Configuring TN3270

IBM 3270 display terminals are among the computing community's most widely implemented and emulated environments for host-based computing. This chapter will help you understand the TN3270 terminal emulation environment and how to use and create files that allow terminals connected to the routers to be used for TN3270 operation. For a complete description of the commands in this chapter, refer to the chapter "TN3270 Configuration Commands" later in this publication. For information about establishing TN3270 connections, refer to the *Cisco Access Connection Guide*.

Cisco's Implementation of TN3270

The TN3270 terminal emulation software is based on software developed at the University of California, Berkeley. This software allows any terminal to be used as an IBM 3270-type terminal. Users with non-3270 terminals can take advantage of the emulation capabilities to perform the functions of an IBM 3270-type terminal. Specifically, Cisco's implementation supports emulation of an IBM 3278-2 terminal providing an 80-by-24 display.

True IBM 3270-type terminals use a character format referred to as extended binary-coded decimal interchange code (EBCDIC). EBCDIC consists of 8-bit coded characters and was originally developed by IBM. Emulation is made possible by *termcap* and *curses* functions developed by Berkeley UNIX system developers. These functions translate the keyboard and terminal characteristics for ASCII-type terminals into those expected by an IBM host. ASCII characters are listed in the appendix "ASCII Character Set" later in this publication.

Formally, a termcap is a two-part terminal-handling mechanism. It consists of a database and a subroutine library. The database describes the capabilities of each terminal to be supported; the subroutine library allows programs to query the database and to make use of the values it contains. For more information about defining termcaps, refer to the document *termcap & terminfo*, by Jim Strang, Tim O'Reilly, and Linda Mui.

Routers include a default termcap entry for Digital VT100 terminal emulation. More samples are available directly from Cisco on the *ftp.cisco.com* directory using the File Transfer Protocol (FTP) utility.

TN3270 emulation capability allows users to access an IBM host without using a special IBM server or a UNIX host acting as a server (see Figure 3-1). The IBM host must directly support TCP/IP, or have a front-end processor that supports TCP/IP.

Connection to IBM hosts from LAT, TCP, and X.25/PAD environments is accomplished using the two-step translation method. This method is discussed in the chapter "Configuring Protocol Translation Sessions" in this publication and in the *Cisco Access Connection Guide*. In general, TN3270 support for protocol translation allows outgoing TN3270 connections only. In other words, LAT, TCP, and X.25/PAD users must first establish a connection with the router, then use the TN3270 facility from the router to make a connection to the IBM host.
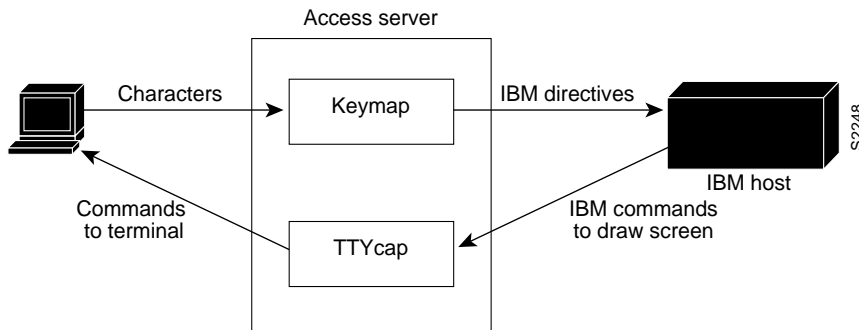
Figure 3-1   **Typical 3270 Connection Environment**
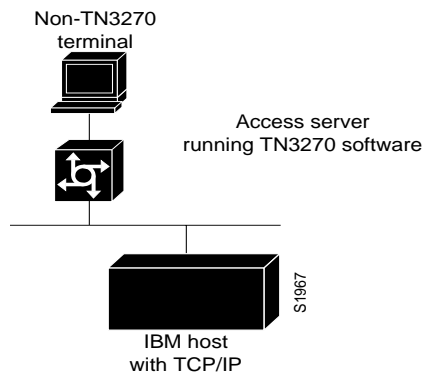
Keymaps and Ttycaps

Figure 3-2 shows how the keymapping and ttycap functionality on the router helps IBM hosts and non-IBM terminals communicate.

