This PDF file is an excerpt from *The Unicode Standard, Version 4.0*, issued by the Unicode Consortium and published by Addison-Wesley. The material has been modified slightly for this online edition, however the PDF files have not been modified to reflect the corrections found on the Updates and Errata page (http://www.unicode.org/errata/). For information on more recent versions of the standard, see http://www.unicode.org/standard/versions/enumeratedversions.html.

Dai Kan-Wa Jiten used as the source of reference Kanji codes was written by Tetsuji Morohashi and published by Taishukan Shoten.

Cover and CD-ROM label design: Steve Mehallo, http://www.mehallo.com

The publisher offers discounts on this book when ordered in quantity for bulk purchases and special sales. For more information, customers in the U.S. please contact U.S. Corporate and Government Sales, (800) 382-3419, corpsales@pearsontechgroup.com. For sales outside of the U.S., please contact International Sales, +1 317 581 3793, international@pearsontechgroup.com

Visit Addison-Wesley on the Web: http://www.awprofessional.com

Library of Congress Cataloging-in-Publication Data
The Unicode Standard, Version 4.0 : the Unicode Consortium /Joan Aliprand... [et al.].
p. cm.
Includes bibliographical references and index.
ISBN 0-321-18578-1 (alk. paper)
1. Unicode (Computer character set). I. Aliprand, Joan.

QA268.U545 2004
005.7'2—dc21

2003052158

ISBN 0-321-18578-1
Text printed on recycled paper
1 2 3 4 5 6 7 8 9 10—CRW—0706050403
First printing, August 2003

Chapter 4

Character Properties

> ***Disclaimer***
>
> The content of all character property tables has been verified as far as possible by the Unicode Consortium. However, the Unicode Consortium does not guarantee that the tables printed in this volume or on the CD-ROM are correct in every detail, and it is not responsible for errors that may occur either in the character property tables or in software that implements these tables. *The contents of all the tables in this chapter may be superseded or augmented by information on the Unicode Web site.*

This chapter describes the attributes of character properties defined by the Unicode Standard and gives mappings of characters to specific character properties. Full listings for all Unicode properties are provided in the *Unicode Character Database* (UCD).

While the Unicode Consortium strives to minimize changes to character property data, occasionally character properties must be updated. When this situation occurs, the relevant data files of the Unicode Character Database are revised. The revised data files are posted on the Unicode Web site as an update version of the standard.

Consistency of Properties. The Unicode Standard is the product of many compromises. It has to strike a balance between uniformity of treatment for similar characters and compatibility with existing practice for characters inherited from legacy encodings. Because of this balancing act, one can expect a certain number of anomalies in character properties. For example, some pairs of characters might have been treated as canonical equivalents but are left unequivalent for compatibility with legacy differences. This situation pertains to U+00B5 µ MICRO SIGN and U+03BC μ GREEK SMALL LETTER MU, as well as to certain Korean jamo.

In addition, some characters might have had properties differing in some ways from those assigned in this standard, but those properties are left as is for compatibility with existing practice. This situation can be seen with the halfwidth voicing marks for Japanese (U+FF9E HALFWIDTH KATAKANA VOICED SOUND MARK and U+FF9F HALFWIDTH KATAKANA SEMI-VOICED SOUND MARK), which might have been better analyzed as spacing combining marks, and with the conjoining Hangul jamo, which might have been better analyzed as an initial base character, followed by formally combining medial and final characters. In the interest of efficiency and uniformity in algorithms, implementations may take advantage of such reanalyses of character properties, as long as this does not conflict with the conformance requirements with respect to normative properties. See *Section 3.5, Properties*, *Section 3.2, Conformance Requirements*, and *Section 3.3, Semantics*, for more information.

4.1 Unicode Character Database

The Unicode Character Database (UCD) consists of a set of files that define the Unicode character properties and internal mappings. For each property, the files determine the assignment of property values to each code point. The UCD also supplies recommended property aliases and property value aliases for textual parsing and display in environments such as regular expressions.

The properties include the following:

- Name
- General Category (basic partition into letters, numbers, symbols, punctuation, and so on)
- Other important general characteristics (whitespace, dash, ideographic, alphabetic, noncharacter, deprecated, and so on)
- Character shaping (bidi category, shaping, mirroring, width, and so on)
- Casing (upper, lower, title, folding; both simple and full)
- Numeric values and types
- Script and Block
- Normalization properties (decompositions, decomposition type, canonical combining class, composition exclusions, and so on)
- Age (version of the standard in which the code point was first designated)
- Boundaries (grapheme cluster, word, line and sentence)
- Standardized variants

See the Unicode Character Database for more details on the character properties, their distribution across files, and the file formats.

