Network Working Group
Request for Comments: 1843
Category: Informational

F. Lee
Stanford University
August 1995

HZ - A Data Format for Exchanging Files of Arbitrarily Mixed Chinese and ASCII characters

Status of this Memo

Abstract

The content of this memo is identical to an article of the same title written by the author on September 4, 1989. In this memo, GB stands for GB2312-80. Note that the title is kept only for historical reasons. HZ has been widely used for purposes other than "file exchange".

1. Introduction

Most existing computer systems which can handle a text file of arbitrarily mixed Chinese and ASCII characters use 8-bit codes. To exchange such text files through electronic mail on ASCII computer systems, it is necessary to encode them in a 7-bit format. A generic binary to ASCII encoder is not sufficient, because there is currently no universal standard for such 8-bit codes. For example, CCDOS and Macintosh’s Chinese OS use different internal codes. Fortunately, there is a PRC national standard, GuoBiao (GB), for the encoding of Chinese characters, and Chinese characters encoded in the above systems can be easily converted to GB by a simple formula. (* The ROC standard BIG-5 is outside the scope of this article.)

HZ is a 7-bit data format proposed for arbitrarily mixed GB and ASCII text file exchange. HZ is also intended for the design of terminal emulators that display and edit mixed Chinese and ASCII text files in real time.

2. Specification

The format of HZ is described in the following.

Without loss of generality, we assume that all Chinese characters (HanZi) have already been encoded in GB. A GB (GB1 and GB2) code is a two byte code, where the first byte is in the range $21-$77 (hexadecimal), and the second byte is in the range $21-$7E.

A graphical ASCII character is a byte in the range $21-$7E. A non-graphical ASCII character is a byte in the range $0-$20 or of the value $7F.

Since the range of a graphical ASCII character overlaps that of a GB byte, a byte in the range $21-$7E is interpreted according to the mode it is in. There are two modes, namely ASCII mode and GB mode.

By convention, a non-graphical ASCII character should only appear in ASCII mode.

The default mode is ASCII mode.

In ASCII mode, a byte is interpreted as an ASCII character, unless a '~' is encountered. The character '~' is an escape character. By convention, it must be immediately followed ONLY by '~', '{' or '\n' (<LF>), with the following special meaning.

- o The escape sequence '~~' is interpreted as a '~'.
- o The escape-to-GB sequence '~{' switches the mode from ASCII to GB.
- o The escape sequence '~\n' is a line-continuation marker to be consumed with no output produced.

In GB mode, characters are interpreted two bytes at a time as (pure) GB codes until the escape-from-GB code '~}' is read. This code switches the mode from GB back to ASCII. (Note that the escape-from-GB code '~}' ($7E7D) is outside the defined GB range.)

The decoding process is clear from the above description.

The encoding process is straightforward. Note that an (ASCII) '~' is always encoded as '~~'. A sequence of GB codes is enclosed in '~{' and '~}'.

3. Remarks & Recommendations

We choose to encode any ASCII character except '~' as it is, rather than as a two byte code, and we choose ASCII as the default mode for the following reasons. The computer systems we use is ASCII based. A HZ file containing pure ASCII characters (i.e. no Chinese characters) except '~' is precisely a pure ASCII file. In general, the English (ASCII) portion of a HZ file is directly readable.

The escape character '~' is chosen not only because it is commonly used in the ASCII world, but also because '~' ($7E) is outside the defined range ($21-$77) of the first byte of a GB code.

In ASCII mode, other potential escape sequences, i.e., two byte sequences beginning with '~' (other than '~~', '~{', '~\n') are currently invalid HZ sequences. Hence, they can be used for future extension of HZ with total upward compatibility.

The line-continuation marker '~\n' is useful if one wants to encode long lines in the original text into short lines in this data format without introducing extra newline characters in the decoding process.

There is no limit on the length of a line. In fact, the whole file could be one long line or even contain no newline characters. Any DECODER of this HZ data format should not and has no need to operate on the concept of a line.

It is easy to write encoders and decoders for HZ. An encoder or decoder needs to lookahead at most one character in the input data stream.

Given the current mode, it is also possible and easy to decode a HZ data stream by scanning backward. One of the implication is that "backspaces" can be handled correctly by a terminal emulator.

To facilitate the effective use of programs supporting line/page skips such as "more" on UNIX with a terminal emulator understanding the HZ format, it is RECOMMENDED that the ENCODER (which outputs in HZ) sets a maximum line size of less than 80 characters. Since '\n' is an ASCII character, the syntax of HZ then automatically implies that GB codes appearing at the end of a line must be terminated with the escape-from-GB code '~}', and the line-continuation marker '~\n' should be inserted appropriately. The price to paid is that the encoded file size is slightly larger.

It is important to understand the following distinction. Note that the above recommendation does NOT change the HZ format. It is simply an encoding "style" which follows the syntax of HZ. Note that this "style" is not built into HZ. It is an additional convention built "on top of" HZ. Other applications may require different "styles", but the same basic HZ DECODER will always work. The essence of HZ is to provide such a flexible basic data format for files of arbitrarily mixed Chinese and ASCII characters.

4. Examples

To illustrate the "stylistic" issue of HZ encoding, we give the following four examples of encoded text, which should produce the same decoded output. (The recommendation in the last section refers to Example 2.)

```
Example 1:  (Suppose there is no line size limit.)
This sentence is in ASCII.
The next sentence is in GB.~{<:Ky2;S{#,NpJ)l6HK!#~}Bye.

Example 2:  (Suppose the maximum line size is 42.)
This sentence is in ASCII.
The next sentence is in GB.~{<:Ky2;S{#,~}~
~{NpJ)l6HK!#~}Bye.

Example 3: (Suppose a new line is started whenever there is a mode
           switch.)
This sentence is in ASCII.
The next sentence is in GB.~
~{<:Ky2;S{#,NpJ)l6HK!#~}~
Bye.
```

Acknowledgement

Edmund Lai was the first one who brought my attention to this topic. Discussions with Ed, Tin-Fook Ngai, Yagui Wei and Ricky Yeung were very helpful in shaping the ideas in this article. Thanks to Tin-Fook for his careful review of the draft and numerous interesting suggestions.

References

[1] Fung Fung Lee, "HZ - A Data Format for Exchanging Files of Arbitrarily Mixed Chinese and ASCII Characters," September 4, 1989.
As part of //ftp.ifcss.org/software/unix/convert/HZ-2.0.tar.gz

Security Considerations

Security issues are not addressed in this memo.

Author's Address

Fung Fung Lee
Computer Systems Laboratory
Stanford University
Stanford, CA 94309

Phone: +1 415 723 1450
EMail: lee@csl.stanford.edu