Network Working Group
INTERNET DRAFT

R. Gellens
Unisys
February 21, 1996
Document: draft-gellens-telnet-char-option-01.txt
Postscript: draft-gellens-telnet-char-option-01.ps

TELNET CHARSET Option

Status of this Memo

This document is an Internet-Draft. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as ``work in progress."

To learn the current status of any Internet-Draft, please check the ``1id-abstracts.txt" listing contained in the Internet-Drafts Shadow Directories on ftp.is.co.za (Africa), nic.nordu.net (Europe), munnari.oz.au (Pacific Rim), ds.internic.net (US East Coast), or ftp.isi.edu (US West Coast).

1. Abstract

This document specifies a mechanism for passing character set and translation information between a TELNET client and server. Use of this mechanism enables an application used by a TELNET user to send and receive data in the correct character set.

Either side can (subject to option negotiation) at any time request that a (new) character set be used.

2. Command Names and Codes

CHARSET.. xx

REQUEST 01
ACCEPTED.. 02
REJECTED... 03
TTABLE-SEND... 04
TTABLE-IS 05
TTABLE-REJECTED... 06
TTABLE-ACK... 07
TTABLE-NAK 08

As a convenience, standard TELNET text and codes for commands used in this document are reproduced here (excerpted from [1]):

> All TELNET commands consist of at least a two byte sequence: the "Interpret as Command" (IAC) escape character followed by the code for the command. The commands dealing with option negotiation are three byte sequences, the third byte being the code for the option referenced. ... [O]nly the IAC need be doubled to be sent as data, and the other 255 codes may be passed transparently. The following are [some of] the defined TELNET commands. Note that these codes and code sequences have the indicated meaning only when immediately preceded by an IAC.

NAME	CODE	MEANING
SE	240	End of subnegotiation parameters.
SB	250	Indicates that what follows is subnegotiation of the indicated option.
WILL (option code)	251	Indicates the desire to begin performing, or confirmation that you are now performing, the indicated option.
WON'T (option code)	252	Indicates the refusal to perform, or continue performing, the indicated option.
DO (option code)	253	Indicates the request that the other party perform, or confirmation that you are expecting the other party to perform, the indicated option.
DON'T (option code)	254	Indicates the demand that the other party stop performing, or confirmation that you are no longer expecting the other party to perform, the indicated option.
IAC	255	Data Byte 255.

3. Command Meanings

IAC WILL CHARSET

The sender REQUESTS permission to, or AGREES to, use CHARSET option subnegotiation to choose a character set.

IAC WON'T CHARSET

The sender REFUSES to use CHARSET option subnegotiation to choose a character set.

IAC DO CHARSET

The sender REQUESTS that, or AGREES to have, the other side use CHARSET option subnegotiation to choose a character set.

IAC DON'T CHARSET

The sender DEMANDS that the other side not use the CHARSET option subnegotiation.

IAC SB CHARSET REQUEST <char set list> IAC SE

Char set list:

<sep> <character set>

or

<sep> <character set> …<sep> <character set>

This message initiates a new CHARSET subnegotiation. It can only be sent by a side that has received a DO CHARSET message and sent a WILL CHARSET message (in either order).

The sender requests that all text sent to and by it be encoded in one of the specified character sets.

<Char set list> is a sequence of 7-BIT ASCII printable characters. The first octet defines the separator character (which must not appear within any character set). It is terminated by the IAC SE sequence. Case is not significant. It consists of one or more character sets. The character sets should appear in order of preference (most preferred first).

<Sep> is a separator octet, the value of which is chosen by the sender. Examples include a space or a semicolon. Any value other than IAC is allowed. The obvious choice is a space or any other punctuation symbol which does not appear in any of the character set names.

<Character set> is a sequence of 7-BIT ASCII printable characters. Case is not significant.