4.2 Case—Normative

Case is a normative property of characters in certain alphabets whereby characters are considered to be variants of a single letter. These variants, which may differ markedly in shape and size, are called the *uppercase* letter (also known as *capital* or *majuscule*) and the *lowercase* letter (also known as *small* or *minuscule*). The uppercase letter is generally larger than the lowercase letter.

Because of the inclusion of certain composite characters for compatibility, such as U+01F1 LATIN CAPITAL LETTER DZ, a third case, called *titlecase*, is used where the first character of a word must be capitalized. An example of such a character is U+01F2 LATIN CAPITAL LETTER D WITH SMALL LETTER Z. The three case forms are UPPERCASE, Titlecase, and lowercase.

For those scripts that have case (Latin, Greek, Cyrillic, Armenian, Deseret, and archaic Georgian), uppercase characters typically contain the word *capital* in their names. Lowercase characters typically contain the word *small*. However, this is not a reliable guide. The word *small* in the names of characters from scripts other than those just listed has nothing to do with case. There are other exceptions as well, such as small capital letters that are not formally uppercase. Some Greek characters with *capital* in their names are actually titlecase. (Note that while the archaic Georgian script contained upper- and lowercase pairs,

they are not used in modern Georgian. See *Section 7.5, Georgian*.) The only reliable source for case information is the Unicode Character Database.

Case Mapping

The Unicode Standard normative default case mapping tables are in the Unicode Character Database. Case mapping can be an unexpectedly tricky process. For more information on case mappings, see *Section 5.18, Case Mappings*.

The Unicode Character Database contains four files with case mapping information, as shown in *Table 4-1*.

Table 4-1. Sources for Case Mapping Information

File Name	Description
UnicodeData.txt	Contains the case mappings that map to a single character. These do not increase the length of strings, nor do they contain context-dependent mappings. *Only legacy implementations that cannot handle case mappings that increase string lengths should use UnicodeData case mappings alone. The single-character mappings are insufficient for languages such as German.*
SpecialCasing.txt	Contains additional case mappings that map to more than one character, such as "ß" to "SS". It also contains context-dependent mappings, with flags to distinguish them from the normal mappings. Some characters have a "best" single-character mapping in UnicodeData as well as a full mapping in SpecialCasing.txt.
CaseFolding.txt	Contains data for performing locale-independent case folding, as described in "Caseless Matching," in *Section 5.18, Case Mappings*.
DerivedCoreProperties.txt	Contains definitions of the properties Lowercase and Uppercase.

A set of charts that show the latest case mappings is also available on the Unicode Web site. See "Charts" in *Section B.4, Other Unicode References*.

The full case mappings for Unicode characters are obtained by using the mappings from SpecialCasing.txt *plus* the mappings from UnicodeData.txt, excluding any of the latter mappings that would conflict. Any character that does not have a mapping in these files is considered to map to itself.

When used in case operations, these mappings may depend on the context surrounding each character in the original string. There are few mappings that require that context, but they are required for correct operation. Because there are few context-dependent case mappings, implementations may choose to hard-code the treatment of these characters rather than use data-driven code based on the UCD. When this is done, every time the implementation is upgraded to a new version of the Unicode Standard, the code must be checked for consistency with the updated data.

4.3 Combining Classes—Normative

Each combining character has a normative canonical *combining class*. This class is used with the canonical ordering algorithm to determine which combining characters interact typographically and to determine how the canonical ordering of sequences of combining characters takes place. Class *zero* combining characters act like base letters for the purpose of determining canonical order. Combining characters with non-zero classes participate in

reordering for the purpose of determining the canonical form of sequences of characters. (See *Section 3.11, Canonical Ordering Behavior*, for a description of the algorithm.)

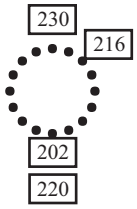
The list of combining characters and their canonical combining class appears in the Unicode Character Database. Most combining characters are nonspacing. The spacing, class zero, combining characters are so noted.

The canonical order of character sequences does *not* imply any kind of linguistic correctness or linguistic preference for ordering of combining marks in sequences. See the information on rendering combining marks in *Section 5.13, Rendering Nonspacing Marks*, for more information.

