to users.
Otherwise, an implementation that uses **a** representation which precision is more accurate/large than IEEE 754 becomes not conformity **to** this standard.

For those point. it might help to consult
ISO/IEC 11404:1995 Information Technology - Language Independent datatypes.
ISO/lEC 10967:1994 Information Technology . Language Independent
Arithmetic - part 1: Integer and floating point arithmetic.

JNB is skeptical if ECMAScript need special values such as NaN, Infinity, etc. for itself. Infinities are returned when the computation yields "overflow"; ECMASoript has no "Notification" mechanism to handle "overflow': and it might be a way to continue the computation without interruption that ECMAScript requires Infinities as continuation values in those cases. But NaN is yielded only when some of arguments ie already a NaN. ECMAScript could permit an implementation which ***has*** representation of Infinities but of NaN.

JNB ie also skeptical if ECMAScript need such a high precieion as **of** current specification on floating point computation. Thin standard **requires** some specific real values, such as E, PI, LNlO, be availabIe as closest possible floating numbers in 53 bite accuracy. Nevertheless all the defined functions are left unspecified about their returning values accuracy. If ever high **precision** computation were mandatory to ECMAScript, those functions should have been specified with severe accuracy requirement on their results.

De jure standard should not hinder the future improvement of technology

as far as possible. Note that some programming language **that** has
requirements on binaryportability, such as BYTE CODE of Java, may
need to specify internal representation **of** data in aadatatype. But,
JNB does not think that ECMAScript has such binary portability requirement.

For improvement of this standard, JNB would suggest that this standard
ehould align with recently approved ISO/IEC TR 10176 and IS0 standard
regarding language independent data types.

3. Character related issues

3.1 Repertoire of *charter*

In the clause **of** X5.3.2, the String.fromCharCode is specified. This method is specified based
on an assumption that BMP of ISO/IEC 10646 can contains every character in the world, since
the method hae 16bit dependency.

Per request from users, right now ISO/IEC JTC l/SC2 is **working to specify** additional
plains of ISO/IEC 10646 with the understanding of 16bit space is not sufficient to
encode characters required for some applications.

Therefore, tbis standard also should not have the 16bit dependency, then the return
value *of* the method should be a integer value regardless of 16bit or 32bit.

In addition to the above particular problem, JNB does not understand
why the method need to be standardized, since I/O functionality
is outside of this standard, and this standard allows addition
of methods and properties to conforming implementations.

If this standard include I/O function as JavaScript or JScript hae, JNB can understand
the requirement to convert *character to* character code supported by its platform.
but it is not the case of ECMAScript.

Also, it is quite bad programming manner to check an attribute of character
from its code point. If there is any requirement to check an attribute of character,
a future revision *of this* standard *should* provide such fuuctionality in the manner
being "locale" sensitive. One suggestion might be simply remove the method from
the standard, and make it enhancement by the specific implementations.

3.2 Upper-/lower.casing

It is widely understood that upper and lower-casing rules are language and/or culture
dependent. Therefore, if **such** language sensitive case conversion in the requirement. the
functionality should be provided in *the manner* of "locale" sensitive in a future *revision* of

this standard.

If not, this standard should specify minimum requirement that is common to every language, such as case conversion rule for the Latin characters specified **in ISO/IEC 646, then** behavior of outside of ISO/IEC 646 should be specified as implementation defined.

3.3 Character literal outside of ISO/IEC 646 repertoire

The ECMA-262 allows literal representation method such as YuXXXX.
Having the format is fine, but the description should be specified, such as
"Character short identifiers headed by Yu are defined as follows. "

Per *request* from the ISO/IEC JTC 1/SC22. the ISO/IEC JTC 1/SC2 **proposed a** short hand representation of character name that has one to one correspondence with character long name used throughout of IS0 character **set** standard, i.e. character code point **of** ISO/IEC **10646.** ***then the*** second edition **of the** ISO/IEC TR 10176 that is approved recently recommended the **support** of **the form** of literal representation **in order to specify character literal in an** character code independent manner.

The character sbort identifier looks very similar to the code point of ISO/IEC **10646, but** the big ***difference*** would be the correspondence between the character short identifier and a character will be maintained even if code point assignment *of* ***the cbaracter*** ie changed by a corrigenda or an amendment **of** ISO/lEC 10646.

JNB thinks, the change *of* the definition of YuXXXX may not impact to the actual technical contents of this standard, but contribute to make **the** standard independent from any encoding sytem.

Also, if the comment 3.1 ie accepted, JNB would suggest ***to*** specify YuXXXXXXXX form in additionto YuXXXX, so that the character **included** in ISO/IEC 10646 but allocated outeide of BMP becomee able to be represented.