Figure 3-2   **Keymaps and Ttycaps**

Keymaps and ttycaps have the following functionality:

- Keymapping—Terminals send a key sequence for every key used to send packets to an IBM host.The keymapping function on the router identifies special sequences and converts them to directives to the IBM host. A minimal level of keymapping is supported by default. Several keys can convert to the same IBM directives.

- Ttycap—IBM sends commands to the terminal, including cursor position, clear screen, and so forth. The ttycap functionality on the router changes IBM directives into the terminal language. By default, routers conform to the ANSI terminal standard, which is VT*xxx* terminal compatible.
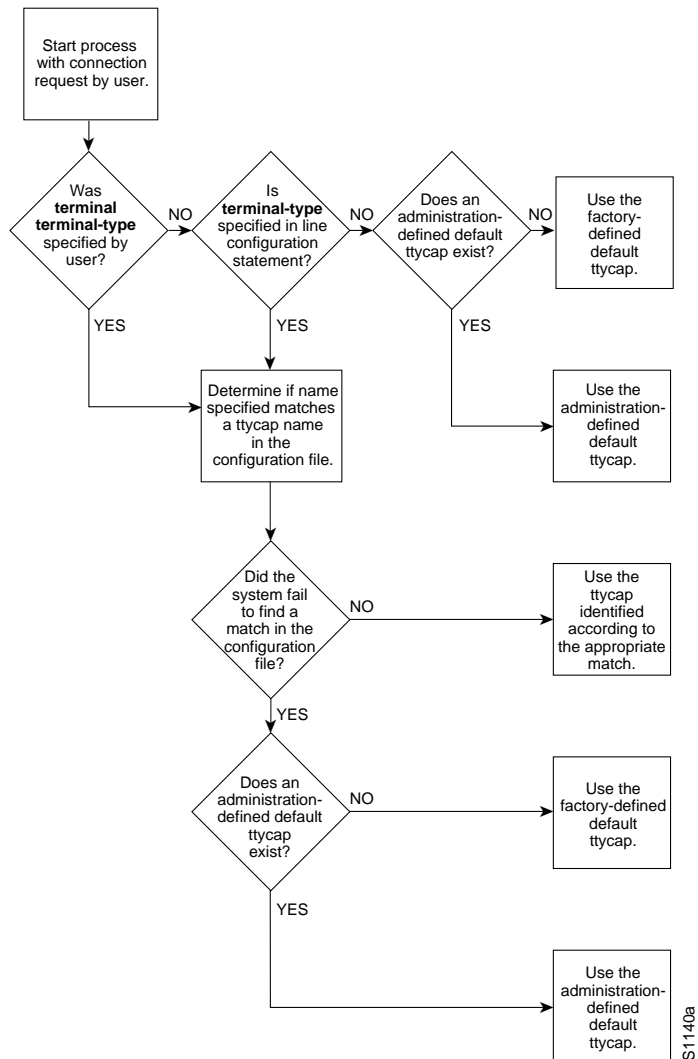
Startup Sequence Priorities

At system startup, the router uses the following decision sequence when selecting a terminal emulation file, also called a *ttycap*:

1 Use a user-supplied terminal emulation filename.

2 Use a terminal emulation filename specified using line configuration commands.

3 Use a default terminal emulation filename supplied by the administrator.

4 Use the default VT100 emulation.

Figure 3-3 illustrates the decision process used by the router to choose a ttycap for a specific TN3270 session.

Figure 3-3   **Typical TN3270 Ttycap Selection Process**

At system startup, the router uses the following decision sequence when selecting a keyboard map file, also called a *keymap*:

**1**   Use a user-supplied keyboard map filename.

2   Use a keyboard map filename specified using line configuration commands.

3   Use a user-supplied terminal emulation filename.

4   Use a terminal emulation filename specified using line configuration commands

5   Use the default keyboard map filename supplied by the administrator.

6   Use the default VT100 emulation.

The router uses the following criteria to determine which file to use:

- If a filename is specified by the user but fails to match any name in the configuration file, the router adopts the default specified by the administrator. If one has not been specifically defined, the factory default emulation file is adopted.

- If a filename is specified for line configuration that does not match any name in the configuration file, the router adopts the default specified by the administrator. If one has not been specifically defined, the factory default VT100 emulation file is used.

Figure 3-4 illustrates the decision process used by the router to choose a keymap for a specific TN3270 session. When one of the first four priority checks fail (that is, the name specified does not match any name in the configuration file), the same rules listed for the terminal emulation file apply.