If a requested character set is registered with the Internet Assigned Number Authority (IANA) [2], it is required that the standardized spelling of its name or a registered alias be used. While it is permitted to request non-standard character sets such as those not registered with IANA, this is strongly discouraged, as such character sets are unlikely to be recognized by the receiver of the CHARSET REQUEST message. Even worse, a non-registered character set could have the same name as some other character set which is registered. Each side would then be using a character set different from that expected by the other.

The receiver responds in one of four ways:

If the receiver is already sending text to and expecting text from the sender to be encoded in one of the specified character sets, it sends a **positive acknowledgment** (CHARSET ACCEPTED); it MUST NOT ignore the message. (Although ignoring the message is perhaps suggested by some interpretations of the relevant RFCs ([1], [3]), in the interests of determinacy it is not permitted. This ensures that the issuer does not need to time out and infer a response, while avoiding (because there is no response to a **positive acknowledgment**) the non-terminating subnegotiation which is the rationale in the RFCs for the non-response behavior.)

If the receiver is capable of handling at least one of the specified character sets, it can respond with a **positive acknowledgment** for one of the requested character sets. It may pick the first set it is capable of handling or choose one based on its own preferences. After doing so, each side MUST encode subsequent text in the specified character set.

If the receiver is not capable of handling any of the specified character sets, but is capable of receiving a translate table to enable it to do so, it can send a **request for translate table** (TTABLE-SEND) response.

If the receiver is not capable of handling any of the specified character sets nor of receiving a translate table, it sends a **negative acknowledgment** (CHARSET REJECTED).

Because it is not valid to reply to a CHARSET REQUEST message with another CHARSET REQUEST message, if a CHARSET REQUEST message is received after sending one, it means that both sides have sent them simultaneously. In this case, the server side MUST issue a **negative acknowledgment**. The user side MUST respond to the one from the server.

IAC SB CHARSET ACCEPTED <Charset> IAC SE

This is a positive acknowledgment response to a CHARSET REQUEST message; the receiver of the CHARSET REQUEST message acknowledges its receipt and accepts the indicated character set.

<Charset> is a character sequence identical to one of the character sets in the CHARSET REQUEST message. It is terminated by the IAC SE sequence.

Text messages which follow this response must now be coded in the indicated character set. This message terminates the current CHARSET subnegotiation.

IAC SB CHARSET REJECTED IAC SE

This is a negative acknowledgment response to a CHARSET REQUEST message; the receiver of the CHARSET REQUEST message acknowledges its receipt but refuses to use any of the requested character sets. Messages can not be sent in any of the indicated character sets. This message can also be sent in response to a TTABLE-IS message, if the receiver of the TTABLE-IS message has problems with it. This message terminates the current CHARSET subnegotiation.

IAC SB CHARSET TTABLE-SEND <version> <char set list> IAC SE

This is a "No, but if you hum a few bars I can fake it" acknowledgment response to a CHARSET REQUEST message; the receiver of the CHARSET REQUEST message acknowledges its receipt and requests the sender to transmit a translate table specifying the mapping between a pair of character sets, one of which appeared in the CHARSET REQUEST message, the other appears in this TTABLE-SEND message. In other words, the sender of the TTABLE-SEND message requests a mapping between any character set of the CHARSET REQUEST message and any character set of this TTABLE-SEND message. The receiver of the TTABLE-SEND message is free to select any convenient mapping.

<Version> is an octet whose binary value is the highest version level of the TTABLE-SEND message which can be sent in response. This field must not be zero. See the TTABLE-IS message for the permitted version values.

Char set list:

<sep> <character set>

or

<sep> <character set> …<sep> <character set>

<Char set list> is a sequence of 7-BIT ASCII printable characters. Case is not significant. The first octet defines the separator character (which must not appear within any character set). It is terminated by the IAC SE sequence. It consists of one or more character sets. The character sets should appear in order of preference (most preferred first). If a character set is registered with IANA, it is required that the standardized spelling of its name or a registered alias be used.

<Sep> is a separator octet, the value of which is chosen by the sender. Examples include a space or a semicolon. Any value other than IAC is allowed. The obvious choice is a space or any other punctuation symbol which does not appear in either of the character set names.