4. Date Object

4.1 Two digit representation of year

As discussed in the ECMA-262, Date.prototype.getYear() and setYear() has 'year 2000 problem". If the rationale of inclusion of theee functionality is only backward compatibility, those methods should be removed form this standard, because
(1) this is the **first** edition of ECMAScript therefore there is no **previous *version nor*** edition from the standard view point.
(2) this standard allows enhancement of properties and methods, therefore implementations of this standard can provide these methode **as *enhancement.***

Also, in some methods when the year value between 0 and 99 is specified.

the 1900 will be added to the value as a base. This specification may not
appropriate to the standard published in 1998 or 1999.

The default base should be removed or amended. For example add 1990 if the
value is somewhat between 70 and 99, and add 2000 if the value is between 0 and 69.

4.2 Local time and daylight saving time

The ECMA-262 have a concept of daylight saving time and functionality to convert local time with daylight saving time to UTC. However, without having any good mechanism the one to
one correspondence between local time and UTC can not be guaranteed.

Let's assume that at 2:00 of September lst, the local time will be back te l:OO.
ln the case, 1:30 of September lst should be converted to what value in UTC?

Until no good mechanism is provided, this standard should not support local time and daylight saving time.

Again, there is no problem from the view point of backward compatibility and
conformance. Implementation can provide those fonctions as extensions. UTC
support would be good enough for this standard. If further revision of this
standard introduce the concept of locale, the local time support should be specified with a mechanism of daylight saving support, at that time.

5. Ambiguous Syntactic Rules

12.5 The IF statement
The following sentence is mandatory just below the syntax rules:

'else' shall associate with the nearest 'if among
'if's in the same block (excluding those 'if's which
are contained in the inner blocks) that precedes the
'else'and has no corresponding'else'.

(The sentence is borrowed from the C language standard.)
Without such a sentence the syntax remains ambiguous,
since one cannot tell whether

```
  if(a==b) if(c==d) x= 1; else x=2;
```

means

```
(1) if(a==b) {if(c==d) x= 1- else x= 2;}
```

or

```
  (2) if(a==b) {if (c==d) x= 1.) else x= 2;
```

This phenomena has been well known for languages which
have both forms of ifO and if0elseO for conditional
branching. Pascal may be consulted with this. It goes:

The token 'else' shall not occur next to any if-statement
which has no 'else' corresponding to its 'if'.

Minor Technical and Editorial comments:

With ECMA-262, JNB found a lot of minor technical and editorial errors.
Therefore, JNB would stongly suggest that a technical corrigenda should be prepared of this
standard, and be republished after spell checking of the standard document.

The followings are just examples and not the complete list of errors. I'm afraid if
a work document was published instead of the final version by accident, since there are so many spell errors.

Minor technical errors:

M1 15.8.2.11 & 15.8.2.12

The behavior in the case that argument x is equal to y is not specified.

M2 .-From the syntactic rules of 11.4 and 11.5, we can produce not well defined arithmetic expressions such as "-3*-2."

Minor Editorial errors:

E1 3 Reference

Only de jure standard referred to be from normative part of the standard that is related with conformance of this standard should be listed in the reference clause. Every reference documents outside of de jure standards and the ones just help to understand about the technical contents this standard should be removed from the clause or move to an informative
annex.

E2 4 Overview

The first word of the clause 4, i.e. "EMCAScript" should be replaced with "ECMAScript"

E3 1 Scope

We need more than one liner to define the scope of the standard.

E4 2 Conformance

"section 0" should be "section 7.4.3"

E5 4.1 There are "server-side" and "server side." They should   be one
representation either with or without hyphen.

E6 4.2 In the last sentence of the last paragraph, "anddefined" should be
"and defined."

E7 4.2.1 In the second sentence of the second paragraph, "aprototype"
should be "a prototype." In the figure. "Cfp" should be "CFp" in order to be consistent
with the description below.

E9 4.3.15 In some Cases, "Boolean object" is used, but in some other
cases, "boolean object" is used. Whether a word like "boolean'
should start with a capital letter or not, should be
consistent throughout the document. The same thing applies to
4.3.21 of "Number object" and "number object,"

E9 5.2 In the last sentence of the third paragraph, "mustbe" should
be "must be."

E1O 7.7.8 In the last paragraph, last part parenthesis of the second
sentence,
"(in the sense defined in section 8.4)", the referred section number should
be "8.5."
Also, in OctalIntegerLiteral ::
0 OctalDigit

OctalLiteral OctalDigit

OctalLiteral" should be 'OctalIntegerLiteral."

E11 8.6.2.3 The item7.,"Return Result(4)"should be"Return Result(6)."

E12 8.7 ln 4th paragraph,"A Reference consist of two parts, the base
object
and the property name" should be clarified such as "A Reference consista of two
components, the base object and the property name" so that the "part" is actually
the "component."

E13 8.7.4 In the item 6, there should be a forward reference to section
10.1.5 for the newly appeared undefined term, 'global object."

E14 Section reference. There are three types of section refernce in parentheses
(see section x.y.z), (section x.y.z), and (x.y.z). It would be consistent and
much better to use the one style instead of mixing various formats.

E15 8.9 In "abrupt completion," the word "completion" should also be italicized.