Figure 3-4   Typical TN3270 Keymap Selection Process
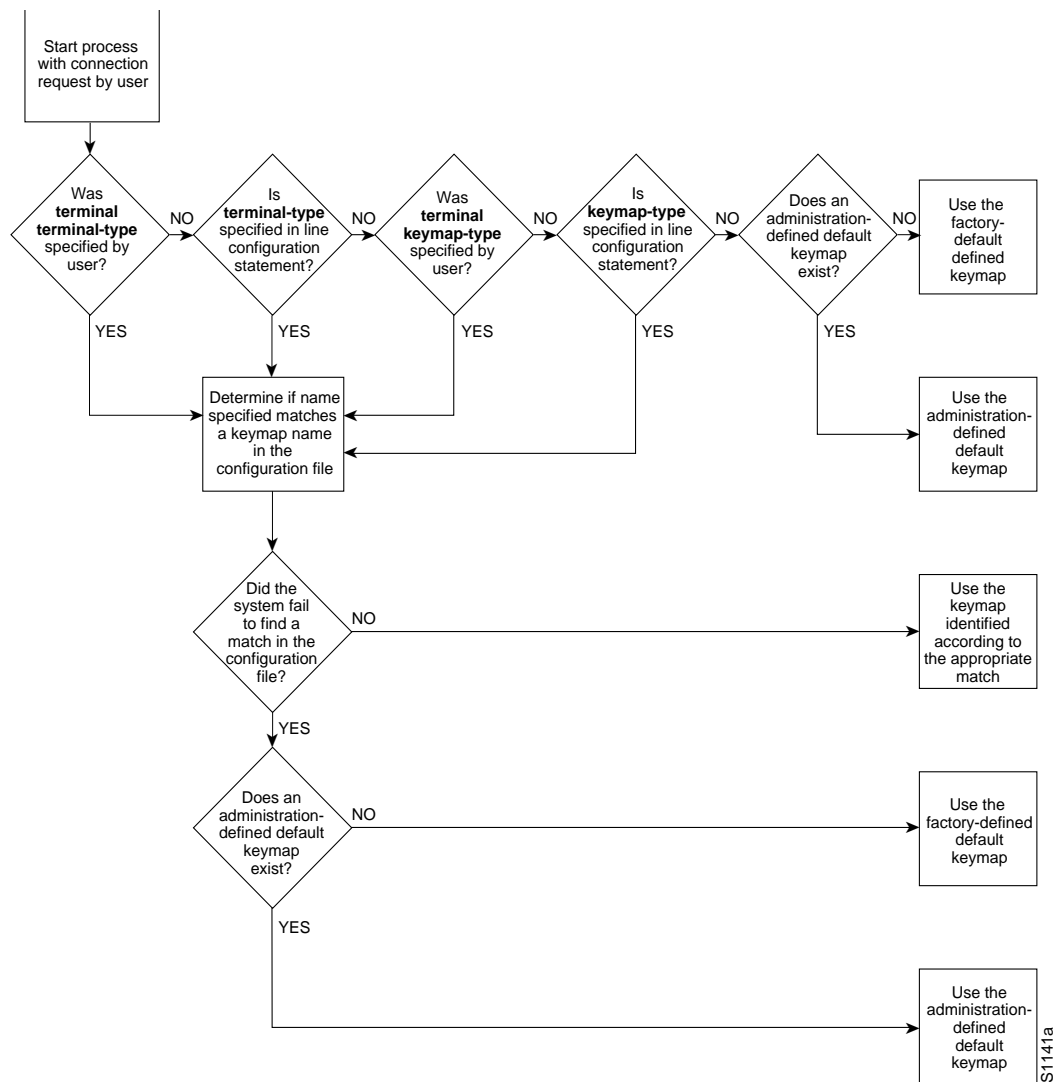
TN3270 Connection and Configuration Task List

You can perform the tasks described in the following sections when making connections to an IBM host:

- Use the Default VT100 Terminal and Keyboard Emulation

- Copy a Sample Terminal Emulation File

- Create a Custom Terminal Emulation File

- Assign Ttycap and Keymap Line Characteristics

- List the Terminal Emulation Files

- Map TN3270 Characters

See the section "TN3270 Configuration Examples" for examples of custom terminal and keyboard emulation files.

