Internet Draft
<draft-jseng-utf5-00.txt>
27th July 1999
Expires End of January 2000

James Seng, BIX
Martin Duerst, W3C
Tin Wee Tan, NUS

UTF-5, a transformation format of Unicode and ISO 10646

Status of this Memo

Abstract

A new transformation format, called UTF-5 for Unicode is proposed. The resulting string of this UTF is within a [A-V][0-9] alphanumeric range. This enables legacy systems or protocols designed for alphanumerical character set only to be multilingual enabled and internationalized immediately. Example of such systems are the domain name system and email addresses.

1. Introduction

The Unicode Standard, version 2.1 [UNICODE], and ISO/IEC 10646-1 [ISO-10646] jointly define a 16 bit character set, UCS-2, which encompasses most of the world's writing systems. ISO 10646 further defines a 31-bit character set, UCS-4, with currently no assignments outside of the region corresponding to UCS-2 (the Basic Multilingual Plane, BMP). The UCS-2 and UCS-4 encodings, however, are hard to use in many current applications and protocols that assume 8 or even 7 bit characters. Even newer systems able to deal with 16 bit characters cannot process UCS-4 data. This situation has led to the development of so-called UCS transformation formats (UTF), each with different characteristics.

At this moment, there are 3 standard UTF, namely UTF-7 [UTF7], UTF-8 [UTF8] and UTF-16 [UTF16], each is a variable length transformation which gives 7 bit, 8 bit and 16 bit strings respectively. While these are sufficient for most application uses, there are however some legacy systems which are, unfortunately, unable to handle even 7 bit strings either due to technical restriction or common uses.

The object of this memo is to propose a UTF-5 which gives a transformed string that is within [A-V][0-9] alphanumerical character set. This enables legacy system designed for alphanumerical character set only to be multilingual enabled and internationalized immediately.

UTF-8 is the transformation format for all IETF standards [IETFPC]. UTF-5 is not here to change this. It is proposed to support legacy applications or protocols that cannot be modify in a simple way to handle 8 bits using UTF-8 encoding. See Section 4 on the discussion on how UTF-5 can be used for Domain Name System [DNS] and Simple Mail Transfer Protocol [SMTP] Address.

2. UTF-5 definition

In UTF-5, each character are encoded using a sequence of 1 to 8 octets. Two transformations are needed for UTF-5, namely

1. Determine the quintet ("5-bit") binary sequence.
2. From a table, translate the quintet to the resulting string.

Take note that the UTF-5 is not a sequence of quintets but a sequence of octets where each octets are in the alphanumeric range. Alphanumeric is defined as A to V (uppercase only) and 0 to 9 in this context.

This memo does not specify the binary pattern of the alphanumeric characters as the purpose of the transformation is to get a alphanumeric string which represent a multilingual string. However, it is presumed that US-ASCII [US-ASCII] is use for most purposes.

2.1 Determine the quintet binary sequence

The first quintet of a binary sequence will have the highest-order bit set to 1 and the remaining quintet will have the highest-order bit set to 0. The remaining 4 bits of every quintet contain bits from the value of the character to be encoding.

The table below summarizes the format of these different quintet types. The letter x indictes bits available for encoding bits of the UCS-4 character value.

UCS-4 range (hex.)	UTF-5 quintet sequence (binary)
0000 0000-0000 000F	1xxxx
0000 0010-0000 00FF	1xxxx 0xxxx
0000 0100-0000 0FFF	1xxxx 0xxxx 0xxxx
0000 1000-0000 FFFF	1xxxx 0xxxx 0xxxx 0xxxx
...	
1000 0000-7FFF FFFF	1xxxx 0xxxx 0xxxx 0xxxx

2.2 Translation table for quintet and alphanumeric character

Translation table for quintet binary pattern and alphanumeric character are as follows:

quintet		quintet		quintet		quintet	
00000	0	01000	8	10000	G	11000	O
00001	1	01001	9	10001	H	11001	P
00010	2	01010	A	10010	I	11010	Q
00011	3	01011	B	10011	J	11011	R
00100	4	01100	C	10100	K	11100	S
00101	5	01101	D	10101	L	11101	T
00110	6	01110	E	10110	M	11110	U
00111	7	01111	F	10111	N	11111	V

2.3 Encoding from UCS-4 to UTF-5

1) Determine the required number of octets from the character value. Let U be the UCS-4 value, then the required number of octets is log16(U) round up.

2) Prepare the quintet binary sequence. Put the highest order bit of the first quintet as 1 and highest order bit of the rest of the quintet as 0.

3) Fill in the bits marked x from the bits of the character value, starting from the lower-order its of the character value and putting them first in the last quintet of the sequence, then the

next to last, etc until all x bits are filled in.

4) For each quintet, apply the lookup table in Section 2.2 to get the corresponding alphanumeric character.

2.4 Decoding UTF-5 to UCS-4

1) Determine the length of octet sequence. As according to the UTF-5 encoding, every character will have the inital octet within 'G' to 'V'. Thus, the length of the octet sequence can be determined by looking for 'G' to 'V' in the UTF-5 string.

2) Apply the reverse lookup according to the table in Section 2.2 to get the quintet binary sequence.

3) Initialize the 4 octets of the UCS-4 character with all bits set to 0.

4) Distribute the bits from the sequence to the UCS-4 character, first the lower-order bits from the last octet of the sequence and proceeding to the left until no x bits are left.

If the UTF-5 sequence is no more than four octets long, decoding can proceed directly to UCS-2 (or equivalently Unicode).

2.5 Detecting UTF-5 string

As the UTF-5 string is a alphanumeric string, it is difficult to differential between a normal ASCII document or a UTF-5 document.