E16 9.1 In the column of Object, "(see section 8.6.2.5)" should be "(see section 8.6.2.6)."

E17 9.5 In the step 6, "Result(5)" should bc "Result(4)."

918 10.1.l and many other sections.There are two formats for "implementation dependent"; with and without a hyphen. It would bc better to define the technical term "implementation-dependent" and use it throughout the document.

E19 10.2.4 In the 4th bullet,"object object" should be "object."

E20 11.2.3 In the item 2, "section 0" should be "section 8.8."

E21 13 In the first paragraph,"the Identifier" is ambiguous in the sense

whether it is in the FunctionDeclaration, or in the FormalParameterList. It should be clarified such as "the function Identifier."

E22 15.1.2.4 In the item 7, "nonblank ASCII characters" should be "nonblank characters. " It should bc irrelevant whether they are ASCII or not.

E23 15.4.4.4 In the first paragraph and in the item l, the term,"this object," appears which causes some confusion whether "this" is a special "this" or not. Use "the object" or "the array object" to avoid the misleading.

E24 15.4.4.5 In the first paragraph, the second sentence, "The sort is not

necessarily stable." may cause misunderstanding to whom those do not have expertise in the sorting. The precise meaning of the word "stable" should be added or explained.

E25 15.5.3.2 In the first sentence, "asthe" should be "as the."

E26 15.9.1.1 In the 4th sentence, constants "iMin" and"iMax"appear as if they are ECMAScript constants. However, it looks just a convention in this place, and not used anywhere else. Avoid these constants.

END OF BALLOT--r-@z=

Subject: NNI's vote for ISO/IEC DIS 16262 ECMA Script (JTClSC22)
Date: Tue, 07 Apr 1998 11:20:27 +0l00
From: John Bijlsma <John.Bijlsma@nni.nl>
To: votes@iso.ch
CC: nni38122@mailsrv.twi.rudelft.n1

1998 -04 0 7

ISO/IEC DIS 16262
ECMA Script: A genera? purpose, cross-platform programming language
1998-04-09 APPROVAL WITH COMMENT

The NNI wants to make a number of comments on this DIS.
These comments follow the structure of the document and have been categorized as editorial (ed), minor (mi) or major (ma).

1- General comment:
It is disappointing that this document contains quite a large number of typos and some misplaced sections.
We think that the fast-track procedure has not been intended for such textually immature documents.
Careful inspection by the editor would have uncovered many problems.
Below some of these problems have been indicated: we are unsure we caught all.

2- Contents:(ed)
It is requested that annexes containing the collected syntaxes will be provided in the final document.

3- Section 2 (ma)
The conformance clauses in this section, and in particular the last one leave too much room for non-standard extensions to the language.
Such extensions will lead to portability problems.
It is unclear how conformity of implementations will be checked against conformance clauses as have been given here.

4- Section 2 (ed)
-- Typo: specificaitions
-- section 0?

5- Section 3 (ed)
It is requested that, where possible, references to IS0 standards will be provided.
The following standards have been referenced in the document, but are not mentioned here: ASCII, HTML.

6- Section 4 (ed)
A scripting language is intended for use by both professional and non-professional programmers and therefore there may be
-- The implication shovn by the use of 'therefore' is unclear.
-- The sentence seems incomplete.

7- Section 4.1 led)
A web browser provides en .,... (isn't this prescribing too much? 4 times)
--> A conforming web browser can/may provide en .
Typo: error: and abort
Typo: clients, and files. and

8- Section 4.2 (ed)
It is unclear how a syntax can be 'relaxed'.
A syntax is simply a description.
Typo: anddefined
Typo: missing full stop

9- Section 4.2.1 (ed)
Typo: aprototype
Comma: contains. share

1998 -04- 0 7

Figure: Cfp -> CFp
Figure; There iS no meaning given for the normal arrow used form CF to CFp.

lo- Section 4.3.1 (mi)
A type is a set of **data values.**
-- Are non-homogeneous sets allowed?
. In general, the correct functioning of a program is not affected if different data values of the same type are substituted for others.
Well it depends upon what is meant by 'in general' and 'correct', but as a general statement this seems to be incorrect for any programming language.'

ll- Section 4.3.9 (mi)
The notion of a 'variable' has not been defined in this section.

12- Section 4.3.15 (ed)
This is an example of ..
-- This sections seems to be misplaced.

13- Section 4.3.16 (ed)
of the type String and is the set of .
-- the latter part of that sentence seems misplaced (see also 4.3.17)

14- Section 4.3.19 (edI
... a number value _is_

15- Section 5.1.2 (ma)
. . It defines a set of productionsstarting from the goal symbol _Input_. that . . .
-- This svmbol cannot be found in section 7.
Common programming languages do not need full parsers for analysing the lexical structure of a program text.
The set-up **of** this pars er cannot be determined because the structure of the grammar is unclear.