Use the Default VT100 Terminal and Keyboard Emulation

By default, an ASCII terminal and keyboard connected to the router emulate a Digital VT100 terminal type.

Enter the **tn3270** EXEC command at the EXEC prompt to connect to an IBM host. This command makes the connection using the terminal emulation file selected and using the startup sequence priorities outlined in the section "Startup Sequence Priorities" earlier in this chapter. For more information about the **tn3270** command, refer to the *Cisco Access Connection Guide*.

Copy a Sample Terminal Emulation File

If the default file does not work for your terminal and keyboard type or the host that you connect to, you might be able to find a file that will work from the growing list of sample terminal emulation files created by Cisco engineers and customers. Obtain the *tn3270.examples* file in the *ftp.cisco.com* directory at Cisco Systems. Numerous emulation files are listed in *tn3270.examples* that allow various terminal types to emulate an IBM 3270-type terminal.

The following steps describe how to obtain a sample configuration file from the *tn3270.examples* file available from the Cisco *ftp.cisco.com* directory. These steps assume that the system configuration file is stored on a host.

Step 1 Use the FTP utility to connect to address *ftp.cisco.com* and log in using the anonymous FTP convention.

Step 2 Get the *tn3270.examples* file.

Step 3 Use a text editor or word processing application to copy the sample terminal emulation file into the configuration file.

Step 4 Load the configuration file onto the host or network. (Refer to the chapter about loading system images, microcode images, and configuration files in the *Router Products Configuration Guide* for more information on loading configuration files.)

These steps add new terminal emulation capability to the configuration file. Each time the system is started up, or booted, the settings in the file will be used as the default for terminal emulation.

Create a Custom Terminal Emulation File

To use a custom emulation file, you must load the emulation settings into the system configuration file. This establishes the settings in the file as the terminal and keyboard defaults and provides several ways in which the emulation settings can be used within the system, as follows:

- You can provide default settings for all terminals in the network or terminals on a specific host.

- You can set up your system to boot, or load, a specific configuration file using configuration commands described in the chapter about loading system images, microcode images, and configuration files in the *Router Products Configuration Guide*.

- You can temporarily override default settings using terminal EXEC commands.

- You can use the local **terminal terminal-type** and **terminal keyboard-type** EXEC commands described in the *Cisco Access Connection Guide* to load in the files.

- You can configure line-specific emulation types for terminal negotiations with a remote host.

To create a custom terminal emulation file or a custom keyboard emulation file, perform the applicable global configuration task as follows:

Task	Command
Define a new terminal emulation file, or ttycap.	**ttycap** *ttycap-name termcap-entry*
Define a new keyboard emulation file, or keymap.	**keymap** *keymap-name keymap-entry*

Assign Ttycap and Keymap Line Characteristics

If you intend to use an alternate ttycap and keymap, you must assign the following two characteristics:

- Terminal type

- Keymap type

This information is used by the router when negotiating connections with hosts.

To assign ttycap and keymap line characters, perform one or more of the following line configuration tasks:

Task	Command
Specify the type of terminal connected to the line.	**terminal-type** *terminal-name*
Specify the keyboard map for a terminal connected to the line.	**keymap-type** *keymap-name*

List the Terminal Emulation Files

To display a list of ttycap and keymap files available for use, perform the following tasks in EXEC mode:

Task	Command
List the ttycap files available.	**show ttycap** [*ttycap-name*	**all**]
List the keymap files available.	**show keymap** [*keymap-name*	**all**]

Map TN3270 Characters

You can control the mapping of extended binary-coded decimal interface code (EBCDIC) and American Standard Code for Information Interchange (ASCII) characters by performing the tasks described in the following sections:

- Create Character Mappings

- Display Character Mappings

- Obtain the Hexadecimal Value

- Set Data Character Bits

- Allow the Use of Extended Features

- Specify a Null-Processing Option

- Specify Reset after Input Error

Create Character Mappings

You can create character mappings by configuring a two-way binding between EBCDIC and ASCII characters. To set character mappings, perform the following global configuration task:

Task	Command
Create a two-way binding between EBCDIC and ASCII characters.	**tn3270 character-map** *ebcdic-in-hex ascii-in-hex*

To return character mappings to their default settings, perform the following global configuration task:

Task	Command
Reset character mappings to defaults.	**no tn3270 character-map** {**all** | *ebcdic-in-hex*} [*ascii-in-hex*]

Display Character Mappings

To display character mappings, perform the following task in EXEC mode:

Task	Command
Display character mappings.	**show tn3270 character-map** {**all** | *ebcdic-in-hex*}

Obtain the Hexadecimal Value

To display the hexadecimal value of an ASCII character, perform the following task in EXEC mode:

Task	Command
Obtain the hexadecimal value of an ASCII character.	**show tn3270 ascii-hexval**

After you enter this command, type the ASCII character whose hexadecimal value you want to display.

