CONFORMANCE.

Updated Proposed Text for the 2nd paragraph of the "Conformance" clause of ECMAScript, DIS 16262, mailed on April 2, 1998:

"A conforming implementation of this edition of ECMAScript shall interpret characters in conformance with The Unicode Standard, Version 2.0, and ISO/IEC 10646-1 with UCS-2 as the adopted encoding form, implementation level 3. If the adopted ISO/IEC 10646-1 subset is not otherwise specified, it is presumed to be the BMP subset, collection 300."

Originally Proposed Text for the 2nd paragraph of the "Conformance" clause of ECMAScript, DIS 16262, mailed on April 1, 1998:

"A conforming implementation of this edition of ECMAScript shall recognise all the characters of The Unicode Standard, Version 2.0, and ISO/IEC 10646-1, UCS-2 as the adopted encoding form, implementation level 3 and the BMP subset, collection 300."

Please note that the text below is included here to explain the rationale of the proposed text and does not need to be included in the normative part of ECMAScript.

The above proposed text *does talk* about implementation level 3 as described in 10646, but it does not mean that ECMAScript would have to process a character and a combining sequence as yet another uniquely identified character. ECMAScript can treat a combining sequence as just another 16-bit value. Combining sequences are encoded for use with some base characters especially for Indic, Thai, Arabic and Hebrew scripts. Platforms and application software decides how to process a base character and a combining sequence. Note that combining sequences can be used for Latin as well but the vast majority of Latin script characters are already encoded in 10646. From a Unicode conformance point of view, what is important is that combining sequences are not damaged as they go through a process, such as ECMAScript. Unicode supports combining sequences and provides an equivalence algorithm that facilitates comparing precomposed characters with a base character followed by a combining sequence. All ECMAScript, and other programming languages, need to do is let a data stream of 16-bit values pass through the process cleanly and with no damage to the data.

REFERENCING.

Proposed text for referencing Unicode and ISO 10646 in clause 3:

Unicode: "Unicode Inc. (1996), *The Unicode Standard* TM, Version 2.0. ISBN: 0-201-48345-9, Addison-Wesley Publishing Co.: Menlo Park, California"

ISO 10646: "ISO/IEC 10646-1:1993 Information technology -- Universal Multiple-Octet Coded Character Set (UCS) -- Part 1: Architecture and Basic Multilingual Plane"

The additional information provided below is only for reference and not for inclusion in the ECMAScript normative part of the standard. The information highlights the contents of the conformance clauses of Unicode 2.0 and ISO 10646:

Unicode 2.0 Conformance:
This is spelled out in detail in Chapter 3, Conformance, of The Unicode Standard 2.0. Here are the bulleted items:

- **Byte Ordering:**
  - *C1 A process shall interpret Unicode code values as 16-bit quantities.*
  - *C2 The Unicode Standard does not specify any order of bytes inside a Unicode value.*
  - *C3 A process shall interpret a Unicode value that has been serialized into a sequence of bytes, by most significant byte first, in the absence of higher-level protocols.*
- ***Invalid Code Values***
  - *C4 A process shall not interpret an unpaired high- or low-surrogate as an abstract character.*
  - *C5 A process shall not interpret either U+FFFE or U+FFFF as an abstract character.*

  - *C6 A process shall not interpret any unassigned code value as an abstract character.*
- ***Interpretation***
  - *C7 A process shall interpret a coded character representation according to the character semantics established by this standard, if that process does interpret that coded character representation.*
  - *C8 A process shall not assume that it is required to interpret any particular coded character representation.*
  - *C9 A process shall not assume that the interpretations of two canonical-equivalent character sequences are distinct.*
- ***Modification***
  - *C10 A process shall make no change in a valid coded character representation other than the possible replacement of character sequences by their canonical-equivalent sequences, if that process purports not to modify the interpretation of that coded character representation.*

ISO 10646 Conformance clause:
This is spelled out in detail in clause 2, Conformance, of ISO/IEC 10646-1:1993. Here is the essence of that clause.

2 Conformance

2.1 General

Whenever private use characters are used as specified in ISO/IEC 10646, the characters themselves shall not be covered by these conformance requirements.

2.2 Conformance of information interchange

A coded-character-data-element (CC-data-element) within coded information for interchange is in conformance with ISO/IEC 10646 if
a) all the coded representations of graphic characters within that CC-data-element conform to clauses 6 and 7, to an identified form chosen from clause 14 or Annex Q or Annex R, and to an identified implementation level chosen from clause 15;
b) all the graphic characters represented within that CC-data-element are taken from those within an identified subset (clause 13);
c) all the coded representations of control functions within that CC-data-element conform to clause 16.
A claim of conformance shall identify the adopted form, the adopted implementation level and the adopted subset by means of a list of collections and/or characters.