16- Section 5.1.2 (mi)
A multi-line comment is likewise simply discarded if it contains no line terminator: but . . .
-- it is unclear how a _multi-line comment cannot contain a line terminator (but see also a later comment)

17- section 5.1.4 (mi)
. if an end-of-line character ___
-- is an end-of-line character equivalent co a line terminator?

18- Section 7.3 (mi)
The description is unclear about Line terminators in Multi-line comments

19- Section 7.3 (mi)
The production for
```
  MultiLineNotForwardSlashOrAsteriskChar ::
     SourceCharacter but not forward-slash / or asterisk *
```
seems to be better written as:
```
  SourceCharacter but not ( forward-slash / or asterisk • I
```
This case occurs more often.

20- Section 7.8.1 (ed)
The notion of 'the header of a for statement' has not been defined.

21- Section 8
There are six standard types . .
In section 4.2 these are called built-in types.

22- Section S.6 (mi)
Each property consists of a name. a Value and a set of attributes.
This seems inconsistent with section 4.2

23- Section 9.1 (ed)
The table contains an incorrect reference to section 8.6.2.5

24- Section 10 (ed)
. When control is transferred to ECMAScript executable code, we . .
The use of 'we' is not common in standardization documents.

25- Section 10.1.1 third bullet (ed)
. . .The use of these attributes are described . . .
is described, is probably intended here.

26- Section 11.11 (ed)
6 Call GetValue((Result(5))
Bracket mismatch

27- Section 12.2 (ed)
The reference to section 0 seems incorrect.

28- Section 12.5 (mi)
The syntax given here allows for so-called dangling else problems.
These seem not to be resolved.

29- Section 14 (ed)
. 1. Process SourceElements for function declarations ..
From the description given, it can't be determined whether this needs
to be done left to right or right to left.
1. Evaluate SourceElements.
....
. 4. Return Result(l)
It is unclear whether the result of step 4 is the result of the first term
or of the last term of (1).
On two occasions 'is' is printed in italics.
On one occasion 'is' is written vithout preceding space.

30- Section 15.1.3.5 (ed.)
. 15,6,2.
Commas?

31- Section 15.9.1.8 (ed)
. . /t.... ??

eof

J.(John) Bijlsma NNI/NEC
standardization consultant
inform.tech. & celecomm.
tel. +31 15 2690126 fax:+31 15 2690242
p.o.box 5059
2600GB DELFT Netherlands
john.bijlsma@nni.nl
~

1998 -04- 0 6

Subject: US **vote on ISO/IEC DIS 16262**
Date: Fri. 3 Apr 1998 16:06:50 -0500
From: Matthew Deane <mdeane@ANSI.org>
To: "'DIS Votes (ITTF)"' <votes@iso.ch>
CC: 'Deborah Donovan' <ddonovan@itic.nw.dc.us>

```
> Please accept this email transmission as official notification of the
> US National Body vote forISO/IEC DIS 16262. ECMAScript: A general
> purpose, cross-platform programming language.
>
> The US National Body votes to Approve with the comments below.
>
> Regards.
>
> Matthew Deane
> For the US P-member JTC l/SC 22
> ******************************************
>
> The US National Body votes to Approve with Conrments ISO/I.EC DIS 16262,
> ECMAScript; A general purpose, cross-platform programming language.
> See comments below.
>
> Comments A,B. and F are editorial.
> Comments C, D, and G are technical.
> Comment E is general.
>
> ---------------------------------------------------------
> A) Comment set #l:
>
> Line numbers are relative to the start/end of the corresponding
> section
> (or page if there is a page break in a section).
>
> ---------------------------------------------------------
>
> 1) pp. 1, 2 Conformance, line -1.
>
> Replace 'section 0' with appropriate section number.
>
> 2) PP. 1, 3 References, line I.
>
> ANSI X3.159-19989 is the original C standard. That was withdravn and
> replaced by the ANSI/IS0 C standard ANSI/IS0 9899:1990, adopted in
> 1990. (An addendum was added in 1996, but I chink the 1990 version
> reference will be sufficient.)
>
> 31 pp. 1, 4 Overview, line -3
>
> Re 'informalities and build', either strike 'and6r add the missing
> noun
> that should follow it.
>
> 41 PP. 2. 4.2 Language Overview, line -4.
>
> Should Java be indicated as a trademark here (or possibly in the front
> matter)?
>
> 5) pp. 2, 4.2 Language Overview, line -2.
>
> add a space in 'anddefined'.
>
> 6) pp. 2. 4.2.1 Objects. line -3.
>
> Add a space in 'aprototype'.
>
, 7) PP. 3. 4.2.1 Objects, line 8.
```