Class zero combining marks are never reordered by the canonical ordering algorithm. Except for class zero, the exact numerical values of the combining classes are of no importance in canonical equivalence, although the relative magnitude of the classes is significant. For example, it is crucial that the combining class of the cedilla be lower than the combining class of the dot below, although their exact values of 202 and 220 are not important for implementations.

Certain classes tend to correspond with particular rendering positions relative to the base character, as shown in *Figure 4-1*.

Figure 4-1. Positions of Common Combining Marks



4.4 Directionality—Normative

Directional behavior is interpreted according to the Unicode bidirectional algorithm (see Unicode Standard Annex #9, "The Bidirectional Algorithm"). For this purpose, all characters of the Unicode Standard possess a normative *directional* type. The directional types left-to-right and right-to-left are called *strong types*, and characters of these types are called strong directional characters. Left-to-right types include most alphabetic and syllabic characters, as well as all Han ideographic characters. Right-to-left types include Arabic, Hebrew, Syriac, and Thaana, and most punctuation specific to those scripts. In addition, the Unicode bidirectional algorithm uses *weak types* and *neutrals*. Interpretation of directional properties according to the Unicode bidirectional algorithm is needed for layout of right-to-left scripts such as Arabic and Hebrew.

For the directional types of Unicode characters, see the Unicode Character Database.

4.5 General Category—Normative

The Unicode Character Database defines a *General Category* for all Unicode code points. This General Category constitutes a partition of the code points into several major classes, such as letters, punctuation, and symbols, and further subclasses for each of the major classes.

Each Unicode code point is assigned a General Category value. Each value of the General Category is defined as a two-letter abbreviation, where the first letter gives information about a major class and the second letter designates a subclass of that major class. In each class, the subclass "other" merely collects the remaining characters of the major class. For example, the subclass "No" (Number, other) includes all characters of the Number class that are not a decimal digit or letter. These characters may have little in common besides their membership in the same major class.

Table 4-2 enumerates the General Category values, giving a short description of each value. See *Table 2-2* for the relationship between General Category values and basic types of code points.

Table 4-2. General Category

Lu = Letter, uppercase
Ll = Letter, lowercase
Lt = Letter, titlecase
Lm = Letter, modifier
Lo = Letter, other

Mn = Mark, nonspacing
Mc = Mark, spacing combining
Me = Mark, enclosing

Nd = Number, decimal digit
Nl = Number, letter
No = Number, other

Zs = Separator, space
Zl = Separator, line
Zp = Separator, paragraph

Cc = Other, control
Cf = Other, format
Cs = Other, surrogate
Co = Other, private use
Cn = Other, not assigned (including noncharacters)

Pc = Punctuation, connector
Pd = Punctuation, dash
Ps = Punctuation, open
Pe = Punctuation, close
Pi = Punctuation, initial quote (may behave like Ps or Pe depending on usage)
Pf = Punctuation, final quote (may behave like Ps or Pe depending on usage)
Po = Punctuation, other

Sm = Symbol, math
Sc = Symbol, currency
Sk = Symbol, modifier
So = Symbol, other

A common use of the General Category of a Unicode character is to assist in determination of boundaries in text, as in Unicode Standard Annex #29, "Text Boundaries." Other common uses include determining language identifiers for programming, scripting, and markup, as in *Section 5.15, Identifiers*, and in regular expression languages such as Perl. For more information, see Unicode Technical Report #18, "Unicode Regular Expression Guidelines."

This property is also used to support common APIs such as `isDigit()`. Common functions such as `isLetter()` and `isUppercase()` do not extend well to the larger and more complex repertoire of Unicode. While it is possible to naively extend these functions to

Unicode using the General Category and other properties, they will not work for the entire range of Unicode characters and range of tasks for which people use them. For more appropriate approaches see *Section 5.15, Identifiers*; Unicode Standard Annex #29, "Text Boundaries"; *Section 5.18, Case Mappings*; and *Section 4.9, Letters, Alphabetic, and Ideographic*.

4.6 Numeric Value—Normative

Numeric value is a normative property of characters that represent *numbers*. This group includes characters such as fractions, subscripts, superscripts, Roman numerals, currency numerators, encircled numbers, and script-specific digits. In many traditional numbering systems, letters are used with a numeric value. Examples include Greek and Hebrew letters as well as Latin letters used in outlines (II.A.1.b). These special cases are not included here as numbers.

Decimal digits form a large subcategory of numbers consisting of those digits that can be used to form decimal-radix numbers. They include script-specific digits, not characters such as Roman numerals (<1, 5> = 15 = fifteen, but <I, V> = IV = four), subscripts, or superscripts. Numbers other than decimal digits can be used in numerical expressions, but it is up to the user to determine the specialized uses.