<Character set> is a sequence of 7-BIT ASCII printable characters. Case is not significant.

If the receiver of the TTABLE-SEND message is not capable of sending a translate table for any of the character sets, or is not capable of doing so without using a version of the TTABLE-IS message higher than <version>, it sends a TTABLE-REJECTED message.

IAC SB CHARSET TTABLE-IS <version> <syntax for version> IAC SE

In response to a TTABLE-SEND message, the receiver of the TTABLE-SEND message acknowledges its receipt and is transmitting a pair of tables which define the mapping between the specified character sets.

<Version> is an octet whose binary value is the version level of this TTABLE-IS message. Different versions have different syntax. The lowest version level is one (zero is not valid). The current highest version level is also one. This field is provided so that future versions of the TTABLE-SEND message can be specified, for example, to handle character sets for which there is no simple one-to-one character-for-character translation. This might include some forms of multi-octet character sets for which translation algorithms or subsets need to be sent.

Syntax for Version 1:

<sep> <char set name 1> <sep> < char size 1> < char count 1> <char set name 2> <sep> <char size 2> <char count 2> <map 1> <map 2>

<Sep> is a separator octet, the value of which is chosen by the sender. Examples include a space or a semicolon. Any value other than IAC is allowed. The obvious choice is a space or any other punctuation symbol which does not appear in either of the character set names.

<Char set name 1> and **<Char set name 2>** are sequences of 7-BIT ASCII printable characters which identify the two character sets for which a mapping is being specified. Each is terminated by <sep>. Case is not significant. If a character set is registered with IANA, it is required that the standardized spelling of its name or a registered alias be used. **<Char set name 1>** should be chosen from the **<char set list>** in the CHARSET REQUEST message. **<Char set name 2>** should be chosen from the **<char set list>** in the TTABLE-SEND message. Text on the wire should be encoded using **<char set name 1>**.

<Char size 1> and **<char size 2>** are single octets each. The binary value of the octet is the number of bits nominally required for each character in the corresponding table. It should be a multiple of eight. *[**Note to implementers:** since TCP/IP works in octets, it is possible for octets of value 255 to appear "spontaneously" when using non-8-bit characters.*]

<Char count 1> and **<char count 2>** are each three-octet binary fields in Network Byte Order [6]. Each specifies how many characters (of the maximum 2**<char size>) are being transmitted in the corresponding map.

<Map1> and **<Map 2>** each consist of the corresponding <char count> number of characters. These characters form a mapping from all or part of the characters in one of the specified character sets to the correct characters in the other character set. If the indicated <char count> is less than 2**<char size>, the first <char count> characters are being mapped, and the remaining characters are assumed to not be changed (and thus map to themselves). That is, each map contains characters 0 through <char count> -1. <Map 1> maps from <char set name 1> to <char set name 2>. <Map 2> maps from <char set name 2> to <char set name 1>. Translation between the character sets is thus an obvious process of using the binary value of a character as an index into the appropriate map. The character at that index replaces the original character. If the index exceeds the <char count> for the map, no translation is performed for the character.

IAC SB CHARSET TTABLE-REJECTED IAC SE

In response to a TTABLE-SEND message, the receiver of the TTABLE-SEND message acknowledges its receipt and indicates it is unable to comply with the request. This message terminates the current CHARSET subnegotiation.

This message could be sent, for example, because the receiver does not have a mapping between the character sets specified in the CHARSET REQUEST message. Or perhaps it cannot send such a mapping using a version of the TTABLE-IS message which is less than or equal to the version specified in the TTABLE-SEND message.

IAC SB CHARSET TTABLE-ACK IAC SE

The sender acknowledges the successful receipt of the translate table. Text messages which follow this response must now be coded in the character set specified as **<char set name 1>** of the TTABLE-IS message. This message terminates the current CHARSET subnegotiation.