1998 -04- 0 6

>
> 'The following diagram may illustrate this discussion:' That doesn't
> sound like it definitely does. Either make the diagram illustrate it
> or improve the wording.
>
> 8) pp. 3, 4.2.1 Objects, line 13.
>
> Strike the colon.
>
> 9) pp. 3, 4.2.1 Objects. line 15.
>
> 'on the fly' sounds a bit colloquial to me. How about 'dynamically' or
> 'at
> run time'?
>
> 10) pp. 3. 4.2.1 Objects, line 16.
>
> Re 'any of its properties.' What is the subject referred to by 'its'?
> I'm
> guess its refers to an object, but 'its' is singular and 'objects' is
> plural.
>
> 11) pp. 3, 4.3.3 Object, line 1.
>
> Change 'properties which contain' to 'properties each of which
> contains'.
>
> 12) pp. 4, 4.3.9 Undefined, heading.
>
> Add 'value' to the heading as in 4.3.13 and 4.3.16.
>
> 13) pp. 4, 4.3.11 Null, heading.
>
> Add 'value' to the heading as in 4.3.13 and 4.3.16.
>
> 14) PP. 4. 4.3.13 Boolean value, line 1.
>
> Strike 'either'.
>
> 15) PP. 4, 4.3.15 Boolean object, line 5.
>
> Replace 'in this case it is' with 'the ability'.
>
> 16) pp. 4. 4.3.16 String value, line 1.
>
> It seems to me that a string value is one of the set of all finite
> ordered
> sequences not the whole set.
>
> 17) pp. 5, 4.3.19 String object, line 1.
>
> A number value [is] a.
>
> 18) PP. 5, 4.3.20 Number type, line 1.
>
> "In ECMAScript the set of values represent the double-precision
> 64-bit
> format IEEE 754 value ...'
>
> sounds like there is only 1 64-bit format value. Perhaps it should say
>
> "In ECMAScript the set of values represent all the double-precision
> 64-bit format IEEE 754 values including the special "Nor-a-Number"
> (NaNI value, positive infinity. and negative infinity."
>
> 19) pp. 5. 4.3.22 Infinity. heading.
>
> Add 'value' to the heading as in 4.3.13 and 4.3.16.

>
> 20) pp. 5, 4.3.22 Infinity, line.
>
> Is 'Infinity' set in the correct typeface? The values true and false
> are
> set differently in 4.3.13.
>
> 211 pp. 6, 5.1.5 Grammar Notation, line -7.
>
> Change 'recursive, that is to say' to 'recursive; that is'.
>
> 2.2) pp. 8, 5.2 Algorithm conventions, heading.
>
> Upcase C in 'conventions' to match all other level-2 heads.
>
> 23) PP. 8, 5.2 Algorithm conventions, line 12.
>
> Add a space in 'mustbe'.
>
> 24) PP. 9, 7 Lexical conventions, line 1.
>
> The source text of a[n] ECMAScript program .
>
> 25) pp. 9. 7 Lexical Conventions, line 2.
>
> A token is a sequence that comprise [s] .

> 26) pp. 9. 7.1 White Space, line 2.
>
> . each other[,] but . .
>
> 27) pp. 10, 7.2 Line Terminators, line 1.
>
> Replace
>
> "Line terminator characters, like whitespace characters, are used to
> improve source text readability and to separate tokens (indivisible
> lexical units) from each ocher. Unlike whitespace characters, "
>
> with
>
a "Like whitespace characters, line terminator characters are used to
> improve source text readability and to separate tokens (indivisible
> lexical units) from each other. However, unlike whitespace characters,
> . .."
>
b 28) pp. 13. 7.7.3 Numeric Literals, line -4.
>
> '. ideally using IEEE 754 round-to-nearest ...' The word 'ideally'
> doesn't sound like good standard's language. What's the implication if
> the implementer doesn't use this?
>
, 29) pp 15, 7.7.3 Numeric Literals, line 7.
>
> The use of nested parentheses is rather unusual. How about replacing
>
> ''A digit is significant if it is not part of an ExponentPart and
> (either it
> is not 0 or (there is a nonzero digit to its left and there is a
> nonzero
> digit not in the ExponentPart. to its right))."
>
> with
>
> "A digit is significant if it is not part of an ExponentPart and
>
> -- either it is not 0 or,
>