The Unicode Standard assigns distinct codes to the forms of digits that are specific to a given script. Examples are the digits used with the Arabic script, Chinese numbers, or those of the Indic languages. For naming conventions relevant to Arabic digits, see the introduction to *Section 8.2, Arabic*.

The Unicode Character Database gives the numeric values of Unicode characters that can represent numbers.

Ideographic Numeric Values

CJK ideographs also may have numeric values. The primary numeric ideographs are shown in *Table 4-3*. When used to represent numbers in decimal notation, zero is represented by U+3007. Otherwise, zero is represented by U+96F6.

Table 4-3. Primary Numeric Ideographs

U+96F6	0
U+4E00	1
U+4E8C	2
U+4E09	3
U+56DB	4
U+4E94	5
U+516D	6
U+4E03	7
U+516B	8
U+4E5D	9
U+5341	10
U+767E	100
U+5343	1,000
U+4E07	10,000
U+5104	100,000,000 (10,000 × 10,000)
U+4EBF	100,000,000 (10,000 × 10,000)
U+5146	1,000,000,000,000 (10,000 × 10,000 × 10,000)

Ideographic accounting numbers are commonly used on checks and other financial instruments to minimize the possibilities of misinterpretation or fraud in the representation of numerical values. The set of accounting numbers varies somewhat between Japanese, Chi-

nese, and Korean usage. *Table 4-4* gives a fairly complete listing of the known accounting characters. Some of these characters are ideographs with other meanings pressed into service as accounting numbers; others are used only as accounting numbers.

Table 4-4. Ideographs Used as Accounting Numbers

1	U+58F9, U+58F1, U+5F0C[a]
2	U+8CAE,[a] U+8D30,[a] U+5F10,[a] U+5F0D[a]
3	U+53C3, U+53C2, U+53C1,[a] U+5F0E[a]
4	U+8086
5	U+4F0D
6	U+9678, U+9646
7	U+67D2[b]
8	U+634C
9	U+7396
10	U+62FE
100	U+4F70,[a] U+964C
1,000	U+4EDF
10,000	U+842C

a. These characters are used *only* as accounting numbers; they have no other meaning.
b. In Japan, U+67D2 is also pronounced *urusi*, meaning "lacquer," and is treated as a variant of the standard character for "lacquer," U+6F06.

The Unicode Character Database gives the most up-to-date and complete listing of primary numeric ideographs and ideographs used as accounting numbers, including those for CJK repertoire extensions beyond the Unified Repertoire and Ordering.

4.7 Bidi Mirrored—Normative

Bidi Mirrored is a normative property of characters such as parentheses, whose images are mirrored horizontally in text that is laid out from right to left. For example, U+0028 LEFT PARENTHESIS is interpreted as *opening parenthesis*; in a left-to-right context it will appear as "(", while in a right-to-left context it will appear as the mirrored glyph ")". The list of mirrored characters appears in the Unicode Character Database. Note that mirroring is not limited to paired characters, but that any character with the mirrored property will need two mirrored glyphs—for example, U+222B INTEGRAL. This requirement is necessary to render the character properly in a bidirectional context. It is the default behavior in Unicode text. (For more information, see the "Semantics of Paired Punctuation" subsection in *Section 6.2, General Punctuation*.)

This property is not to be confused with the related *Bidi Mirroring Glyph* property, an informative property, which can assist in rendering mirrored characters in a right-to-left context. For more information, see BidiMirroring.txt in the Unicode Character Database.

4.8 Unicode 1.0 Names

The *Unicode 1.0 character name* is an informative property of the characters defined in Version 1.0 of the Unicode Standard. The names of Unicode characters were changed in the process of merging the standard with ISO/IEC 10646. The Version 1.0 character names can be obtained from the Unicode Character Database. Where the Version 1.0 character name provides additional useful information, it is listed in *Chapter 16, Code Charts*. For example, U+00B6 PILCROW SIGN has its Version 1.0 name, PARAGRAPH SIGN, listed for clarity.

