

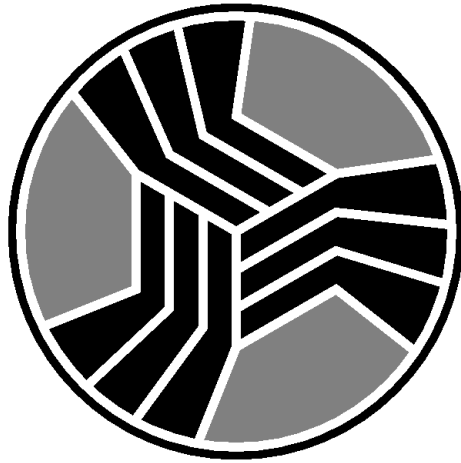
# CABLES

**Terminal Emulation**



---

# Cables



System Administrator's Guide

---

## NOTICE

Yrrid Software, Inc. (Yrrid) has written this document for use by its staff, customers and prospective customers. No part of the information contained in this document shall be reproduced without written approval of Yrrid.

Yrrid reserves the right to change the information contained in this document without notice.

THE TERMS AND CONDITIONS GOVERNING THE LICENSING OF YRRID SOFTWARE CONSIST SOLELY OF THOSE SET FORTH IN THE WRITTEN CONTRACTS BETWEEN YRRID AND ITS CUSTOMERS. IN NO EVENT SHALL YRRID BE LIABLE FOR ANY INCIDENTAL, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES WHATSOEVER ARISING OUT OF OR RELATED TO THIS DOCUMENT OR THE INFORMATION CONTAINED IN IT, EVEN IF YRRID HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Cables is a trademark of Yrrid Software, Inc.

Legacy Objects Framework and Legacy Object Framework are trademarks of Yrrid Software, Inc.

NeXT, NEXTSTEP and OPENSTEP are registered trademarks of NeXT Software, Inc.

Digital Librarian, and Workspace Manager are trademarks of NeXT Software, Inc.

WebObjects and Mac OS are registered trademarks of Apple Computer, Inc.

Microsoft and MS-DOS are registered trademarks of Microsoft Corporation.

Windows NT is a registered trademark of Microsoft Corporation.

DEC is a registered trademark of Digital Equipment Corporation.

DASHER is a registered trademark of Data General Corporation.

IBM is a registered trademark of International Business Machines Corporation.

WYSE is a registered trademark of Wyse Technology.

Wyseword, WyseWorks, WY-60, WY-50, and WY-50+ are trademarks of Wyse Technology.

WordStar is a registered trademark of MicroPro International Corporation.

All other trademarks mentioned belong to their respective owners.

Manual version 2.0, July 1999.

Copyright © 1995-1999 by Yrrid Software, Inc. All Rights Reserved.

---

---

Cables  
by Christopher Lloyd

Thanks to:  
Yrrid staff,  
Our customers.

Technical writer: Shambhunath Borah  
Editor: Robert Niedbalski  
Information and Cables logo by Christopher Lloyd

---

---

---

---

About this Manual 1-1  
Introduction 2-1  
Getting Started 3-1  
Preferences 4-1  
Launch Preferences 4-2  
General Preferences 4-3  
Quit Preferences 4-4  
Keyboard Preferences 4-5  
Using Cables 5-1  
Standard commands 5-1  
Session 5-3  
Control 5-5  
Windows 5-6  
Configuration 6-1  
Inspector 6-2  
Emulation 6-3  
Connection 6-8  
Display Attributes 6-12  
General Attributes 6-19  
File Transfer Protocols 6-28  
Session Manager 6-35  
Transfer 6-38  
Key Map 6-42  
Advanced Cables Configuration 7-1  
Configuring Cables defaults 7-1  
Partial configuration files 7-4  
Cables Light 8-1  
Configuring Cables Light 8-2  
Copilot Language 3  
Copilot Script Library Panel 4  
Defining a procedure 6  
Statements 7  
Built-in operations 14  
Exceptions 15  
Macro Syntax 17  
IBM 3270 Key Macros 20  
DEC VT320 A-1  
Emulation Attributes A-1  
Key Macros A-4  
Key Panels A-5  
References A-6  
IBM 3270 B-1  
Emulation Attributes B-1  
Copy & Paste Options B-3  
IND\$FILE Transfer B-5  
Key Panels B-10

IBM 5250 C-1  
Emulation Attributes C-1  
IBM 5250 Edit C-2  
Key Panels C-3  
References C-3  
ANSI-PC D-1  
Emulation Attributes D-1  
References D-3  
DG D215 E-1  
Emulation Attributes E-1  
Key Macros E-2  
Key Panels E-3  
References E-3  
TI 931 F-1  
Emulation Attributes F-1  
Key Macros F-2  
Key Panel F-3  
References F-3  
WYSE WY-60 G-1  
Emulation Attributes G-1  
Screen G-3  
Tabs & Labels G-5  
Key Macros G-6  
Key Panels G-8  
Assigning Line Attributes to Colors G-9  
References G-9



---

# 1     *About this Manual*

---

The *Cables System Administrator's Guide* can help readers configure and use Cables terminal sessions. It presumes a reasonably good understanding of the *Mac OS X Server*® or *Windows NT*® user interface.

Although this guide does describe how Cables supports attributes of specific terminal emulations, it deals mostly with the Cables terminal and communications application itself. It does not detail the functional properties of the proprietary terminals; for that you must consult the appropriate terminal's reference manual.

This manual is organized as follows:

- The *Introduction* gives a brief overview of Cables and its features.
- *Getting Started* walks through a simple example of how to configure and save a Cables session.
- *Preferences* describes how to configure the Cables user preferences.
- *Using Cables* describes Cables' commands to open, close, save, and control terminal sessions.
- *Configuration* details the configuration options Cables supports for terminal sessions.
- *Advanced Cables Configuration* describes advanced system administration topics.
- *Using Cables Light* describes how to configure and use the simplified, pre-configured version of Cables.
- *Copilot* describes the Copilot scripting facility.
- The appendices describe the configureable attributes supported on each of Cables terminal emulation.



---

## 2 *Introduction*

---

Cables is a terminal emulation and communications application for Mac OS X Server environment and for the WebObjects® environments under Windows NT and other platforms.

Cables is a unified application which supports multiple concurrent sessions of different terminal emulations and connectivity to a wide range of platforms. Among the industry-standard terminals Cables emulates are:

- DEC VT320, VT220, VT102, VT52
- IBM 3279, 3278
- IBM 3270(EPI)
- IBM 5292, 5291, 5251, 3180, 3179
- WYSE WY-60/50+
- DG Dasher D215
- ANSI-PC
- TI 931

In addition to being highly configureable and easy to administer, Cables is also extremely easy for users to learn.

Cables' main features are:

### **Standard Interface**

Cables' object-oriented design lets you make use of many standard user interface features. You can:

- copy text and graphics to and from Cables using Cut, Paste, and Copy,
- use inspectors to customize attributes of the current terminal session,
- set printing parameters such as margins, black and white/color, page layout,
- print the output from the entire terminal, a selected portion of the terminal, or the visible portion of the screen (all emulations); or print to an attached printer (DEC VT320, WYSE WY-60/50+, and TI 931 only),
- configure window attributes -- titles, resizing and closing behaviors,
- resize the terminal window changing the number of rows and columns or rescaling the font,
- set the color of each terminal's foreground, background, and text attributes,
- configure scroll, drag and drop, cut and paste behavior,
- configure emulation specific parameters,
- load and store custom configurations as .cable files,
- autolaunch configurations on Cables start-up.

---

## Scripting Language

Cables provides a powerful Copilot scripting language to automate common tasks. You can:

- use a *Watch Me* facility to automatically generate scripts,
- start a script automatically after the terminal connects,
- map scripts to keys,
- run multiple scripts in parallel,
- program procedures, timeouts, asynchronous handlers, and exception processing,
- call built-in procedures such as `alert-panel()`, and `system()`.

## Keyboard and function key support

Cables provides extensive support for keyboard customization. You can:

- map keys from the emulated terminal's keyboard onto your machine's keyboard using a graphical interface,
- remap keys to arbitrary string values or macros,
- configure pop up keyboard panels to provide keys unique to the emulated terminal's keyboard.

## File transfers

Cables supports the following file transfer protocols:

- Xmodem, Ymodem, and Zmodem,
- Kermit,
- IBM IND\$FILE (IBM 3270 only).

You can specify files to transfer by "dragging and dropping", from an inspector, or from a Copilot script.

## Communications

Cables allows you to connect a terminal to:

- remote hosts through any serial port,
- a specified shell or command,
- IBM hosts through TCP/IP (IBM 3270 and IBM 5250 only).