Nevertheless, if the string is sufficient long, it is possible to do some detection of UTF-5 string base on the fact that
1. UTF-5 strings only have characters within '0'-'9' and 'A'-'V'.
2. UTF-5 strings have a well-defined inital octet of 'G' to 'V'.
3. The 'G' character always occurs as the inital and only octet.

3. Examples of UTF-5

The Unicode sequence "A<NOT IDENTICAL TO><ALPHA>." (0041, 2262, 0391, 002E) may be encoded as follows:

"K1I262J91IE"

The Unicode sequence "Hi Mom <WHITE SMILING FACE>!" (0048, 0069, 0020, 004D, 006F, 006D, 0020, 263A, 0021) may be encoded as follows:

"K8M9I0KDMFMDI0I63AI1"

The Unicode sequence representing the Han characters for the Japanese word "nihongo" (65E5, 672C, 8A9E) may be encoded as follows:

"M5E5M72COA9E"

Note that from the examples, it is obvious that there is a short-cut to the UTF-5 transformation which goes like this:

1. Write down the hexdecimal of the Unicode character as a string.
2. For the first character of the hexdecimal string, change 0 to G, 1 to H, 2 to I, ... F to V.

This will yield you the UTF-5 string of the Unicode character.

4. Applications

There are many applications whereby UTF-5 would be useful for Internationalization ("i18n"). Here are some of the possible uses.

a. Internationalization of Domain Names System

In the Domain Name System, although the technical standard does not prevent 8-bits character to be use as domain names, general use of the system restrict it to only A-Z (upper and lower), 0-9 and "-" as a valid domain name. This pose some great difficulty when doing i18n of domain names as the current UTF-7, UTF-8 and UTF-16 is not compatible with the existing software system already in used.

Please see draft-xxx-xxx-xxx.txt for detail discussion on Internationalization of Domain Names System ("iDNS").
http://www.idns.org/

b. Internationalization of Simple Mail Transfer Protocol Address

While it is possible for a person to send SMTP Mail in different language on different character set to each another using Multi-purpose Internet Mail Extensions [MIME], the SMTP Mail Address remains a challenge to be Internationalized. Internationalization of SMTP Address has two barrier, 1. the Internationalization of Domain Name System and 2. the Internationalization of the mailbox or username. SMTP mailbox have a very strict check [RFC822] dues to many potential security risks when using symbols or special characters in mailbox. UTF-5 will allow Unicode to be used immediately as mailbox with minimual change in system and without additional security risks.

Please see draft-xxx-xxx-xxx.txt for detail discussion on Internationalization of Simple Mail Transfer Protocol Address ("iMail").

Internationalization of URIs is not discussed in this memo. Please refer to http://www.w3.org/International/0-URL-and-ident.html.

However, uses for UTF-5 goes beyond Internet back to old legacy system such as Telegram system or even Morse code allowing Multilingual characters to be transmitted.

5. Security Considerations

This memo does not address any security consideration at the moment.

6. Acknowledgments

UTF-5 was first defined by Martin Duerst at the University of Zurich in draft-duerst-dns-i18n-00.txt.

Contributors (not in any order):
Marc Blanchet <Marc.Blanchet@viagenic.qc.ca>

7. Bibliography

[ISO-10646]   ISO/IEC 10646-1:1993. International Standard --
[UTF16]   Information technology -- Universal Multiple-Octet Coded Character Set (UCS) -- Part 1: Architecture and Basic Multilingual Plane. UTF-8 is described in Annex R, adopted but not yet published. UTF-16 is described in Annex Q, adopted but not yet published.

[UTF7]   Goldsmith, D., and M. Davis, "UTF-7: A Mail-safe Transformation Format of Unicode", RFC 1642, Taligent, Inc., July 1994.

[UTF8]   F. Yergeau "UTF-8: a transformation format of Unicode and ISO 10646", RFC2044, Alis Technologies, October

1996.

[UNICODE] The Unicode Consortium, "The Unicode Standard -- Worldwide Character Encoding -- Version 1.0", Addison-Wesley, Volume 1, 1991, Volume 2, 1992.

[US-ASCII] Coded Character Set--7-bit American Standard Code for Information Interchange, ANSI X3.4-1986.

[DNS] P. Mockapetris "Domain Names - Concepts and Facilities", RFC1034, ISI, November 1987, "Domain Names - Implementation and Specification", RFC1035, ISI, November 1987.

[SMTP]
[RFC822] Jonathan B. Postel "Simple Mail Transfer Protocol", RFC821, ISI, August 1982. David H. Crocker "Standard for ARPA Internet Text Messages", RFC822, Dept of Electrical Engineering, Univeristy of Delaware, August 1982.

[MIME] "Multipurpose Internet Mail Extensions", RFC1341, N. Borensten, Bellcore, N. Freed, Innosoft, June 1992.

[IETFPC] "IETF Policy on Character Sets and Languages", RFC2277 BCP18, H. Alvestrand, Jan 1998.

8. Author Address

James C.H Seng
BioInformatrix Pte Ltd
102 Elm Street
Menlo Park CA 94025

Tel: (650) 322-6505
E-mail: jseng@pobox.org.sg

Martin J. Duerst
World Wide Web Consortium
Keio Research Institute at SFC
Keio University
Fujisawa
252-8520 Japan

Tel: +81 446 49 11 70
E-mail: mduerst@w3.org

NOTE -- Please write the author's name with u-Umlaut wherever possible, e.g. in HTML as D&uuml;rst.

Tin Wee Tan, Dr
National University of Singapore
c/o BioInformatic Center
National University Hospital
Lower Kent Ridge Road
Singapore 119074

Tel: +65 774 7149
E-mail: tinwee@post1.com

This memo is also archived at http://www.idns.org/technical.html