The status of the Version 1.0 character names in the case of control codes differs from that for other characters. *The Unicode Standard, Version 1.0,* gave names to the C0 control codes, U+0000..U+001F, U+007F, based on then-current practice for reference to ASCII control codes. Unicode 1.0 gave no names to the C1 control codes, U+0080..U+009F. Currently, however, the Unicode 1.0 character name property defined in the Unicode Character Database has been updated for the control codes to reflect the ISO/IEC 6429 standard names for control functions. Those names can be seen as annotations in *Chapter 16, Code Charts*. In a few instances, because of updates to ISO/IEC 6429, those names may differ from the names that actually occurred in Unicode 1.0. For example, the Unicode 1.0 name of U+0009 was HORIZONTAL TABULATION, but the ISO/IEC 6429 name for this function is CHARACTER TABULATION, and the commonly used alias is, of course, merely *tab*.

4.9 Letters, Alphabetic, and Ideographic

The concept of letters is used in many contexts. Computer language standards often characterize identifiers as consisting of letters, syllables, ideographs, and digits, but do not specify exactly what a "letter," "syllable," "ideograph," or "digit" is, leaving the definitions implicitly either to a character encoding standard or to a locale specification. The large scope of the Unicode Standard means that it includes many writing systems for which these distinctions are not as self-evident as they may once have been for systems designed to work primarily for Western European languages and Japanese. In particular, while the Unicode Standard includes various "alphabets" and "syllabaries," it also includes writing systems that fall somewhere in between. As a result, no attempt is made to draw a sharp property distinction between letters and syllables.

Alphabetic. The alphabetic property is an informative property of the primary units of alphabets and/or syllabaries, whether combining or noncombining. Included in this group would be composite characters that are canonical equivalents to a combining character sequence of an alphabetic base character plus one or more combining characters; letter digraphs; contextual variants of alphabetic characters; ligatures of alphabetic characters; contextual variants of ligatures; modifier letters; letterlike symbols that are compatibility equivalents of single alphabetic letters; and miscellaneous letter elements. Notably, U+00AA FEMININE ORDINAL INDICATOR and U+00BA MASCULINE ORDINAL INDICATOR are simply abbreviatory forms involving a Latin letter and should be considered alphabetic rather than nonalphabetic symbols.

Ideographic. The ideographic property is an informative property defined in the Unicode Character Database. The ideographic property is used, for example, in determining line breaking behavior. Characters with the ideographic property include Unified CJK Ideographs and characters from other blocks—for example, U+3007 IDEOGRAPHIC NUMBER ZERO and U+3006 IDEOGRAPHIC CLOSING MARK. For more information about Han ideographs, see *Section 11.1, Han*. For more about ideographs and logosyllabaries in general, see *Section 6.1, Writing Systems*.

4.10 Boundary Control

A number of Unicode characters have special behavior in the context of determining text boundaries. For more information about text boundaries and these characters, see Unicode Standard Annex #14, "Line Breaking Properties," and Unicode Standard Annex #29, "Text Boundaries."

4.11 Characters with Unusual Properties

The behavior of most characters does not require special attention in this standard. However, the characters in *Table 4-5* exhibit special behavior. There are many other characters which behave in special ways but which are not noted here, either because they do not affect surrounding text in the same way, or because their use is intended for well-defined contexts. Examples include the compatibility characters for block drawing, the symbol pieces for large mathematical operators, and many punctuation symbols that need special handling in certain circumstances. Such characters are more fully described in the following chapters.

Table 4-5. Unusual Properties

Function	Description	Code Point and Name
Fraction formatting	*Section 6.2*	2044 FRACTION SLASH
Special behavior with nonspacing marks	*Section 6.2* and *Section 15.2*	0020 SPACE 00A0 NO-BREAK SPACE
Double nonspacing marks	*Section 7.7*	035D COMBINING DOUBLE BREVE 035E COMBINING DOUBLE MACRON 035F COMBINING DOUBLE LOW LINE 0360 COMBINING DOUBLE TILDE 0361 COMBINING DOUBLE INVERTED BREVE 0362 COMBINING DOUBLE RIGHTWARDS ARROW BELOW
Combining half marks	*Section 7.7*	FE20 COMBINING LIGATURE LEFT HALF FE21 COMBINING LIGATURE RIGHT HALF FE22 COMBINING DOUBLE TILDE LEFT HALF FE23 COMBINING DOUBLE TILDE RIGHT HALF
Cursive joining and ligation control	*Section 15.2*	200C ZERO WIDTH NON-JOINER 200D ZERO WIDTH JOINER
Grapheme joining	*Section 15.2*	034F COMBINING GRAPHEME JOINER
Bidirectional ordering	*Section 15.2*	200E LEFT-TO-RIGHT MARK 200F RIGHT-TO-LEFT MARK 202A LEFT-TO-RIGHT EMBEDDING 202B RIGHT-TO-LEFT EMBEDDING 202C POP DIRECTIONAL FORMATTING 202D LEFT-TO-RIGHT OVERRIDE 202E RIGHT-TO-LEFT OVERRIDE
Mathematical expression formatting	*Section 15.3*	2061 FUNCTION APPLICATION 2062 INVISIBLE TIMES 2063 INVISIBLE SEPARATOR
Deprecated alternate formatting	*Section 15.4*	206A INHIBIT SYMMETRIC SWAPPING 206B ACTIVATE SYMMETRIC SWAPPING 206C INHIBIT ARABIC FORM SHAPING 206D ACTIVATE ARABIC FORM SHAPING 206E NATIONAL DIGIT SHAPES 206F NOMINAL DIGIT SHAPES
Prefixed format control	*Section 8.2* and *Section 8.3*	0600 ARABIC NUMBER SIGN 0601 ARABIC SIGN SANAH 0602 ARABIC FOOTNOTE MARKER 0603 ARABIC SIGN SAFHA 06DD ARABIC END OF AYAH 070F SYRIAC ABBREVIATION MARK