> -- there is a nonzero digit to ies left and there is a nonzero
> digit,
> not in the ExponentPert, to its right."
>
> 30) pp 18, 7.8.1 Rules of automatic semicolon insertion, line 14.
>
> Replace "These are all the restricted " to "These are the only
, restricted ...'
>
> 31) pp. 19, 8 Types, line 2.
>
> Strike 'called' and put the list "Reference. List, and Completion'
2 in
> parens as for the six standard types in the line above.
>
z 32) pp 19, 8.3 The Boolean Type, line 3.
>
> Replace
>
> "The Boolean type represents a logical entity and consists of exactly
> two unique values. One is called true and the other is called false."
>
> with
>
> "The Boolean type represents a logical entity having two unique
> values,
> called true and false."
>
> 33) pp. 20, 8.5 The Number type, line 7.
>
> Instead of "... all NaN values are the same." might it be better to
> say
> that "... all NaN values compare equal"?
>
> 34) pp. 22, 8.6.2 Internal Properties and Methods, line 4.
>
> Add ", respectively" to the end.
>
> 35) pp. 22, 8.6.2 Internal Properties and Methods. line -2.
>
> ... implement[ation]-dependent ...
>
> 36) pp. 22, 8.6.2 Internal Properties and Methods, line -7.
>
> Re 'it is used internally . .." is the subject 'ic' referring to the
> value
> of a ((Class]] property? It's probably worth naming the subject
> explicitly.
>
> 37) pp. 30, 9.8 ToString, line 1.
>
> Re 'attempts', what happens if it cannot convert its argument? I see
> no
> provision for the generation of a runtime error (like 9.9 provides on
> conversion failures).
>
, 38) pp. 33. 10.1.3 Variable instantiation. lines l-2.
>
> Remove extra vertical space between these lines.
>
> 39) pp. 33, 10.1.3 Variable instantiation, lines 6-7.
>
> Remove extra vertical space between these lines.
>
z 40) pp. 44, 11.7.1 The left shift operator (<<), line 1.
>
.' Replace both occurrences of 'argument' with 'operand'.
>

> 41) pp. 44, 11.7.2 The signed right shift operator (>>), line 1.
>
> Replace both occurrences of 'argument' with 'operand'.
>
> 42) pp. 45. 11.7.3 The unsigned right shift operator (>>>], line 1.
>
> Replace both occurrences of 'argument' with 'operand'.
>
> 43) pp. 55, 12.7 The CONTINUE Statement, line 4.
>
> Re '. . . may not be executed ..' The use of 'may' in a formal
> specification can be troublesome, especially when used with the
> negative. Specifically, does 'may not' mean 'might not' or does it
> mean 'shall not'? One imposes conformance requirements while the other
> doesn't.
>
> 44) pp. 55. 12.7 The CONTINUE Statement line 5.
>
> Replace 'at least one' with 'a'. It seems to me that the number of
> nested
> while or forsfatements is irrelevant..
>
> 451 pp. 55. 12.8 The BREAK Statement, lines 4 and 5.
>
> See the comments for CONTINUE above.
>
> 461 pp. 55. 12.9 The RETURN Statement, lines 4.
>
> See the first comment for CONTINUE above.
3
> 47) pp. 68, 15.4.4 Array.prototype.sort(comparefn). line -6.
>
> Replace 'compared' with 'compares'.
>
> 48) pp. 77, 15.8.1.5 LOG1OE. line 2.
>
> Re "(Note that the value of Math.LOG2E is approximately the
> reciprocal of thevalue ofMach.LN2.j"
>
> I'm no mathematician. but I'm guessing that this section was cloned
> from 15.8.1.4, and that it should read as follows instead:
>
> "(Note that the value of Math.LOGlOE is approximately the reciprocal
> of
> the value of Math.LNl0.I"
>
> 49) pp. 77, 15.8.2 Function Properties of the Math Object. line -2.
>
> Re '[XXXREF]', is a cross-reference missing here?
>
> 50) General comment. Some level-2 and level-3 headings have each word
> with a leading capital letter and some don't. Make them consistent.
>
> 51) General comment. It would probably be an improvement to set all
> reserved words, function names, and operators in headings in a
> constant-width typeface. Having the level-2 heads 12.7 to 12.10 be all
> caps whereas those in 12.6.1 to 12.6.3 are not, looks strange.
>
>. ---
>
> B) comment set #Z:
>
> The term "runtime error", "compile-rime error" and "error" are all
> used.
> but not defined. This document should define how a "compile time
> error'* is recognized, for example by an implementation defined
> diagnostic message or return code Same with "runtime error" and error.
> Without these definitions measuring conformance will be impossible.

>
> The reference to the C standard should be ISO/IEC 9899:1993, not the
> ANSI document.
>
> --
>
> C) Comment set #3:
>
> Paragraphs 15.9.5.5 Data.prototype.gecYear() end 15.9.5.38
> Data.prototype.setYeerO should be removed because, as noted in
> ECMA-262, they may contribute to the "Year 2000" problem. The
> rationale for their inclusion. backwards compatibility (with what, as
> there are no prior standards), is not sufficient for such a strongly
> deprecated programming practice.
>
> --
>
> D) Comment set #4:
>
> (I had some problems with page numbering when reading the text
> on-line. but I have tried to link the reference to the numbers that
> appear on the pages themselves in the PDF version.)
>
on p. 1. Section 3 (references). the references should be to the
> ISO/IEC
> versions of the standards mentioned.
>
> On p. 2, Section 4.2. "anddefined" --> and defined
>
> on p. 3. Section 4.2.1. 'aprototype" -> a prototype
>
> on p. 3. Section 4.3.1 (type), "A type is a set of data values. In
> general, the correct functioning of a program is not affected if
> different data values of the same type are substituted for others."
> This sentence is unclear because the correct functioning of a program
> does depend on the proper sequencing of values.
>
> on p. 9. Section 6. they describe the use of Unicode in comments and
> string literalsls We program in English and seldom use Unicode, I would
> want to be sure that others feel chat the approach here is consistent
> with other programming languages and the intended use of Unicode.
>
> on p. 10-11, Section 7.3, the syntax for comments seems more
> complicated than necessary particularly whe things like special
> Unicode characters are described earlier. Is this the way the similar
> syntax is done in C or C++?
>
> On pp. 11-12. Sections 7.4.2 and 7.4.3, keywords and future reserved
> words, it is not clear whether the case of the characters is
> significant.
> I thought ECMAScript was case sensitive, but I didn't see that
> mentioned.
>
> On p. 20. Section 8.5 (and then other pages and sectionsl, the number
> type is described in terms of IEEE 754. (Isn't there an ISO/IEC number
> for this standard?) In Section 11.5.3, they have some differences in
> the % operator. There has been some discussion about how this
> standard is used in Java. I would hope chat things are done
> appropriately here. The use in this proposed standard (and in Java and
> other languages) might motivate a review of IEEE 754.
>
> On p. 60, Section 15.1.2.4 (escape string). it's not clear why a
> different
> format is being used. There also seems to be over specification of
> some of the rules in this and surrounding sections.
>
> On p. 76, Section 15.7.3.2 (Number.MAX_VALUE) starts "The value of
> Number.MIN_VALUE ..."