## Administration

- Users can share Cables configurations. Anyone can create libraries of Cables configurations which can be used at the owner's discretion.
- You can configure Cables into a more user-friendly version called Cables Light.
- Cables does not require a license server.

## Remote API

The Cables Remote API is available in the YrridDeveloper package. It allows you to:

- establish a connection to the Cables application and open configurations,

- 
- take a snapshot of a terminal's active screen area and examine the contents:  
Unicode encoded characters and the associated attributes (bold, underline, reverse, etc.),
  - get the contents of emulator defined fields, attributes, screen location and size,
  - transmit text, control codes, and macros to a Cables terminal,
  - execute Copilot scripts on a Cables terminal,
  - write Copilot scripts that notify your application of events.



---

## 3 *Getting Started*

---

If you've installed Cables for the first time, you'll need to know how to create a new session, change an important attribute like the emulation, optionally set up an automatic connection to a host, and save the configuration so that it can be used later. Mac OS X Server users can launch Cables.app double-clicking on the Cables.app entry in the Workspace File Viewer or on the Cables icon in your application dock. Click on the session item of the *Cables* menu and then the *New* item in the *Session* menu.

Cables	Session
Info ▾	Open... o
Session ▾	New n
Edit ▾	Save s
Format ▾	Library ▾
Tools ▾	Save As... S
Control ▾	Save To...
Windows ▾	Load...
Print... p	Revert to Saved
Services ▾	Close
Hide h	
Quit q	

A new terminal window will appear. The terminal window will have the following default title:

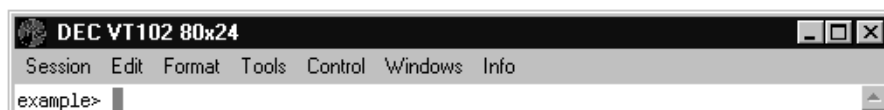


The new terminal session is, by default, an 80 column, 24 line VT102 connected to a UNIX C-shell (/bin/csh).

Windows NT users can launch Cables.app by clicking on *Start -> Programs -> Yrrid-LegacyObjects -> Cables*. This should bring up a VT102 Cables Terminal window, and an *Enter Hostname* panel. If you want to use the VT102 terminal, simply enter the name of the host you would like to connect to via TCP/IP. .



Once connected (to the host **example**), the terminal window will have the following default title:



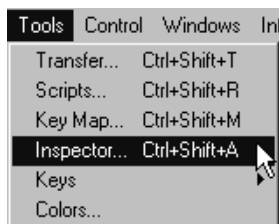
To create a new session, click on the *Session* item of the *Cables* main menu and then the *New* item in the *Session* menu.



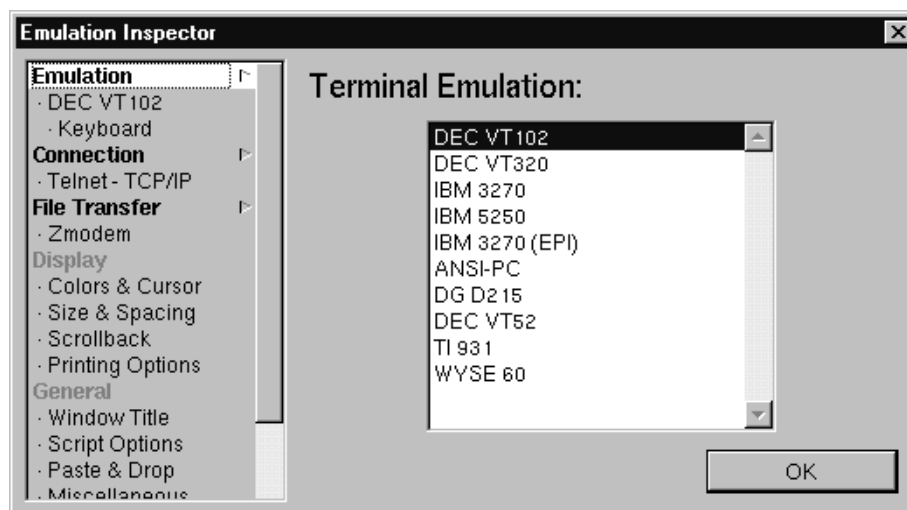
A new terminal window will appear along with an *Enter Hostname* panel. The new terminal session is, by default, an 80 column, 24 line VT102.



To select a different emulation or connection option, click *Cancel* in the *Enter Hostname* panel, and then click on *Tools->Inspector...*

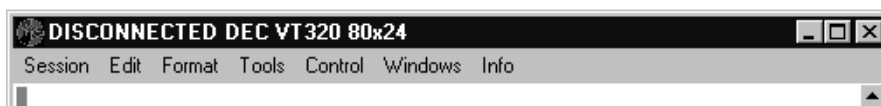


The *Inspector* will appear. Click on *Emulation* in the *Inspector* directory (the left scrollview); the *Emulation Inspector* will appear with the current emulation selected:

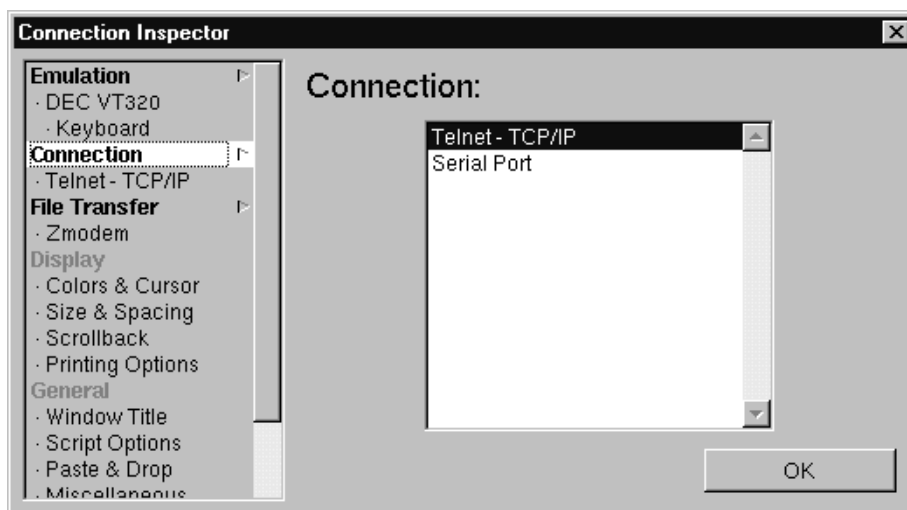


Note that these inspectors always focus on the current or key session. Right now the *Emulation Inspector* is inspecting your new VT102 session. Select the terminal emulation you want by clicking the cursor on the terminal in the right scrollview and pressing the *OK* button.

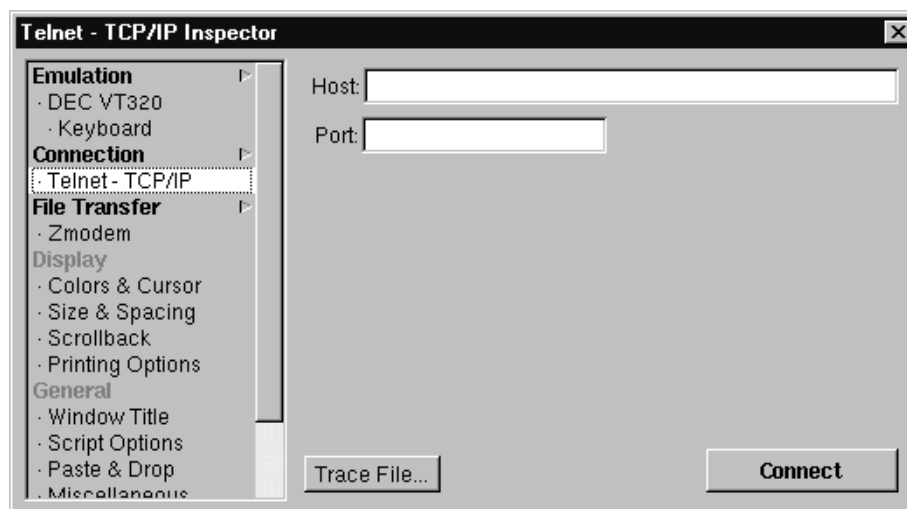
For this example, we've selected VT320. The *VT320 Inspector* should now be visible, and the current terminal session, which is also the key window, becomes the selected emulation:



This VT320 terminal session is still disconnected. Suppose you want this terminal session to connect to a particular host via TCP/IP. Click on *Connection->* in the *Inspector* directory to display the *Connection Inspector*.