Table 4-5. Unusual Properties (Continued)

Function	Description	Code Point and Name
Brahmi-derived script dead-character formation	*Chapter 9* and *Chapter 10*	094D DEVANAGARI SIGN VIRAMA 09CD BENGALI SIGN VIRAMA 0A4D GURMUKHI SIGN VIRAMA 0ACD GUJARATI SIGN VIRAMA 0B4D ORIYA SIGN VIRAMA 0BCD TAMIL SIGN VIRAMA 0C4D TELUGU SIGN VIRAMA 0CCD KANNADA SIGN VIRAMA 0D4D MALAYALAM SIGN VIRAMA 0DCA SINHALA SIGN AL-LAKUNA 0E3A THAI CHARACTER PHINTHU 1039 MYANMAR SIGN VIRAMA 1714 TAGALOG SIGN VIRAMA 1734 HANUNOO SIGN PAMUDPOD 17D2 KHMER SIGN COENG
Historical viramas with other functions	*Section 9.11* and *Section 9.12*	0F84 TIBETAN MARK HALANTA 193B LIMBU SIGN SA-I
Mongolian variation selectors	*Section 12.2*	180B MONGOLIAN FREE VARIATION SELECTOR ONE 180C MONGOLIAN FREE VARIATION SELECTOR TWO 180D MONGOLIAN FREE VARIATION SELECTOR THREE 180E MONGOLIAN VOWEL SEPARATOR
Generic variation selectors	*Section 15.6*	FE00..FE0F VARIATION SELECTOR-1..VARIATION SELECTOR-16 E0100..E01EF VARIATION SELECTOR-17..VARIATION SELECTOR-256
Tag characters	*Section 15.10*	E0001 LANGUAGE TAG E0020..E007F LANGUAGE TAG SPACE..CANCEL TAG
Ideographic variation indication	*Section 6.2*	303E IDEOGRAPHIC VARIATION INDICATOR
Ideographic description	*Section 11.1*	2FF0..2FFB IDEOGRAPHIC DESCRIPTION CHARACTER LEFT TO RIGHT..IDEOGRAPHIC DESCRIPTION CHARACTER OVERLAID
Interlinear annotation	*Section 15.9*	FFF9 INTERLINEAR ANNOTATION ANCHOR FFFA INTERLINEAR ANNOTATION SEPARATOR FFFB INTERLINEAR ANNOTATION TERMINATOR
Object replacement	*Section 15.9*	FFFC OBJECT REPLACEMENT CHARACTER
Code conversion fallback	*Section 15.9*	FFFD REPLACEMENT CHARACTER
Musical format control	*Section 14.11*	1D173 MUSICAL SYMBOL BEGIN BEAM 1D174 MUSICAL SYMBOL END BEAM 1D175 MUSICAL SYMBOL BEGIN TIE 1D176 MUSICAL SYMBOL END TIE 1D177 MUSICAL SYMBOL BEGIN SLUR 1D178 MUSICAL SYMBOL END SLUR 1D179 MUSICAL SYMBOL BEGIN PHRASE 1D17A MUSICAL SYMBOL END PHRASE
Line break controls	*Section 15.2*	00AD SOFT HYPHEN 200B ZERO WIDTH SPACE 2060 WORD JOINER
Byte order signature	*Section 15.9*	FEFF ZERO WIDTH NO-BREAK SPACE