>
> ____~_________________________~~---~~____--~~~__~__-_--______--__
>
1 E) *comment set* #5:
2.
> *During the standarditazion process for ECMA 262,the naming of the*
> *standard was dealt with in an unsatisfactory manner with a less than*
> *optimal result. The original name put forward to the committee,*
> *LiveScript, was agreed on by all members but was withdrawn at the last*
> *moment with no alternative proposed. This resulted in a standard*
> *named ECMAScript Because this is not a copyrighted or trademarked*
. Label it is *unlikely that any implementation of the language will be*
>*called* ECMASCript. This *has and wi11 result in confusion for users as*
3 *to what the standard means and* what Language engines *support the*
> *standard.*
>
2. *We believe that in* the case *of standards applicable to the mass market*
> (where *both the contributors to the standardization effort and the*
> *public at large have an expectation of interoperability) names are*
> *very*
> *importantn both asn a*indicator *of the openness of the process where*
> *the parties cooperating expect to* begin to *compete* from a common
> *footing at the end of the process, and as an assurance co the public*
> *that their* expectations are forthrightly met (e.g. *codewords and*
> *numbers are*
> unacceptable *when assuring* the *public* that they can connect an
> appliance into a wal1 outlet). *Where the community of customers are*
> *small or well informed, this is less an issue than in the case of*
> *ECMA-2* 62.
>
> *Also, the lack of a marketable name and ownership (or at least*
> *unencumbered use)* of a popular name *by ECMA (or any standards* body
> *chat is unable to assert ownership of their efforts)* for a *highly*
> *visible*
> interoperability *standard* diminishes chat *body's ability to generate*
> *revenues from the* publication *and ongoing* maintenance of the
> *definitive*
> *standard. This can be seen in the commercial publication of other*
> *"standards" in the same* discipline *without regard for the hosting*
J *organization's need for a sustaining revenue stream Lack of*
> *ownership*
-I will bias *future efforts cowards less formal consensus efforts not*
3 *dependent on publication revenues and further undercut the traditional*
3 *standards bodies.*
>
> ---
>
> *F) Comment set #6:*
>
> **Missing:**
>
> *If appears chat "undefined behavior" is not* defined *anywhere*
>
> *Missing definitions:*
>
> *client-server architecture*
> client-side
>
> Replacement:
>
> Unicode *should* be *replaced with* IS0 10646-l BMP (universal charactet)
> *throughout the document.*
>
> *Look for all copies of the word "we" and replace* with *standards*
> *uording.*
>
> === *Specific Comments:*
>

1998 -04- 0 6

s- Page vi: typo "I" on top of page in PDF version.
>
> clause 2. page 13:
>
> If a conforming implementation can support *any" syntax, then how are
> conforming implementations tested? The conformance clause should be
> worded differently to identify non-conforming programs.
>
> Clause 3, page 13:
>
> Change reference of ANSI C to ISO C.
>
> Add URL for RFC 1738.
>
a Unicode should be replaced with IS0 10646-l BMP
>
> Subclause 4.3:
>
> This should be broken out as a separate clause. The beginning of
a clause 4 implies that the clause is informative; it appears that
> subclause 4.3
> should be normative.
>
> Subclause 5.2:
>
> Remove paragraph 1, "We often use ..."
>
a Replace "X is Y" with wording that uses "shall".
>
> Clause 6:
>
> ASCII is mentioned but no reference in the References clause.
>
> An IS0 standard should be referenced rather than the ASCII standard.
>
> Subclause 7.2:
>
> Line terminators should include the Next Line character (I think is it
a 0x84 or 0X85).
>
> Subclause 7.3.3:
>
a Bullets on page 26: exponents appear in font too small.
>
> Subclause 10.1.1
>
> change to standards wording "which we refer to..."
>
> subclause 10.1.3
>
> Vertical whitespace after first sentence -- remove ic
>
> Subclauses 12.8, 22.9, 12.10:
>
> These subclauses refer to a program as "syntactically incorrect', but
> the corresponding behavior is not clear: what happens when the program
> fails. Furthermore, this response to bad syntax should be defined
> early
> in the document, say, in the Conformance clause.
>
> Subclause 15.9.1.1:
>
z Reword "This span easily covers all of recorded human history ..." as
> specific year numbers in the common era (A.D.) and before the common
> era (B.C.,.
>
> subclause 15.9.1.3. para 2:
>