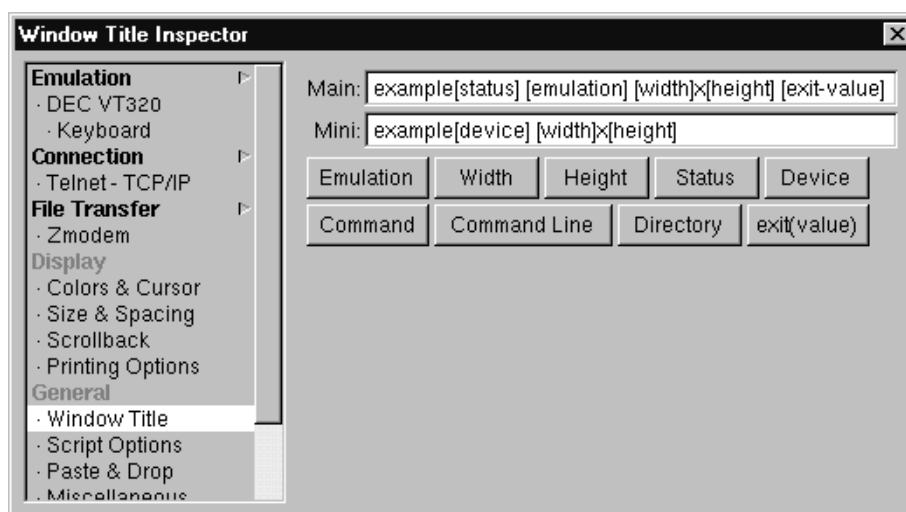
Click on *Telnet - TCP/IP* to display the *Telnet - TCP/IP Inspector*:



Click in the *Host* text field and enter the hostname. If you want to connect to a specific port number, enter it in the *Port* text field. Click the *Connect* button.

Before you save this terminal session you will probably want to add the hostname to the window title. To do this click on *Tools-> Inspector* and select *Window Title* to display the *Window Title Inspector*. Click in the *Main* and *Mini* text fields and add the hostname to the window titles.

The *Window Title Inspector* will now look like this:



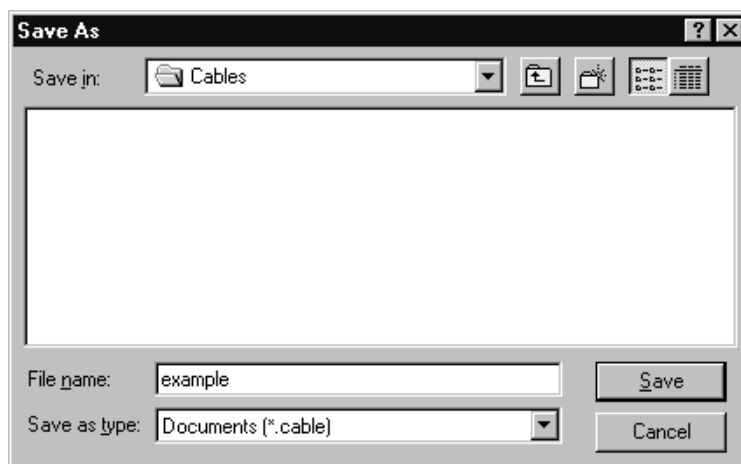
The main window will look like this:



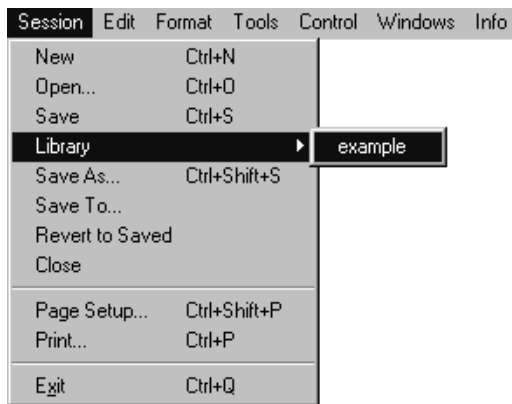
When you miniaturize the window it will now look like this:



To save this configuration for later use, click *Session->Save As...* A *Save* panel will appear. Use this browser to find the directory to save the configuration. Unless you're grouping configurations in special directories, the default directory *~/Library/Cables* is as good a place as any (~ indicates your home directory). Enter the configuration name in the *Name* text field and press Return.



In order to open a saved configuration in the directory ~/Library/Cables click on *Session->Library*. The configurations you've saved will appear in the library menu:



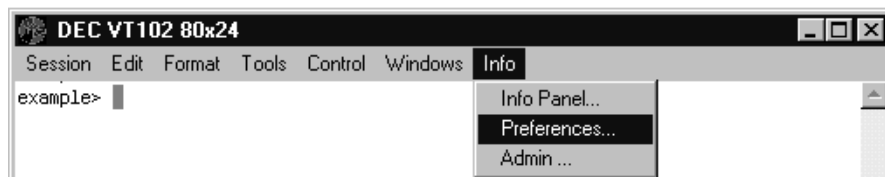
To open a terminal session with attributes of the saved configuration, click on the configuration name **example**.

You can also open a configuration as a .cable file by clicking on *Session->Open...* or by double-clicking on a .cable file in the Windows NT Explorer.

---

## 4 *Preferences*

---



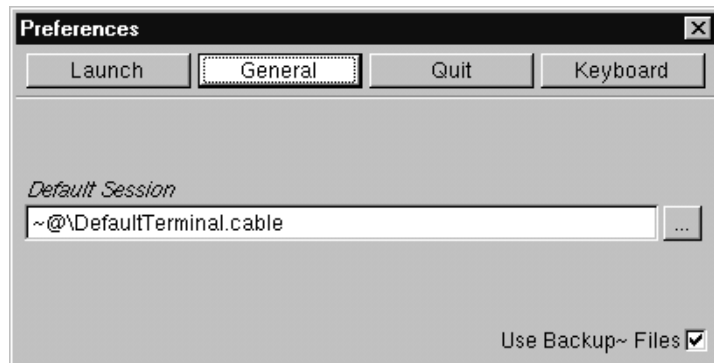
*Info->Preferences...* allows you to change Cables' general behavior for your account. Cables stores these preferences in each user's defaults database. You can specify, for example, what sessions Cables will open when you launch it, what additional directories are in the Cables PATH environment variable, what session Cables will open for the *Session->New* command, which terminal the Workspace Manager and ProjectBuilder will use as their defaults, and how Cables responds to the *Quit* command.

## Launch Preferences



In Windows NT, when you launch Cables, Cables, by default, opens an unconnected VT102 session. If you click the *Add...* button, an *Open at Launch* panel will appear. Use this directory browser to select the .cable files that Cables will open when you launch it. Note that if you've just installed Cables for the first time, the only configuration you have is the default configuration, `~@/DefaultTerminal.cable` (`~@` indicates the Cables.app directory of the currently executing Cables application). The chapter *Getting Started* gives an introductory example of how to configure a session; the chapter *Configuration* details the non-emulation-specific configuration options.

## General Preferences



*General Preferences* allows you to change the default session Cables opens when you use the *Session->New* command. The default session, ~@\DefaultTerminal.cable, is a VT102 (80 columns by 24 lines) with the TCP/IP option selected.

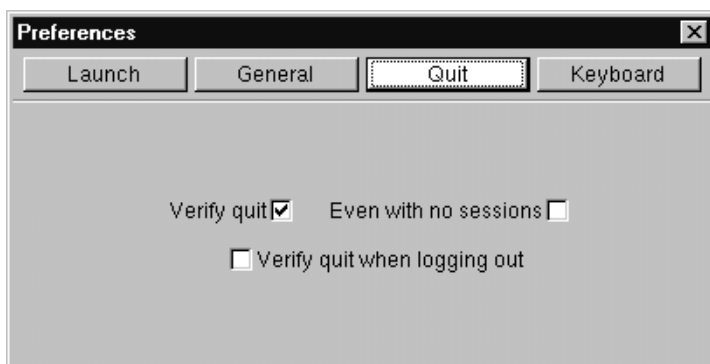
### Default Session

Enter the pathname of the .cable file you want as the default session. Clicking on the ... button brings up an *Open* panel. You can use this directory browser to select a *Default Session* .cable file.

### Use Backup~ Files

If you check the *Use Backup~ Files* box, then whenever you save a configuration, Cables will back up the previous .cable configuration file. For example, if you open the .cable file **just-the-vax.cable**, change its attributes and save it, then Cables saves the previous configuration in the file **just-the-vax.cable~**.