Set Data Character Bits

When you create character mappings between EBCDIC or extended ASCII characters, you must configure the router for the correct data character bit length. The default mask used for TN3270 connections is a 7-bit mask. In certain situations, you must use an 8-bit display. When an 8-bit mask has been set, you can temporarily configure the router to use the 8-bit mask by performing the following task in line configuration command mode:

Task	Command
Temporarily configure the router to use the 8-bit mask.	**tn3270 8bit display**

When you use a file-transfer protocol such as Kermit in 8-bit mode or you use 8-bit graphics, which rely on transparent mode, perform the following line configuration task to configure the router for the 8-bit mask:

Task	Command
Configure the router to use the 8-bit mask.	**tn3270 8bit transparent-mode**

Allow the Use of Extended Features

You can allow the use of extended features by directing the router to append "-E" to the terminal type string sent to the IBM host.

To allow the use of extended features, perform the following line configuration task:

Task	Command
Append "-E" to the terminal type string sent to the IBM host, and allow the use of extended features.	**tn3270 datastream** {**extended**	**normal**}

Specify a Null-Processing Option

If a user enters data at a terminal keyboard, then uses an arrow key to move the cursor across the display screen and enters more data, the blanks spaces are filled with nulls. You can specify a method for processing null characters generated from user input at a terminal, emulating either a 3278x terminal in which nulls are compressed out of the entry string, or a 7171 controller in which nulls are replaced by spaces.

In line configuration mode, perform the following tasks to set a null-processing method and return null-processing to the default setting.

Task	Command
Specify the compression of nulls out of the user-entered string.	**tn3270 null-processing 3270**
Specify the conversion of nulls into spaces.	**no 3270 null-processing improved**

Specify Reset after Input Error

To specify that the terminal keyboard be locked until the user presses the Reset key, perform the following task in line configuration mode:

Task	Command
Lock the keyboard upon input error and do not permit any input until the user presses the Reset key.	**tn3270 reset-required**

TN3270 Configuration Examples

The following sections provide examples that will help you define custom terminal and keyboard emulation files and will help you configure your system to use these files:

- Custom Terminal Emulation File Example
- Custom Keyboard Emulation File Example
- Line Specification for a Custom Emulation Example
- Character Mapping Examples

Custom Terminal Emulation File Example

The following example allows a Televideo 925 terminal to emulate an IBM 3270-type terminal. The file is part of the global **ttycap** command and is included in the system configuration file. Notice that a carriage return (^M) indicates the last character in the file.

```
ttycap ttycap1 \
v8|vi|tvi925|925|televideo model 925:\
        :so=\EG4:se=\EG0:\
        :hs:am:bs:co#80:li#24:cm=\E=%+ %+ :cl=\E*:cd=\Ey:ce=\Et:\
        :al=\EE:dl=\ER:im=:ei=:ic=\EQ:dc=\EW:\
        :ho=^^:nd=^L:bt=\EI:pt:so=\EG4:se=\EG0:sg#1:us=\EG8:ue=\EG0:ug#1:\
        :up=^K:do=^V:kb=^H:ku=^K:kd=^V:kl=^H:kr=^L:kh=^^:ma=^V^J^L :\
        :k1=^A@\r:k2=^AA\r:k3=^AB\r:k4=^AC\r:k5=^AD\r:k6=^AE\r:k7=^AF\r:\
        :k8=^AG\r:k9=^AH\r:k0=^AI\r:ko=ic,dc,al,dl,cl,ce,cd,bt:\
        :md=\E(:me=\E):ti=\E):te=\E(:\
        :ts=\Ef:fs=\Eg:ds=\Eh:sr=\Ej:xn:\
        :is=\El\E"^M\E3^M        \E1        \E1        \E1        \E1        \E\
1        \E1        \E1        \E1        \E1^M
```

