INTERNATIONAL STANDARD

ISO/IEC 10646-1

First edition
1993-05-01

AMENDMENT 2
1996-10-15

Information technology — Universal Multiple-Octet Coded Character Set (UCS) —

Part 1:
Architecture and Basic Multilingual Plane

AMENDMENT 2: UCS Transformation Format 8 (UTF-8)

Technologies de l'information — Jeu universel de caractères codés à plusieurs octets —

Partie 1: Architecture et table multilingue

AMENDEMENT 2: Format de transformation UCS 8 (UTF-8)

Reference number
ISO/IEC 10646-1:1993/Amd.2:1996(E)

Contents

Page

Foreword iii
Introduction iv
2 Conformance 1
5 General structure of the UCS 1

Annexes

F The use of "signatures" to identify UCS 1
M External references to character repertoires 1
R UCS Transformation Format 8 (UTF-8) 2
R.1 Features of UTF-8 2
R.2 Specification of UTF-8 2
R.3 Notation 4
R.4 Mapping from UCS-4 form to UTF-8 form 4
R.5 Mapping from UTF-8 form to UCS-4 form 4
R.6 Identification of UTF-8 5
R.7 Incorrect sequences of octets: Interpretation by receiving devices 5

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Amendment 2 to International Standard ISO/IEC 10646-1:1993 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

Introduction

ISO/IEC 10646 specifies the Universal Multiple-Octet Coded Character Set (UCS). It is applicable to the representation, transmission, interchange, processing, storage, input and presentation of the written form of the languages (scripts) of the world as well as additional symbols.

This amendment to ISO/IEC 10646 specifies an additional transformation format, UTF-8. In UTF-8 all the characters of the UCS have a coded representation which is suitable for use in communications and other environments where some octet values of the code are assumed to have a fixed definition according to ISO/IEC 4873.

Information technology — Universal Multiple-Octet Coded Character Set (UCS) —

Part 1:

Architecture and Basic Multilingual Plane

AMENDMENT 2: UCS Transformation Format 8 (UTF-8)

2 Conformance

Clause 2 as amended by Amendment 1 applies with 2.2a) amended as follows. Replace:

or Annex Q,

with:

or Annex Q or Annex R,

5 General structure of the UCS

Clause 5 applies with the following new paragraph added at the end of the clause:

A UCS Transformation Format (UTF-8) is specified in Annex R which can be used to transmit text data through communication systems which are sensitive to octet values for control characters coded according to the 8-bit structure of ISO/IEC 2022, and to ISO/IEC 4873. UTF-8 also avoids the use of octet values according to ISO/IEC 4873 which have special significance during the parsing of file-name character strings in widely-used file-handling systems.

Annex F - The use of "signatures" to identify UCS

Annex F applies with the text amended as follows. After:

UCS-4 signature: 0000 FEFF

insert:

UTF-8 signature: EF BB BF

Annex M - External references to character repertoires

Annex M as amended by Amendment 1 applies with M.3 amended as follows. In the third paragraph replace:

- UTF16-form (5).

with:

- UTF16-form (5), or
- UTF8-form (8).

Replace:

- "ISO 10646 part-1 utf-16".

with:

- "ISO 10646 part-1 utf-16"
- "ISO 10646 part-1 utf-8".

Add the following new annex:

Annex R

(normative)

UCS Transformation Format 8 (UTF-8)

UTF-8 is an alternative coded representation form for all of the characters of the UCS. It can be used to transmit text data through communication systems which assume that individual octets in the range 00 to 7F have a definition according to ISO/IEC 4873, including a C0 set of control functions according to the 8-bit structure of ISO/IEC 2022. UTF-8 also avoids the use of octet values in this range which have special significance during the parsing of file-name character strings in widely-used file-handling systems.

The number of octets in the UTF-8 coded representation of the characters of the UCS ranges from one to six; the value of the first octet indicates the number of octets in that coded representation.