IAC SB CHARSET TTABLE-NAK IAC SE

The sender reports the unsuccessful receipt of the translate table and requests that it be resent. If subsequent transmission attempts also fail, a TTABLE-REJECTED or CHARSET REJECTED message (depending on which side sends it) should be sent instead of additional futile TTABLE-IS and TTABLE-NAK messages.

Any system which supports the CHARSET option MUST fully support the CHARSET REQUEST, ACCEPTED, REJECTED, and TTABLE-REJECTED subnegotiation messages. It MAY optionally fully support the TTABLE-SEND, TTABLE-ACK, and TTABLE-NAK messages. If it does fully support the TTABLE-SEND message, it MUST also fully support the TTABLE-ACK and TTABLE-NAK messages. If it does not fully support the TTABLE-SEND message, it MUST at least recognize it and respond with a TTABLE-REJECTED message.

4. Default

WON'T CHARSET

DON'T CHARSET

5. Motivation for the Option

Many computer systems now utilize a variety of character sets. Increasingly, a server computer needs to translate transmissions and receptions using different pairs of character sets on a per-application or per-connection basis. This is becoming more common as user and server computers become more geographically disperse. (And as servers are consolidated into ever-larger hubs, serving ever-wider areas.) In order for files, databases, etc. to contain correct data, the server must determine the character set in which the user is sending, and the character set in which the application expects to receive.

In some cases, it is sufficient to determine the character set of the end user (because every application on the server expects to use the same character set), but in other cases different server applications expect to use different character sets. In the former case, an initial CHARSET subnegotiation suffices. In the latter case, the server may need to initiate additional CHARSET subnegotiations as the user switches between applications.

6. Description of the Option

When the user TELNET program is able to determine the user's character set it should offer to specify the character set by sending IAC WILL CHARSET.

If the server system is able to make use of this information, it replies with IAC DO CHARSET. The user TELNET is then free to request a character set in a subnegotiation at any time.

Likewise, when the server is able to determine the expected character set(s) of the user's application(s), it should send IAC DO CHARSET to request that the user system specify the character set it is using. Or the server could send IAC WILL CHARSET to offer to specify the character sets.

Once a character set has been determined, the server can either perform the translation between the user and application character sets itself, or request by additional CHARSET subnegotiations that the user system do so.

Once it has been established that both sides are capable of character set negotiation (that is, each side has received either a WILL CHARSET or a DO CHARSET message, and has also sent either a DO CHARSET or a WILL CHARSET message), subnegotiations can be requested at any time by whichever side has sent a WILL CHARSET message and also received a DO CHARSET message (this may be either or both sides). Once a CHARSET subnegotiation has started, it must be completed before additional CHARSET subnegotiations can be started (there must never be more than one CHARSET subnegotiation active at any given time). When a subnegotiation has completed, additional subnegotiations can be started at any time.

If either side violates this rule and attempts to start a CHARSET subnegotiation while one is already active, the other side MUST reject the new subnegotiation by sending a CHARSET REJECTED message.

Receipt of a CHARSET REJECTED or TTABLE-REJECTED message terminates the subnegotiation, leaving the character set unchanged. Receipt of a CHARSET ACCEPTED or TTABLE-ACK message terminates the subnegotiation, with the new character set in force.

In some cases, both the server and the user systems are able to perform translations and to send and receive in the character set(s) expected by the other side. In such cases, either side can request that the other use the character set it prefers. When both sides simultaneously make such a request (send CHARSET REQUEST messages), the server MUST reject the user's request by sending a CHARSET REJECTED message. The user system MUST respond to the server's request. (See the CHARSET REQUEST description, above.)

When the user system makes the request first, and the server is able to handle the requested character set(s), but prefers that the user system instead use the server's (user application) character set, it may reject the request, and issue a CHARSET REQUEST of its own. If the user system is unable to comply with the server's preference and issues a CHARSET REJECTED message, the server can issue a new CHARSET REQUEST message for one of the previous character sets (one of those which the user system originally requested). The user system would obviously accept this character set.

