UTF-7

A Mail-Safe Transformation Format of ISO/IEC 10646-1/Unicode

Mark Davis and David Goldsmith
mark_davis@taligent.com
david_goldsmith@taligent.com

Status of this Memo

This document is a preliminary proposal, intended to be eventually submitted as either an Internet standard or an ISO/IEC 10646-1 annex. This draft is for discussion purposes only.

Abstract

ISO/IEC 10646-1:1993(E) and the Unicode Standard, version 1.1, jointly define a 16 bit character set (hereafter referred to as BMP, the Basic Multilingual Plane of 10646) which encompasses most of the world's writing systems. However, Internet mail (STD 11, RFC 822) currently supports only 7-bit US ASCII as a character set. MIME (RFC 1521 and RFC 1522) extends Internet mail to support different media types and character sets, and thus could support BMP in mail messages. As defined, however, MIME would only support encoding of BMP in a way unintelligible to recipients who do not have MIME or BMP support on their system. Indeed, the current version of RFC 1521 does not list BMP as a supported character set at all.

This document proposes a new transformation format of BMP that contains only 7-bit ASCII characters and is intended to be readable by humans in the limiting case that the document consists of characters from the US-ASCII repertoire. See the companion document, "Encoding of ISO/IEC 10646-1/Unicode in MIME" for details on how this transformation format would be used in the context of RFC 1521 and RFC 1522.

Motivation

Although other transformation formats of BMP exist and could conceivably be used in this context (most notably UTF-1 and UTF-FSS), they suffer the disadvantage that they use octets in the range decimal 128 through 255 to encode BMP characters outside the US-ASCII range. Thus, in the context of mail, those octets must themselves be encoded. This requires putting text through two successive encoding processes, and leads to a significant expansion of characters outside the US-ASCII range, putting non-English speakers at a disadvantage. For example, using UTF-FSS together with the Quoted-Printable content transfer encoding of MIME represents US-ASCII characters in one octet, but other characters may require up to nine octets. See the companion document "Encoding of ISO/IEC 10646-1/Unicode in MIME" for a discussion of alternatives to UTF-7.

Overview

UTF-7 encodes BMP characters as US-ASCII, together with shift sequences to encode characters outside that range. For this purpose, a few of the characters in the US-ASCII repertoire are reserved for use as shift characters.

Many mail gateways and systems cannot handle the entire US-ASCII character set (those based on EBCDIC, for example), and so UTF-7 contains provisions for encoding characters within US-ASCII in a way that all mail systems can accomodate.

> **Note**. In some ways, UTF-7 duplicates some of the functionality of MIME's Quoted-Printable content transfer encoding. UTF-7 already supported most of the functions of Quoted-Printable, and adding the remaining few functions allows text to be prepared for mailing by passing through one rather than two filters.

Definitions

First, the definition of BMP:

> The 16 bit character set BMP is defined by the international standard ISO/IEC 10646-1:1993(E); Coded Representation Form=UCS-2; Subset=300; Implementation Level=3. This character set is identical with the character repertoire and coding of The Unicode Standard, Version 1.1.
>
> **Note**. Unicode 1.1 further specifies the use and interaction of these character codes beyond the ISO standard. However, any valid BMP sequence is a valid Unicode sequence; Unicode supplies interpretations of sequences on which the ISO standard is silent as to interpretation.

Next, some handy definitions of US-ASCII character subsets:

> Set D (directly encoded characters) consists of the following characters (derived from RFC 1521, Appendix B): the upper and lower case letters A through Z and a through z, the 10 digits 0-9, and the following nine special characters (note that + and = are omitted):

Character	ASCII & BMP Value (decimal)
'	39
(	40
)	41
,	44
-	45
.	46
/	47
:	58
?	63

Set O (optional direct characters) consists of the following characters:

Character	ASCII & BMP Value (decimal)
!	33
"	34
#	35
$	36
%	37
&	38
*	42
;	59
<	60
=	61
>	62
@	64
[	91
\	92
]	93
^	94
_	95
`	96
{	123
|	124
}	125
~	126

Set B (Modified Base 64) is the set of characters in the Base64 alphabet defined in RFC 1521, excluding the pad character = (decimal value 61).