## Quit Preferences



If you issue the quit command and still have sessions open, Cables will, by default, prompt you with the following *Alert* panel:

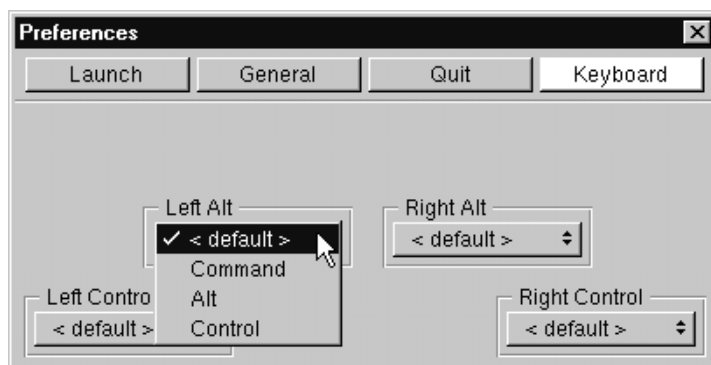


If you want a similar *Alert* panel even when there are no open sessions, then check the *Even with no sessions* box. Now when you close the last Cables session, you will see the *Really Quit?* panel. If you press the *Cancel* button, Cables will still remain running. Now, to open a new session, you must hit **Ctrl-N**. Note: You must first make sure that Cables is the active application. In Windows NT, you can do this by clicking on a Cables inspector window, or by clicking *Start -> Programs -> Yrrid - LegacyObjects -> Cables*.

If you want a similar *Alert* panel when logging out, then check the *Verify quit when logging out* box. If you do not want any *Alert* panels, then uncheck all boxes.



## Keyboard Preferences



You can set the behavior of the left and right Alt- and Ctrl-keys. Initially set to their default values, these keys can be set to be the Alt-key, the Ctrl-key or the Command-key. This feature allows you to approximate the behavior of the Mac OS X Server or OPENSTEP user-environments by creating a key like the Cmd-key in OPENSTEP.

In Windows NT, the Ctrl-key is used to trigger menu shortcuts. For example, pressing Ctrl-N usually creates a new document for a Windows program. To set the left Alt-key to trigger menu shortcuts, choose *Command* from the browser list in the *Left Alt* box. Save the session. When you relaunch Cables, the Alt-key can now be used to trigger menu shortcuts. So in Cables, you can now open a new session by pressing Alt-N.

If you want, you can reassign the Ctrl-key to be the Alt-key, etc.



---

## 5 *Using Cables*

---

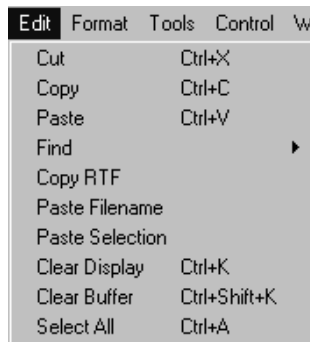
This chapter describes Cables' standard Windows NT commands, how to initiate and save sessions, how to control the state of a session, and describes the behavior of miniwindows.

### Standard commands

Cables supports the following standard commands:

- Edit
- Format
- Windows
- Print
- Quit

#### Edit



The *Cut*, *Copy*, *Paste*, *Find*, and *Select All* commands are standard commands. The following are specific to Cables:

#### Copy RTF

Copies the selected text with Rich Text Formatting. You can paste the selection into an application like **TextEdit**.

#### Paste Filename

Pastes the filename and path you have copied from the *Mac OS X Server File Viewer* or the *Windows NT Explorer*.

### **Paste Selection**

Takes text selected in a Cables window and pastes it at the prompt in the same Cables window.

### **Clear Display**

Empties the scrollback buffer and clears the screen.

### **Clear Buffer**

Empties the scrollback buffer, but does not clear the screen.

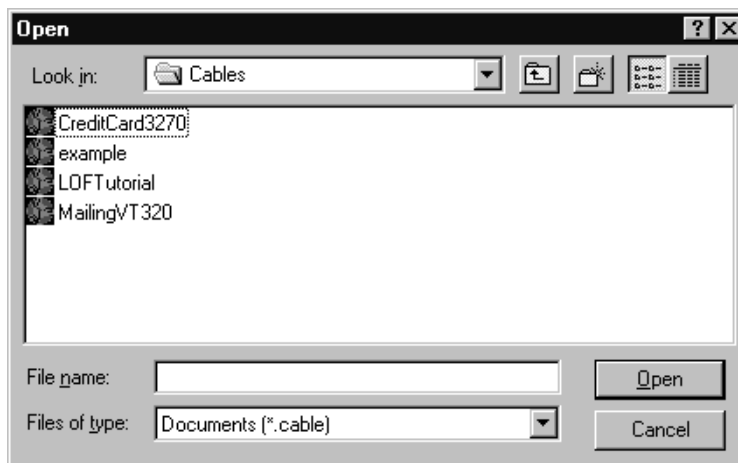
# Session



## Initiating sessions

### Open

*Open* allows you to create a session using any Cable configuration file you can read. Clicking on *Open* brings up an *Open* panel. Use the *Open* panel's browser to locate the Cable file you want to open:



### New

*New* opens a new session using the *Default Session* configuration specified in the *Info->Preferences->General* panel.

**NOTE:** If you use *New* to create a new default session, alter the session's configuration and *Save*, then Cables will bring up the *Save As* panel which you can use to specify the name of the new configuration.

### Library

Selecting *Library* brings up a menu of configuration files. Click on a *Library* menu item to start a session. To access Cables sessions through the *Library* menu item, the corresponding .cable files should be placed in ~/Library/Cables. You can change the directory location of the *TerminalLibrary* by editing the Cables.defaults or Cables-winnt.defaults file.

**NOTE:** If you use *Library* to open a session, alter its configuration and *Save*, then Cables will not automatically overwrite the .cable file you opened. It will, instead, bring up the *Save As* panel which you can use to specify the name of the new configuration.

## Saving session configurations

### Save

*Save* saves the session configuration in the currently opened Cable file. Cables will automatically save the configuration if you already *Opened* the session or if you used the *Save* panel to save the configuration. If Cables cannot automatically save the configuration, it will bring up the *Save* panel so that you can specify the .cable file name.

### Save As

*Save As* brings up the *Save As* panel so that you can save the session configuration under a different name. Once you have used the *Save As* panel to save the configuration as a .cable file, you can change the session configuration and *Save*, and Cables will automatically overwrite the .cable file.

### Save To

*Save To* is the same as the *Save As* command except that it allows you to keep working with the original configuration.

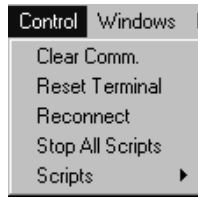
## Revert to Saved

If you *Opened* the session or saved it using the *Save* panel, then *Revert to Save* will reconfigure the current session to its last saved state.

## Close

*Close* terminates the current session. If you checked the *Prompt when closing* box in *Tools->Attributes->Window Attributes* and close a session, then Cables will bring up an *Alert* panel to confirm that you really want to close the session. Clicking *Close* is equivalent to clicking the session window's close button.

# Control



The *Control* menu allows you to control the scrollback buffer, reset the terminal, and control various communication-related parameters of the key terminal.

## Clear Comm.

*Clear Comm.* clears the terminal's internal buffers. This is especially useful when you want to abort a large paste operation.

## Reset Terminal

*Reset Terminal* resets the terminal leaving the terminal disconnected.

## Reconnect

*Reconnect* disconnects the terminal and then connects it again. It is equivalent to clicking the *Disconnect* and the *Connect* button in the *Connection Inspector*.

## Stop all Scripts

*Stop all Scripts* stops all running Copilot scripts.

## Scripts

The *Scripts* menu lists all Copilot scripts in the .copilot files in the *TerminalLibrary* path directories. Select a script to run it in the key terminal session.

# Windows



The *Windows* menu allows you to arrange Cables windows that are open. The list of open Cables windows is at the bottom of the menu. You can select a window from that list to make it the key window.

*Arrange in Front* arranges all your Cables windows in a stacked formation in the center of your monitor.

*Minimize All* minimizes all Cables windows.

*Minimize Window* minimizes the key Cables window.



---

## 6 *Configuration*

---

Each Cables session has its own set of attributes which you can configure to meet your requirements and save in a configuration file. Configuration files have the extension `.cable` and are also called Cable files. By default, Cables saves `.cable` files in the directory `~/Library/Cables` in Mac OS X Server or `~\Library\Cables` in Windows NT (where `~` is the user's home directory). You can, however, save `.cable` files in any directory you have write access to; anyone with read access to them can open them to create sessions. A `.cable` file, **example.cable**, will have the icon:



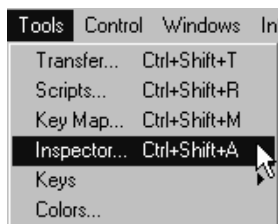
example

When you double-click on a `.cable` file, a session with the configured attributes will open.