Custom Keyboard Emulation File Example

The following example allows a keyboard to emulate an asynchronous connection to an IBM 7171 keyboard. The file is part of the **keymap** global configuration command and is included in the system configuration file.

```
keymap ibm7171 \
vt100av|vt100|vt100nam|pt100|vt102|vt125{ \
enter = '^m';\
erase = '^?'; reset = '^g'; clear = '^z' | '\EOM';\
nl = '^j'; tab = '^i'; btab = '^b';\
left = '\EOD'; right = '\EOC'; up = '\EOA'; down = '\EOB';\
home = '^h'; delete = '^d'; eeof = '^e' | '\E^?'; einp = '^w'; insrt = '\EOn';\
pfk1 = '\EOP' | '\E1'; pfk2 = '\EOQ' | '\E2'; pfk3 = '\EOR' | '\E3';\
pfk4 = '\EOw' | '\E4'; pfk5 = '\EOx' | '\E5'; pfk6 = '\EOy' | '\E6';\
pfk7 = '\EOt' | '\E7'; pfk8 = '\EOu' | '\E8'; pfk9 = '\EOv' | '\E9';\
pfk10 = '\EOq' | '\E0'; pfk11 = '\EOr' | '\E-';\
```

```
pfk12 = '\EOs' | '\E='; pfk13 = '\EOp\EOP' | '^f13';\
pfk14 = '\EOp\EOQ' | '^f14'; pfk15 = '\EOp\EOR' | '^f15';\
pfk16 = '\EOp\EOw' | '^f16'; pfk17 = '\EOp\EOx' | '^f17';\
pfk18 = '\EOp\EOy' | '^f18'; pfk19 = '\EOp\EOt' | '^f19';\
pfk20 = '\EOp\EOu' | '^f20'; pfk21 = '\EOp\EOv' | '^f21';\
pfk22 = '\EOp\EOq' | '^f22'; pfk23 = '\EOp\EOr' | '^f23';\
pfk24 = '\EOp\EOs' | '^f24';\
pa1 = '^p1' | '\EOS';\
pa2 = '^p2' | '\EOm';\
pa3 = '^p3' | '\EOl';\
}
```

Line Specification for a Custom Emulation Example

The following example sets up a line with specific terminal and keyboard characteristics that are used during negotiation with a host upon connection. The line configuration commands in the example must follow the **ttycap** and **keymap** global configuration commands containing the emulation settings to be used.

```
line 3
terminal-type ttycap1
keymap-type ibm7171
```

Character Mapping Examples

The following example shows the configuration of the EBCDIC, ASCII character mappings listed in Table 3-1.

Table 3-1   Sample EBCDIC, ASCII Character Mapping

EBCDIC	ASCII
a	x
b	y
c	z

```
tn3270 character-map 0x81 0x78
tn3270 character-map 0x82 0x79
tn3270 character-map 0x83 0x7A
```

The following example displays all nonstandard character mappings:

```
PT# show tn3270 character-map all

EBCDIC 0x81 <=> 0x78 ASCII
EBCDIC 0x82 <=> 0x79 ASCII
EBCDIC 0x83 <=> 0x7A ASCII
```

The following example shows the standard key mapping for *d*:

```
PT# show tn3270 character-map 83

EBCDIC 0x83 <=> 0x63 ASCII = `c'
EBCDIC 0x84 <=> 0x64 ASCII = `d'
```

The following example unmaps a specific key, first with the optional *ascii-in-hex* argument, then without the argument:

```
PT# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
PT(config)# no tn3270 character-map 0x80 0x78
PT(config)# ^Z
PT# show tn3270 character-map all
EBCDIC 0x82 <=> 0x79 ASCII
EBCDIC 0x83 <=> 0x7A ASCII
PT# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
PT(config)# no tn3270 character-map 0x82
PT(config)# ^Z
PT# show t3270 character-map all
EBCDIC 0x82 <=> 0x79 ASCII
```

The following example displays character mappings, then removes all mappings with the **all** keyword:

```
PT# show tn3270 character-map all
EBCDIC 0x81 <=> 0x78 ASCII
EBCDIC 0x82 <=> 0x79 ASCII
EBCDIC 0x83 <=> 0x7A ASCII
PT# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
PT(config)# no tn3270 character-map all
PT(config)# ^Z
PT# show tn3270 character-map all
```