Rationale. The pad character = is excluded because UTF-7 is designed for use within header fields as set forth in RFC 1522. Since the only readable encoding in RFC 1522 is “Q” (based on RFC 1521’s Quoted-Printable), the = character is not available for use (without a lot of escape sequences). This was very unfortunate but unavoidable.The = could otherwise have been used as the escape character as well (rather than using +).

Note that all characters in US-ASCII have the same value in BMP when zero-extended to 16 bits.

UTF-7 Encoding

A UTF-7 stream represents 16-bit BMP characters in 7-bit US-ASCII as follows:

Rule 1: (direct encoding) BMP characters in set D above may be encoded directly as their ASCII equivalents. BMP characters in Set O may optionally be encoded directly as their ASCII equivalents, bearing in mind that many of these characters are illegal in header fields, or may not pass correctly through some mail gateways.

Rule 2: (BMP shifted encoding) Any BMP character sequence may be encoded using a sequence of characters in set B, when preceded by the shift character + (US-ASCII character value decimal 43). The + signals that subsequent octets are to be interpreted as elements of the Modified Base64 alphabet until a character not in that alphabet is encountered. Such characters include control characters such as carriage returns and line feeds; thus, a BMP shifted sequence always terminates at the end of a line. As a special case, if the sequence terminates with the character - then that character is absorbed; other terminating characters are not absorbed and are processed normally. Also as a special case, the sequence “+-” may be used to encode the character “+”.

Rationale. A terminating character is necessary for cases where the next character after the Modified Base64 sequence is part of character set B.

BMP is encoded using Modified Base64 by first converting BMP 16-bit quantities to an octet stream (with the most significant octet first).

Rationale. ISO/IEC 10646-1:1993(E) specifies that when characters in the UCS-2 form are serialized as octets, that the most significant octet appear first. This is also in keeping with common network practice of choosing a canonical format for transmission.

Next, the octet stream is encoded by applying the Base64 content transfer encoding algorithm as defined in RFC 1521, modified to omit the = pad character. Instead, when encoding, zero bits are added to pad to a Base64 character boundary. When decoding, any bits at the end of the Modified Base64 sequence that do not constitute a complete 16-bit BMP character are discarded.

Rationale. The pad character = is not used when encoding Modified Base64 because of the conflict with its use as an escape character for the Q content transfer encoding in RFC 1522 header fields, as mentioned above.

Rule 3: (White Space, equivalent to Rule 3 of RFC 1521 Quoted-Printable, except that + indicates a soft line break rather than =) [Will be copied rather than referenced when this is polished up]

Rule 4: (Line Breaks, equivalent to Rule 4 of RFC 1521 Quoted-Printable) [Will be copied rather than referenced when this is polished up]

Rule 5: (Soft Line Breaks, equivalent to Rule 5 of RFC 1521 Quoted-Printable, except that + indicates a soft line break rather than =) [Will be copied rather than referenced when this is polished up]

Given this set of rules, BMP characters which may be encoded via rule 1 take one octet per character, and other BMP characters are encoded on average with 2 2/3 octets per character plus one octet to switch into Modified Base64 and an optional octet to switch out.

Example. The BMP sequence “A<NOT IDENTICAL TO><ALPHA>.” (hexadecimal 0041,2262,0391,002E) is encoded as follows:

```
A+ImIDkQ.
```

Example. The BMP sequence “Hi Mom <WHITE SMILING FACE>!” (hexadecimal 0048,0069,0020,004D,006F,004D,0020,263A,0021) is encoded as follows:

```
Hi Mom +Jjo!
```

Example. The BMP sequence representing the Han characters for the Japanese word “nihongo” (hexadecimal 65E5,672C,8A9E) is encoded as follows:

```
+ZeVnLIqe-
```

Summary

The UTF-7 encoding allows BMP characters to be encoded within the US-ASCII 7 bit character set. It is most effective for BMP sequences which contain relatively long strings of US-ASCII characters interspersed with either single BMP characters or strings of BMP characters, as it allows the US-ASCII portions to be read on systems without direct BMP support.

[more later]

References

To be added later.

ISO/IEC 10646-1:1993(E); Unicode v1, v2, 1.1 TR; RFC 822, 1521, 1522; MIME & Unicode; UTF-2 (X/Open)

Acknowledgements

Many thanks to the following people for their helpful comments and suggestions:

Nathaniel Borenstein, Lee Collins, John Jenkins.

[more later]