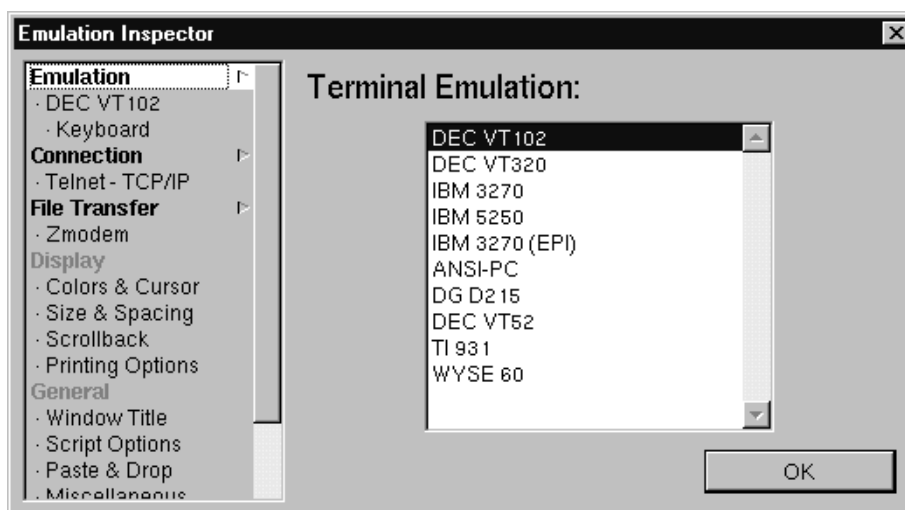
Cables allows you to configure a session by means of the set of inspectors described in this chapter.

**NOTE:** Cables allows a session to consist of more than one terminal. The terminal that can receive keyboard input is called the key terminal.

# Inspector



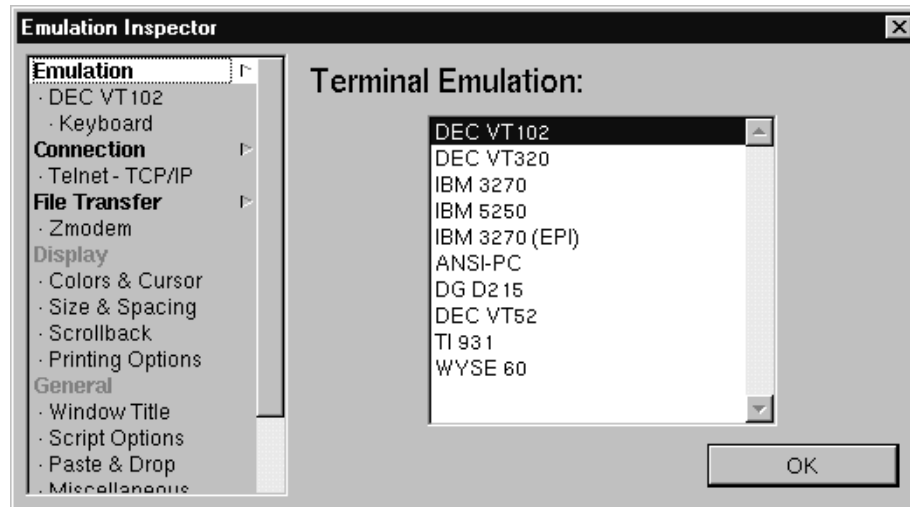
Select *Tools->Inspector...* to bring up an inspector with which you can configure most session options and attributes.



The scrollview on the left is the *Inspector* directory. The items in bold with a right arrow allow you to choose a particular emulation, connection, file transfer protocol, or multi-terminal session style. For example, to change the key terminal's emulation to VT320, you would select *Emulation->* in the *Inspector* directory, then either select *DEC VT320* in the option list (right scrollview) and click *OK* or just double-click on *DEC VT320*.

If you select an *Inspector* directory item preceded by a bullet (•), then the inspector becomes the inspector for that item. For example, select • *DEC VT102* to display the *DEC VT102 Inspector*.

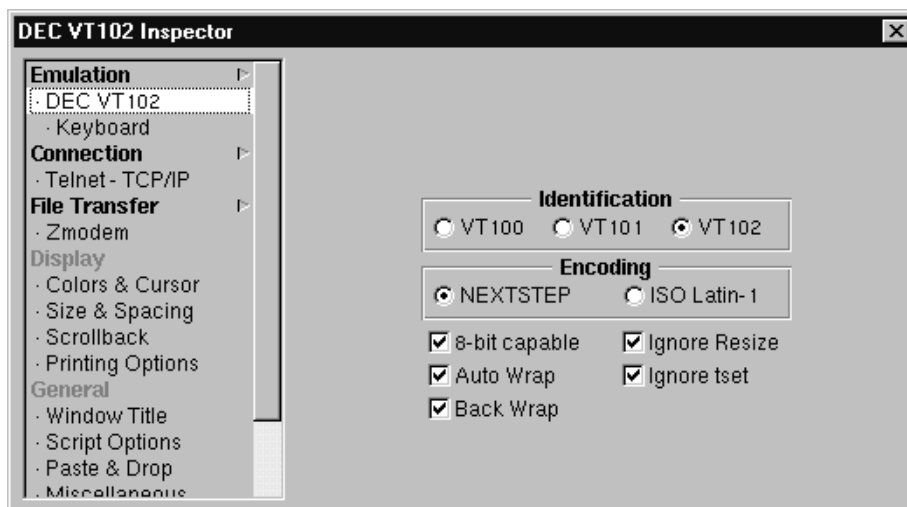
# Emulation



Select *Emulation->* from the *Inspector* directory to display the *Emulation Inspector* which allows you to change the emulation for the key terminal. Double-click on an emulation in the right scrollview to display an inspector which allows you to set emulation-specific attributes. Cables can support the emulations described in the appendices.

## DEC VT102

From the *DEC VT102 Inspector* you can set the following attributes:



### Identification

Select the terminal model identification code you want this emulation to return. The VT102 emulation can identify itself as either a VT100, VT101, or VT102.

### Encoding

Select the encoding you want the terminal to use to translate keycodes from other sources (e.g. from the paste buffer).

- *NEXTSTEP* causes the terminal to map the keycodes in the range 128-255 to the *NEXTSTEP* character encoding.
- *ISO Latin 1* causes the terminal to map the keycodes in the range 128-255 to the *ISO Latin 1* character encoding.

### 8-bit capable

Normally the VT102 terminal supports only 7 bit mode. *8-bit capable* allows a Cables VT102 terminal to display keycodes using the entire 256 value encoding.

### Auto Wrap

Checking *Auto Wrap* causes text entered past the last column to wrap onto the next row.

If you don't check *Auto Wrap*, then, when text is displayed past the last column of the current row, the cursor will stop at the last column; each new character will overwrite the last one.

## Back Wrap

If you check the *Back Wrap* option and use the Delete key to delete text that has wrapped over multiple rows, Cables will backtrack the cursor over the wrapped line. If you do not check *Back Wrap*, then the Delete key will still delete text, but the cursor will stop at the first column of the current line.

Cables selects *Back Wrap* as the default because it is a generally desirable feature (which might not have been supported in the emulated terminal).

Note that *Back Wrap* only affects the behavior of backspace (BS) keycode.

## Ignore Resize

Checking *Ignore Resize* causes the terminal to ignore resize commands from the application.

## Ignore tset

Checking *Ignore tset* causes the terminal to ignore the UNIX **tset** command from the application (**tset** causes the screen to resize).

## Keyboard

Selecting *Emulation/Keyboard* from the *Inspector* directory displays the *Keyboard Inspector*. This inspector allows you to change the mapping of the Cursor, Keypad, the Delete, Return, and Meta Keys, and to edit the answerback string.

### Cursor Keys

Unchecking the *Cursor Keys* box passes the cursor keycodes on to the application *without* translating to a VT series escape sequence.

Checking the *Cursor Keys* box enables one of the following options:

- *Normal* converts the cursor keycodes to the VT series “normal” escape sequence.
- *Application* converts the cursor keycodes to the VT series “application” escape sequence.

### Keypad Keys

Unchecking the *Keypad Keys* box passes the keypad keycodes on to the application *without* translating to a VT series escape sequence.