R.1 Features of UTF-8

- UCS characters from the BASIC LATIN collection are represented in UTF-8 in accordance with ISO/IEC 4873, i.e. single octets with values ranging from 20 to 7E.
- Control functions in positions 0000 0000 to 0000 001F, and the DELETE character in position 0000 007F, are represented without the padding octets specified in clause 16, i.e. as single octets with values ranging from 00 to 1F, and 7F respectively in accordance with ISO/IEC 4873 and with the 8-bit structure of ISO/IEC 2022.
- Octet values 00 to 7F do not otherwise occur in the UTF-8 coded representation of any character. This provides compatibility with existing file-handling systems and communications sub-systems which parse CC-data-elements for these octet values.
- The first octet in the UTF-8 coded representation of any character can be directly identified when a CC-data-element is examined, one octet at a time, starting from an arbitrary location. It indicates the number of continuing octets (if any) in the multi-octet sequence that constitutes the coded representation of that character.

R.2 Specification of UTF-8

In the UTF-8 coded representation form each character from this International Standard shall have a coded representation that comprises a sequence of octets of length 1, 2, 3, 4, 5, or 6 octets.

For all sequences of one octet the most significant bit shall be a ZERO bit.

For all sequences of more than one octet, the number of ONE bits in the first octet, starting from the most significant bit position, shall indicate the number of octets in the sequence. The next most significant bit shall be a ZERO bit.

NOTE 1 - For example, the first octet of a 2-octet sequence has bits 110 in the most significant positions, and the first octet of a 6-octet sequence has bits 1111110 in the most significant positions.

All of the octets, other than the first in a sequence, are known as continuing octets. The two most significant bits of a continuing octet shall be a ONE bit followed by a ZERO bit.

The remaining bit positions in the octets of the sequence shall be "free bit positions" that are used to distinguish between the characters of this International Standard. These free bit positions shall be used, in order of increasing significance, for the bits of the UCS-4 coded representation of the character, starting from its least significant bit. Some of the high-order ZERO bits of the UCS-4 representation shall be omitted, as specified below.

Table 1 below shows the format of the octets of a coded character according to UTF-8. Each free bit position available for distinguishing between the characters is indicated by an x. Each entry in the column "Maximum UCS-4 value" indicates the upper end of the range of coded representations from UCS-4 that may be represented in a UTF-8 sequence having the length indicated in the "Octet usage" column.

Table 1 - Format of octets in a UTF-8 sequence

Octet usage	Format (binary)	No. of free bits	Maximum UCS-4 value
1st of 1	0xxxxxxx	7	0000 007F
1st of 2	110xxxxx	5	0000 07FF
1st of 3	1110xxxx	4	0000 FFFF
1st of 4	11110xxx	3	001F FFFF
1st of 5	111110xx	2	03FF FFFF
1st of 6	1111110x	1	7FFF FFFF
continuing) 2nd .. 6th)	10xxxxxx	6	

Table 1 shows that, in a CC-data-element conforming to UTF-8, the range of values for each octet indicates its usage as follows:

00 to 7F first and only octet of a sequence;

80 to BF continuing octet of a multi-octet sequence;

C0 to FD first octet of a multi-octet sequence;

FE or FF not used.

The mapping between UCS-4 and UTF-8 shall be as shown in R.4; the reverse mapping is shown in R.5.

NOTE 2 - Examples of UCS-4 coded representations and the corresponding UTF-8 coded representations are shown in Tables 2 and 3 below.

Table 2 shows the UCS-4 and the UTF-8 coded representations, in binary notation, for a selection of code positions from the UCS.

Table 3 shows the UCS-4 and the UTF-8 coded representations, in hexadecimal notation, for the same selection of code positions from the UCS.