1998 -04- 0 6

> Remove/reword "Of course"
>
> Subclause 15.9.1.8
>
> Reword "by whatever means available" as standards words
> ("implementation-defined"? -- but this would require defining
> implemencacion defined).
>
> Paragraphs 2 and 3: It is not clear what the required behavior is for
> a
> conforming implementation.
>
> Clause 16:
>
> This clause should be moved to the beginning and. possibly,
3 incorporated
> in the Conformance clause.
>
> ____________________________-__-___-__________________________________
>
> GJ Comment set No. 7
"
> BACKGROUND
>
> The year 2000 date rollover problem is having profound effects on
> government and industry software systems worldwide. Systems are
> already failing due co the erroneous processing of dates that include
> the year 2000. Most software systems are programmed to recognize
> 2-digit years in dates a.5 occurring within the 20th century, with the
> implication that the first two digits of the year are 19. when the
> year 2000 comes into play. these software systems will more than
> likely recognize 00 as 1900 instead of 2000. causing consternation
> among users system managers. and database administrators. Billions of
> dollars are being spent on fixing the problem.
2.
> The U.S. Congress is holding frequent hearings on the progress that
> Federal agencies have made in repairing systems that support a myriad
> of government programs.
>
> A meeting among representacives of the U.S. Office of Management and
> Budget (OMB). 27 Federal agencies, and 41 states in October 1997
> affirmed the action co requite 4-digit years in all dates used in data
> interchange between the Federal and state governments. and that the
> Federal government would act as lead in further actions as necessary.
> Further, the U.S. Securities and Exchange Commission (SEC) is
> examining requirements for publicly held companies to disclose the
> extent of date processing problems and plans for correcting these
z. problems within their mandatory filings. Other Federal agencies. most
1 notably the Nuclear Regulatory Commission (NRC), the Federal Aviation
> Administration (FAA) rhe Environmental Protection Agency (EPA), the
> Food and Drug Administracion (FDA) and others are developing
> implementation plans for overseeing the correction of date
> processing problems within regulated industries. NIST spearheaded the
> government's position when it issued Change Notice 1 to Federal
>information Processing Standard (PIPS) 4-1 in March 1996 which highly
z recommended the use of 4-digit years and deprecated the use of
> optional 2-digit years.
>
> The National Committee on Information Technology Standards (NCITS --
> formerly X3) Technical Committee LB has recommended a new date format
> interchange standard that provides for only a 4-digit year format
> (NCITS L8 3.30). The recommended standard has been forwarded to ANSI
> which has placed it on its list of standards co be published. The
> 2-dlgit year format has been excluded. The incernational standard, ISO
z 8601:1988 (under the auspices of ISO TC154J. has not been changed.
>
> REQUIRED CHANGES
>

2. The ECMAScript specification provides functions for processing Z-digit
> and 4-digit years directly (see sections 15.9.5.5. 15.9.5.6, 15.9.5.7,
z 15.9.5.10, 15.9.5.11, 15.9.5.36, and 15.9.5.38 and other functions
> that
> use 2-digit or 4-digit years based OR the prototype date arguments.
>
> The Z-digit yea: option should be left out of the specification
> entirely
> for the following reasons:
3
> 1. The Year 2000 problem is based on this option and will provide no
> end
> of frustration for implementers who have co justify why this option is
> still part of the ECMAScripc specification, in view of the attention
> the
> problem has received already.
>
> 2. The liability **of resellers** and implementers will come more **and more**
> **into** focus as users rely on the court system to determine who is
> responsible for correcting problems based on this option.
>
> 3. The specification allows for implementations with extended
> capabilities, as long as the extended capabilities do not cause
z erroneous operation of the standard requirements. The ECMAScript
> specification is clear in providing two sets **of** functions that treat
> dates differently. The lack of 2-digit date processing functions
> should, ostensibly, not interfere with 4-digit year processing.
> Two-digit year processing may be implemented within the realm of
z extensions without causing undue influence on the standard use of the
2. specification.
>
2. 4. Not withstanding the need to provide functionality of de facto
> implementations. notes in the specification co the effect chat these
> functions are provided **for** backward compatibility have no meaning
> with respect to this standard since there were no previous nationality
a or
a internationally recognized standards.