Checking the *Keypad Keys* box enables one of the following options:

- *Numeric* converts the keypad keycodes to the VT series “numeric” representation.
- *Application* converts the keypad keycodes to the VT series “application” escape sequence.

### Meta Keys

Unchecking the *Meta Keys* box makes the Alternate key function normally.

Checking the *Meta Keys* box enables one of the following options:

- *Escape* causes the Alternate key to function as the EMACS meta key. If you hold down the Alternate key and press, for example, the **a** key, Cables will send the ESC (escape) character (0x1B) and then the **a** character (0x61).
- *High Bit* causes the Alternate key to set the high bit in the character code. If you hold down the Alternate key and press, for example, the **a** key, Cables will send the code 0xE1.

### Delete

Unchecking the *Delete* box maps the Delete key to its value in the Keyboard mapping.

Checking the *Delete* box maps the Delete key to:

- *DEL* the DEL (delete) character (0x7F).
- *BS* the BS (backspace) character (0x08).

### Return

Unchecking the *Return* box maps the Return key to its value in the Keyboard mapping.

Checking the *Return* box maps the Return key to:

- *CR* the CR (carriage return) character (0x0D).

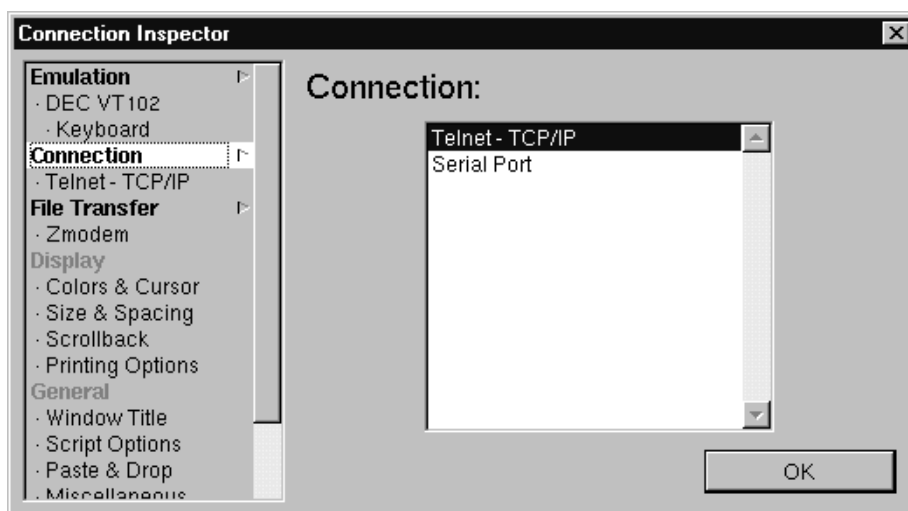
- *CR-LF* both CR and LF (line feed) characters (0x0D and 0x0A)

## Answerback

The DEC VT series terminals send the answerback string back to the host in response to the host sending the ^E (control-E) command.

Click in the answerback text field to add or edit the answerback string.

## Connection

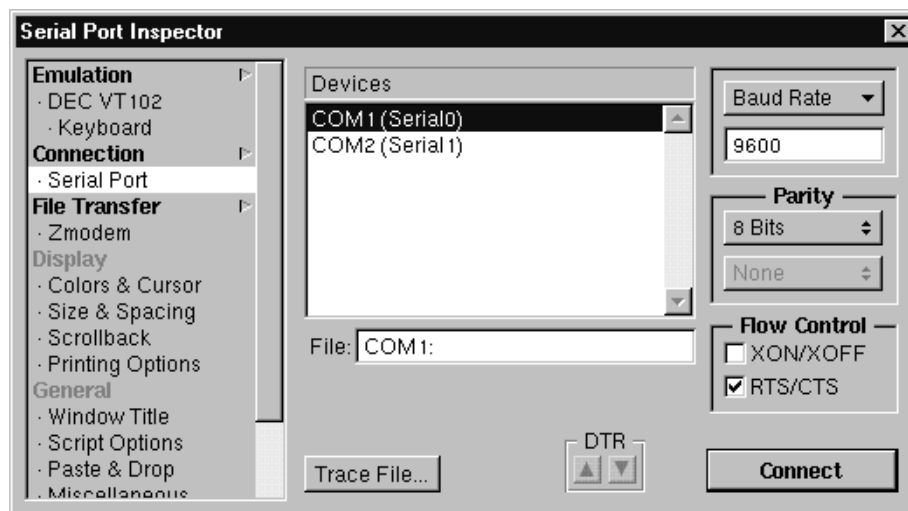


Selecting *Connection*-> from the *Inspector* directory displays the *Connection Inspector*. The *Connection Inspector* controls a terminal's communication attributes. With it you can change how the terminal is connected: to a telnet client or to a serial port.



## Serial Port

Select *Connection* -> *Serial Port* from the Inspector Directory to display the *Serial Port Inspector*.



This option allows you to connect the terminal to one of your host's serial devices. In most cases this port is, in turn, connected to a modem. You can also set the serial port related characteristics of the terminal.

### Devices

To select a device, either click on the device description in the *Devices* text box or click in the *File* text field and enter the pathname of the asynchronous device.

### Baud Rate

In order to select a different baud rate, either drag down the *Baud Rate* option list and release on the number you want, or click on the text field and explicitly enter the number.

### Parity

Set the number of data bits by dragging down the upper option list. If you choose *7 Bits*, then select the parity by dragging down the lower option list.

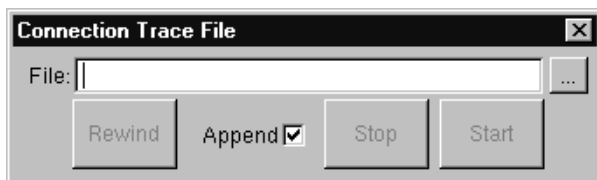
### XON/XOFF

Check XON/XOFF if you want the connection to use XON/XOFF (software) flow control.

### Trace File

Click on the *Trace File...* button to bring up the *Connection Trace File* inspector:

The *Connection Trace File* inspector allows you to record the data transmitted from and received by the terminal. To select the file you want Cables to record to, you can either enter the pathname in the text field directly or click the ... button to bring up the *Session Recording* panel. If you check the *Append* box, Cables will append the terminal's data to the end of the selected file, otherwise the file will be overwritten each time you



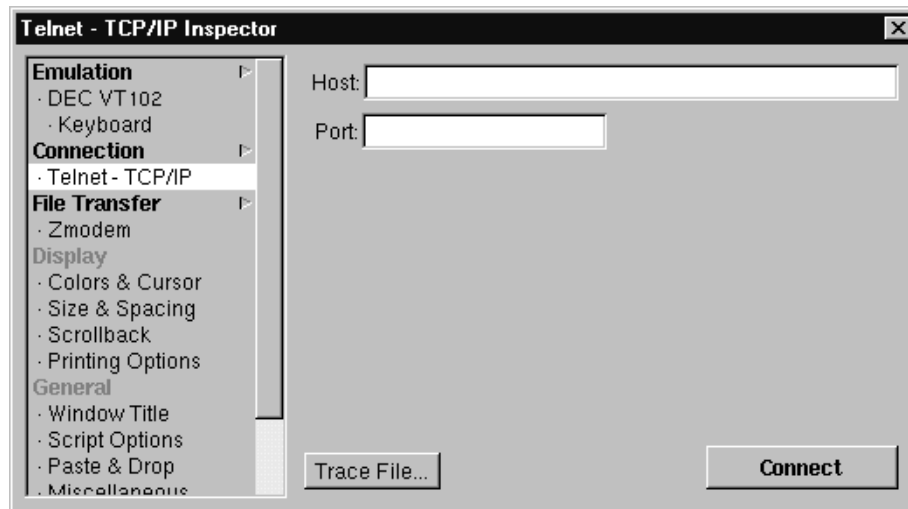
start recording. Click the *Start* button to start recording. Once recording you can click the *Rewind* button to delete the old file's contents. Click the *Stop* button to stop recording. The resulting file contains encoded diagnostic data that you can include with your problem report if one of your terminals is not behaving as it should. Press the *Save Screen Snapshot* button to save the current screen contents in a .snapshot file.

## DTR



Click the UP arrow to raise DTR; click the DOWN arrow to drop DTR.

## Telnet - TCP/IP



Selecting *Connection/Telnet - TCP/IP* option from the *Inspector* directory allows you to connect your terminal to a remote host.

### Host

Click in the *Host* text field to enter the name or the Internet address of the remote host you want this terminal to connect to.