Table 3 - Examples in hexadecimal notation

UCS-4 form	UTF-8 form
0000 0001;	01;
0000 007F;	7F;
0000 0080;	C2; 80;
0000 07FF;	DF; BF;
0000 0800;	E0; A0; 80;
0000 FFFF;	EF; BF; BF;
0001 0000;	F0; 90; 80; 80;
0010 FFFF;	F4; 8F; BF; BF;
001F FFFF;	F7; BF; BF; BF;
0020 0000;	F8; 88; 80; 80; 80;
03FF FFFF;	FB; BF; BF; BF; BF;
0400 0000;	FC; 84; 80; 80; 80;80;
7FFF FFFF;	FD; BF; BF; BF; BF; BF;

Table 2 - Examples in binary notation

Four-octet form - UCS-4	UTF-8 form
00000000 00000000 00000000 00000001;	00000001;
00000000 00000000 00000000 01111111;	01111111;
00000000 00000000 00000000 10000000;	11000010; 10000000;
00000000 00000000 00000111 11111111;	11011111; 10111111;
00000000 00000000 00001000 00000000;	11100000; 10100000; 10000000;
00000000 00000000 11111111 11111111;	11101111; 10111111; 10111111;
00000000 00000001 00000000 00000000;	11110000; 10010000; 10000000;10000000;
00000000 00011111 11111111 11111111;	11110111; 10111111; 10111111;10111111;
00000000 00100000 00000000 00000000;	11111000; 10001000; 10000000;10000000; 10000000;
00000011 11111111 11111111 11111111;	11111011; 10111111; 10111111;10111111; 10111111;
00000100 00000000 00000000 00000000;	11111100; 10000100; 10000000;10000000; 10000000; 10000000;
01111111 11111111 11111111 11111111;	11111101; 10111111; 10111111;10111111; 10111111; 10111111;

R.3 Notation

1. All numbers are in hexadecimal notation, except for the decimal numbers used in the power-of operation (see 5 below).
2. Boundaries of code elements are indicated with semicolons; these are single-octet boundaries within UTF-8 coded representations, and four-octet boundaries within UCS-4 coded representations.
3. The symbol "%" indicates the modulo operation, e.g.: x % y = x modulo y
4. The symbol "/" indicates the integer division operation, e.g.: 7 / 3 = 2
5. Superscripting indicates the power-of operation, e.g.: $2^3 = 8$
6. Precedence is: power-of operation > integer division > modulo operation > integer multiplication > integer addition.

   e.g.: $x / y^z \% w = ((x / (y^z)) \% w)$

R.4 Mapping from UCS-4 form to UTF-8 form

Table 4 defines in mathematical notation the mapping from the UCS-4 coded representation form to the UTF-8 coded representation form.

In the left column (UCS-4) the notation x indicates the four-octet coded representation of a single character of the UCS. In the right column (UTF-8) x indicates the corresponding integer value.

NOTE 3 - Values of x in the range 0000 D800 .. 0000 DFFF are reserved for the UTF-16 form and do not occur in UCS-4. The values 0000 FFFE and 0000 FFFF also do not occur (see clause 8). The mappings of these code positions in UTF-8 are undefined.

NOTE 4 - The algorithm for converting from UCS-4 to UTF-8 can be summarised as follows.

For each coded character in UCS-4 the length of octet sequence in UTF-8 is determined by the entry in the right column of Table 1. The bits in the UCS-4 coded representation, starting from the least significant bit, are then distributed across the free bit positions in order of increasing significance until no more free bit positions are available.

Table 4 - Mapping from UCS-4 to UTF-8

Range of values in UCS-4	Sequence of octets in UTF-8
x = 0000 0000 .. 0000 007F;	x;
x = 0000 0080 .. 0000 07FF;	C0 + x / 2^6; 80 + x %2^6;
x = 0000 0800 .. 0000 FFFF; (see Note 3)	E0 + x/2^{12}; 80 + x/2^6%2^6; 80 + x%2^6;
x = 0001 0000 .. 001F FFFF;	F0 + x/2^{18}; 80 + x/2^{12}%2^6; 80 + x/2^6%2^6; 80 + x%2^6;
x = 0020 0000 .. 03FF FFFF;	F8 + x/2^{24}; 80 + x/2^{18}%2^6; 80 + x/2^{12}%2^6; 80 + x/2^6%2^6; 80 + x%2^6;
x = 0400 0000 .. 7FFF FFFF;	FC + x/2^{30}; 80 + x/2^{24}%2^6; 80 + x/2^{18}%2^6; 80 + x/2^{12}%2^6; 80 + x/2^6%2^6; 80 + x%2^6;

R.5 Mapping from UTF-8 form to UCS-4 form

Table 5 defines in mathematical notation the mapping from the UTF-8 coded representation form to the UCS-4 coded representation form.