While a CHARSET subnegotiation is in progress, data should be queued. Once the CHARSET subnegotiation has terminated, the data can be sent (in the correct character set).

Note that regardless of CHARSET negotiation, translation only applies to text (not commands), and only occurs when in BINARY mode [4]. If not in BINARY mode, all data is assumed to be in NVT ASCII [1].

Also note that the CHARSET option should be used with the END OF RECORD option [5] for block-mode terminals in order to be clear on what character represents the end of each record.

As an example of character set negotiation, consider a user on a workstation using TELNET to communicate with a server. In this example, the workstation normally uses the Cyrillic (ASCII) character set [2] but is capable of using EBCDIC-Cyrillic [2], and the server normally uses EBCDIC-Cyrillic. The server could handle the (ASCII) Cyrillic character set, but prefers that instead the user system uses the EBCDIC-Cyrillic character set. (This and the following examples do not show the full syntax of the subnegotiation messages.)

USER	**SERVER**
WILL CHARSET	WILL CHARSET
DO CHARSET	DO CHARSET
CHARSET REQUEST Cyrillic EBCDIC-Cyrillic	
	CHARSET ACCEPTED EBCDIC-Cyrillic

For another example, consider the previous case, but this time the workstation cannot handle EBCDIC-Cyrillic, nor can it accept a translate table:

USER	**SERVER**
WILL CHARSET	WILL CHARSET
DO CHARSET	DO CHARSET
CHARSET REQUEST Cyrillic	

	CHARSET REJECTED
	CHARSET REQUEST EBCDIC-Cyrillic
CHARSET REJECTED	
	CHARSET REQUEST Cyrillic
CHARSET ACCEPTED Cyrillic	

For the next example, consider the previous case, but this time the workstation **can** accept a translate table:

USER	**SERVER**
WILL CHARSET	WILL CHARSET
DO CHARSET	DO CHARSET
CHARSET REQUEST Cyrillic	
	CHARSET REJECTED
	CHARSET REQUEST EBCDIC-Cyrillic
CHARSET TTABLE-SEND Cyrillic	
	CHARSET TTABLE-IS
CHARSET TTABLE-ACK	

For another example, consider the previous case, but now the user switches server applications in the middle of the session (denoted by ellipses), and the new application requires a different character set:

USER	**SERVER**
WILL CHARSET	WILL CHARSET
DO CHARSET	DO CHARSET
CHARSET REQUEST Cyrillic EBCDIC-INT	
	CHARSET REJECTED
	CHARSET REQUEST EBCDIC-Cyrillic
CHARSET TTABLE-SEND Cyrillic EBCDIC-INT	
	CHARSET TTABLE-IS
CHARSET TTABLE-ACK	
. . .	. . .
	CHARSET REQUEST EBCDIC-INT
CHARSET ACCEPTED EBCDIC-INT	

7. Security Considerations

This document raises no security issues.

8. References

[1] Postel, J. and Reynolds, J., “Telnet Protocol Specification”, STD 8, RFC 854, ISI, May 1983

[2] Reynolds, J., and Postel, J., “Assigned Numbers”, STD 2, RFC 1700, ISI, October 1994.

[3] Postel, J. and Reynolds, J., “Telnet Option Specifications”, STD 8, RFC 855, ISI, May 1983

[4] Postel, J. and Reynolds, J., “Telnet Binary Transmission”, RFC 856, ISI, May 1983

[5] Postel, J., “Telnet End-Of-Record Option”, RFC 885, ISI, December 1983

[6] Postel, J., “Internet Official Protocol Standards”, STD 1, RFC 1780, IAB, March 1995

9. Author’s Address

Randall C. Gellens
Unisys Corporation
25725 Jeronimo Road
Mail Stop 237
Mission Viejo, CA 92691
USA

Phone: +1.714.380.6350
Fax: +1.714.380.5912

Randy@MV.Unisys.Com