### Port

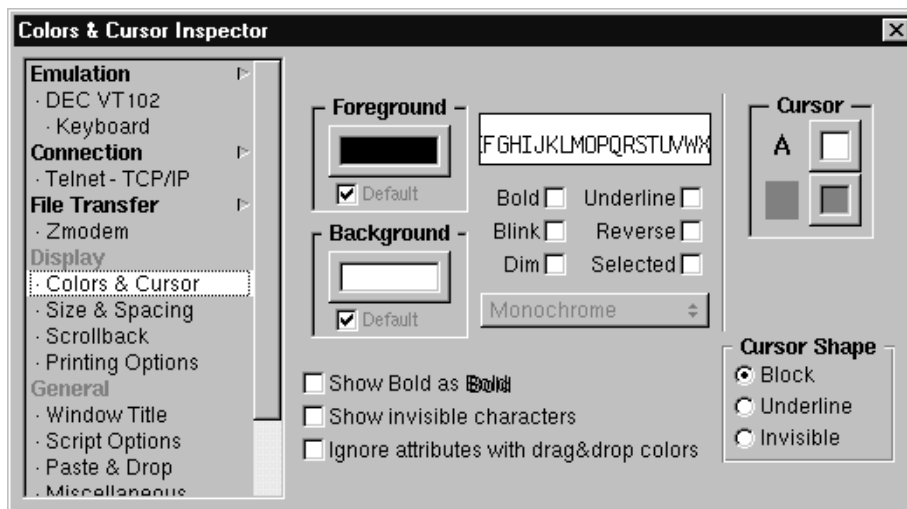
In most cases you'll use the default telnet service to select the port. If you need to use a different service, enter either the service name or the port number.

# Display Attributes

This section of the *Inspector* directory allows you to choose Color & Cursor options, Size & Spacing options, Scrollback options, and Printing Options.

## Colors & Cursor Inspector

Selecting *Display/Color & Cursor* from the *Inspector* directory displays the *Colors & Cursor Inspector*.



This inspector allows you to change the foreground and background color of text with any combination of attributes: bold, underline, blink, reverse video, dim, and selected. It also allows you to specify the color of the cursor.

Note that the *Foreground* and *Background* color wells apply to text with the attributes in the middle check boxes. If none of the attribute check boxes are checked, then the foreground and background colors apply to regular, non-bold, non-underlined, non-blinking, foreground-on-background, non-dimmed, non-selected text. If, for example, you've checked the *Dim* and *Underline* boxes, then the foreground and background colors apply to dimmed, underlined text.

There are 3 ways to change the color of attributed text:

1. Bring up the *Colors* palette. Drag and drop a color into the appropriate *Colors & Cursor Inspector* color well.
2. Bring up the *Colors* palette. Drag and drop a color into the position of the text in the terminal window.
3. Click on the border of a color well. This will bring up the *Colors* palette. As soon as you select a color from the palette, it will be reflected in the terminal.

Cables will display the new color not only in the color well, but also in the text box in the upper center of the inspector. The text box displays how the attributed text will look with the new foreground and background colors. This is generally helpful since the terminal may not be currently displaying text with the same attributes.

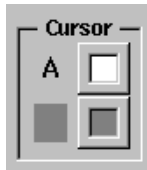
When you check the *Default* check boxes under the *Foreground* and *Background* color wells, Cables automatically resets the foreground and background colors for any combination of attributes.

**NOTE:** The *Default* check boxes are not enabled for text without attributes.

A color terminal such as ANSI-PC or IBM 3279 can be either in monochrome or color mode. When it is in monochrome mode, the pop-up option list will be disabled at the item *Monochrome*; you can set the foreground and background colors for any attribute combination just like you would for a real monochrome terminal.

When the color terminal is in color mode, you can redefine colors for all combinations of attributes for each color that the terminal displays. Select the color from the pop-up option list, select the set of attributes, and change the foreground and background colors.

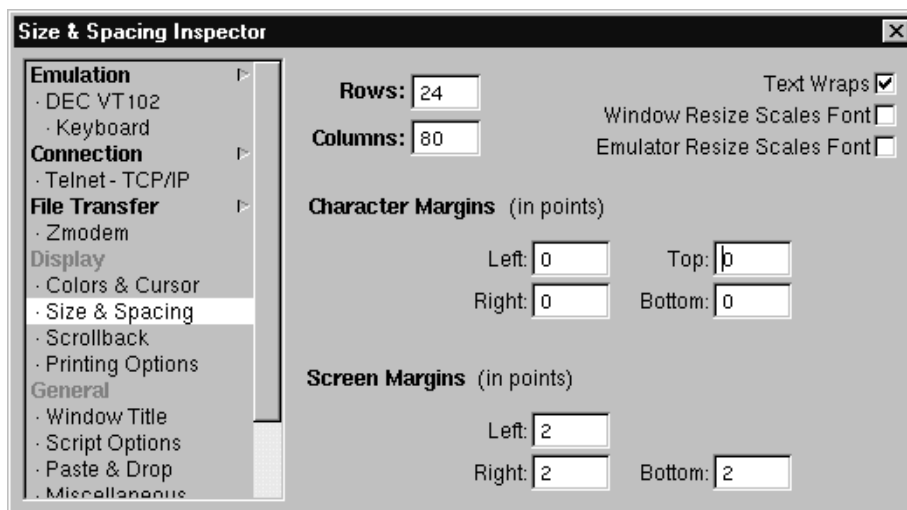
## Cursor



The bottom color well in the *Cursor* box allows you set the color of the cursor. The top color well is the color of the character that shows through the cursor when the cursor is positioned on it.

## Size & Spacing Options

Selecting *Display/Size & Spacing* from the *Inspector* directory displays the *Size & Spacing Inspector*



This inspector allows you to specify spacing of the character and screen margins, the size of the window, and the response to resizing commands from the emulator or from the user.

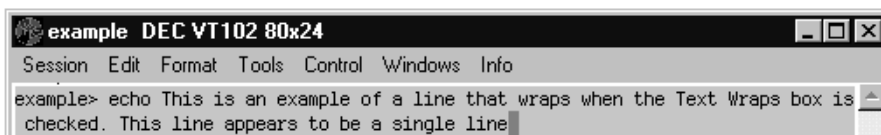
### Rows & Columns

To change the number of rows or columns, simply click in the text fields next to *Rows* or *Columns* and enter the numbers.

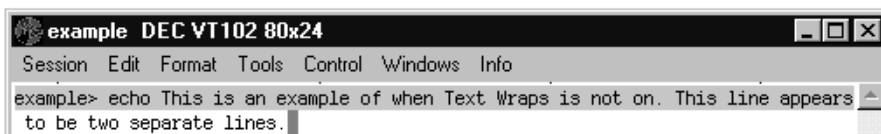
### Text Wraps

If you check the *Text Wraps* option, Cables wraps lines that extend beyond the last column of the screen. If you don't check the *Text Wraps* option, then Cables will break the line at the last column of the row.

For example, if you type a line that wraps and *Text Wraps* is on and triple-click on a part of it, then Cables will select the entire line:



If you do not turn on *Text Wraps* and triple-click on a part of it, then Cables will only select the line you triple-clicked on:



## Window Resize Scales Font

If you check the *Window Resize Scales Font* box, then, when you resize the window, Cables maintains the number of rows and columns in the window scaling the font accordingly.

**NOTE:** This attribute has no effect on a terminal in a multi-terminal session.

## Emulator Resize Scales Font

If you check the *Emulator Resize Scales Font* box, then, if the application sends an emulation-specific resize command to the terminal, the terminal will resize the font to keep the window approximately the same size on the screen.

## Character Margins

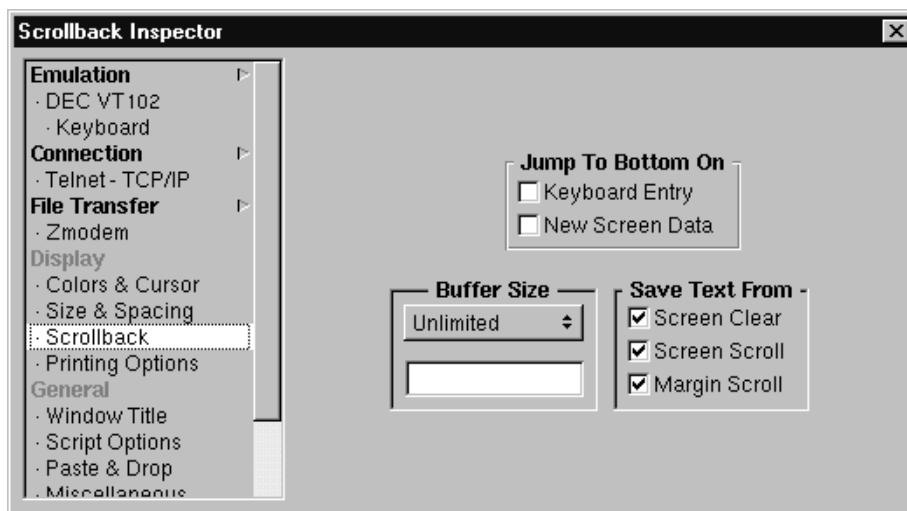
To set the character margins to the left, right, above or below the character, enter the appropriate size, in points, in the text fields *Left*, *Right*, *Top*, and *Bottom*.