In the left column (UTF-8) the following notations apply:

z is the first octet of a sequence. Its value determines the number of continuing octets in the sequence.

y is the 2nd octet in the sequence.

x is the 3rd octet in the sequence.

w is the 4th octet in the sequence.

v is the 5th octet in the sequence.

u is the 6th octet in the sequence.

The ranges of values applicable to these octets are shown in R.2 above, following Table 1.

NOTE 5- The algorithm for converting from UTF-8 to UCS-4 can be summarised as follows.

For each coded character in UTF-8 the bits in the free bit positions are concatenated as a bit-string. The bits from this string, in increasing order of significance, are then distributed across the bit positions of a four-octet sequence, starting from the least significant bit position. The remaining bit positions of that sequence are filled with ZERO bits.

Table 5 - Mapping from UTF-8 to UCS-4

Sequence of octets in UTF-8	Four-octet sequences in UCS-4
z = 00 .. 7F;	z;
z = C0 .. DF; y;	(z-C0)*2^6 + (y-80);
z = E0 .. EF; y; x;	(z-E0)*2^{12} + (y-80)*2^6 + (x-80);
z = F0 .. F7; y; x; w;	(z-F0)*2^{18} + (y-80)*2^{12} +(x-80)*2^6 + (w-80);
z = F8 ..FB; y; x; w; v;	(z-F8)*2^{24} + (y-80)*2^{18} +(x-80)*2^{12} + (w-80)*2^6 + (v - 80);
z = FC, FD; y; x; w; v; u;	(z-FC)*2^{30} + (y-80)*2^{24} +(x-80)*2^{18} + (w-80)*2^{12} + (v-80)*2^6 + (u-80);

R.6 Identification of UTF-8

When the escape sequences from ISO/IEC 2022 are used, the identification of UTF-8 and an implementation level (see clause 15) shall be by a designation sequence chosen from the following list:

ESC 02/05 02/15 04/07

UTF-8 with implementation level 1

ESC 02/05 02/15 04/08

UTF-8 with implementation level 2

ESC 02/05 02/15 04/09

UTF-8 with implementation level 3

If such an escape sequence appears within a CC-data-element conforming to ISO/IEC 2022, it shall consist only of the sequences of bit combinations as shown above.

If such an escape sequence appears within a CC-data-element conforming to ISO/IEC 10646, it shall be padded in accordance with clause 16.

When the escape sequences from ISO/IEC 2022 are used, the identification of a return, or transfer, from UTF-8 to the coding system of ISO/IEC 2022 shall be as specified in 17.5 for a return or transfer from UCS.

NOTE 6 - The following escape sequence may also be used:

ESC 02/05 04/07 UTF-8.

The implementation level is not defined. The escape sequence used for a return to the coding system of ISO/IEC 2022 is not padded as specified in 17.5.

R.7 Incorrect sequences of octets: Interpretation by receiving devices

According to R.2 an octet in the range 00 .. 7F or C0 .. FB is the first octet of a UTF-8 sequence, and is followed by the appropriate number (from 0 to 5) of continuing octets in the range 80 .. BF. Furthermore, octets whose value is FE or FF are not used; thus they are invalid in UTF-8.

If a CC-data-element includes either:

- a first octet that is not immediately followed by the correct number of continuing octets, or
- one or more continuing octets that are not required to complete a sequence of first and continuing octets, or
- an invalid octet,

then according to R.2 such a sequence of octets is not in conformance with the requirements of UTF-8. It is known as a malformed sequence.

If a receiving device that has adopted the UTF-8 form receives a malformed sequence, because of error conditions either:

- in an originating device, or
- in the interchange between an originating and a receiving device, or
- in the receiving device itself,

then it shall interpret that malformed sequence in the same way that it interprets a character that is outside the adopted subset that has been identified for the device (see 2.3c).