## Screen Margins

To set the margins between the characters and the screen, enter the appropriate size, in points, in the text fields *Left*, *Right*, and *Bottom*.

## Scrollback Options

Selecting *Display/Scrollback* displays the *Scrollback Inspector*.



The *Scrollback Inspector* allows you to control the amount of text saved in scrollback buffer and which events cause text to be saved and which events cause the scrollbar to jump to the bottom.

### Jump to Bottom

*Jump to Bottom* allows you to change the behavior of the scroll bar so that it jumps to the bottom of the terminal when there is *Keyboard Entry* of new data (typing or pasting) or when some process writes *New Screen Data* to the terminal.

### Buffer Size

*Buffer Size* allows you to specify how much text the terminal saves in its scrollback buffer.

- *Unlimited*                      The terminal saves all the text it receives.
- *Lines*                          The terminal saves up to the specified number of lines of text.
- *Kilobytes*                      The terminal saves up to the specified number of kilobytes of text. (1024 bytes/Kb)
- *Megabytes*                    The terminal saves up to the specified number of megabytes of text. (1048576 bytes/Mb)
- *Zero*                            The scrollback buffer is disabled.

If, for example, you want to limit the amount of text that's saved in the scrollback buffer to 10 lines, drag the *Buffer Size* button down to the *Lines* option and release.

Now enter 10 in the text field below the *Buffer Size* button and press the Return key. The terminal will now buffer up to 10 lines of text.

### Save Text Events

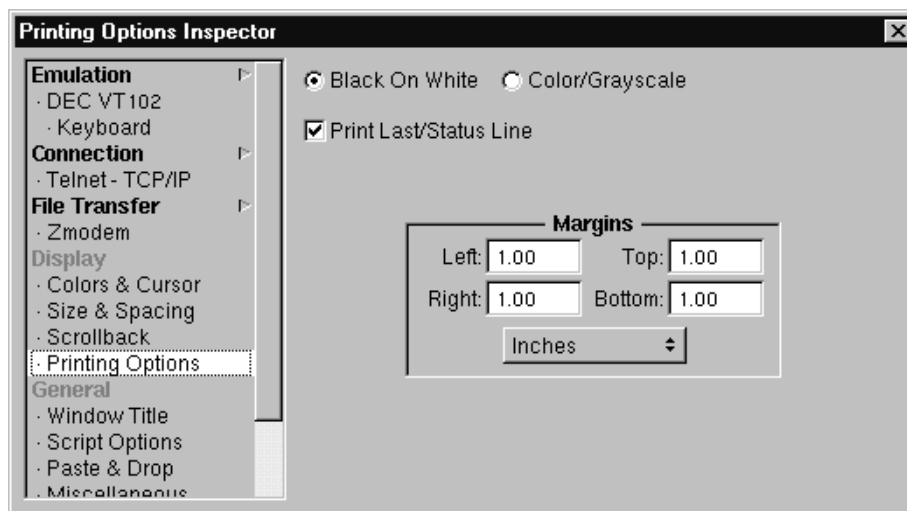
There are 3 events that cause the terminal to save text in its scrollback buffer:



- 
- |                        |   |
|------------------------|---|
| • <i>Screen Clear</i>  | The terminal saves the contents of the screen whenever the screen is cleared. |
| • <i>Screen Scroll</i> | The terminal saves the top line of the screen whenever it's scrolled.         |
| • <i>Margin Scroll</i> | The terminal saves the top line of a window margin whenever it's scrolled.    |

## Printing Options

Selecting *Display/Printing Options* from the *Inspector* directory displays the *Printing Options Inspector*.



This inspector allows you to specify how the terminal's screen contents will look when you print them.

### Colors

- *Black On White* Cables prints the screen contents with a black foreground on a white background regardless of the various foreground and background grayscales or colors on the screen.
- *Color/Grayscale* Cables prints the screen contents in the corresponding grayscale or, if the screen contents are in color and the printer is a color printer, in the corresponding colors.

### Print Last/Status Line

Checking the *Print Last/Status Line* box causes Cables to print the last or status line of the displayed screen. If you do not select this option and you print the entire scrollbar buffer or the visible portion of the screen, then Cables will not print the last line on the screen.

### Margins

*Margins* allows you to set the margins on the pages to be printed.

**NOTE:** Unless you press the Return key after you edit the margins, none of the changes you made will be set for this configuration.

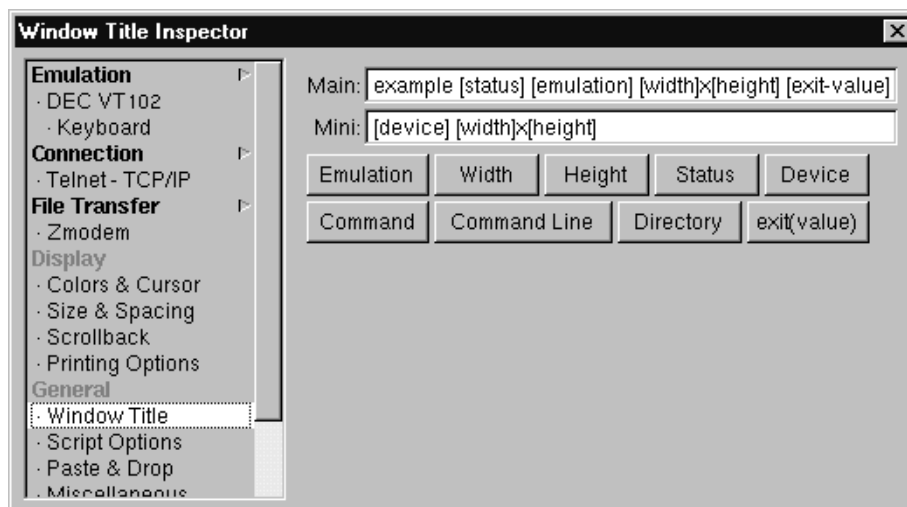
The option list button at the bottom of the *Margins* box allows you to change the units in which the margins are expressed. For example, to choose points, drag the button down to *Points* and release; the size of the margins will now be expressed in points rather than in inches.

# General Attributes

This section of the *Inspector* directory allows you to configure your Window Title options, Script Options, Paste & Drop options, Miscellaneous options, and Print Spooling options.

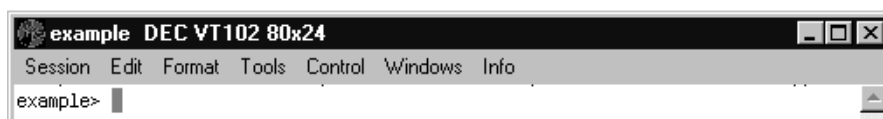
## Window Title Options

Selecting *General/Window Title* displays the *Window Title Inspector*. This inspector allows you to change the visible attributes and behavior of the terminal window.

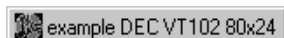


## Titles

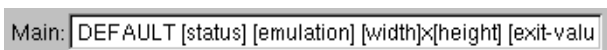
The Main title is the title on the top bar of the main window:



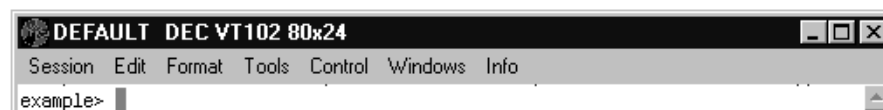
The Mini title is the title on the minimized window. In Windows NT this is a truncated version of the Main title..:



In order to change the window title, click on an insertion point within the text field and edit. For example, if you want to add the string **DEFAULT** to the front of the title of the main window, click at the first position in the *Main* title text field, enter **DEFAULT**, and press the Return key:



As soon as you press the Return key the change will be reflected in the main window title:



The title of the miniwindow can be changed in a similar way except the number of characters that can be displayed on the miniwindow title bar is limited. Always make sure that the miniwindow title is relatively short (~10 characters) and as unique as possible. You should also check that the miniwindow title is what you want by miniaturizing the window after editing its title.

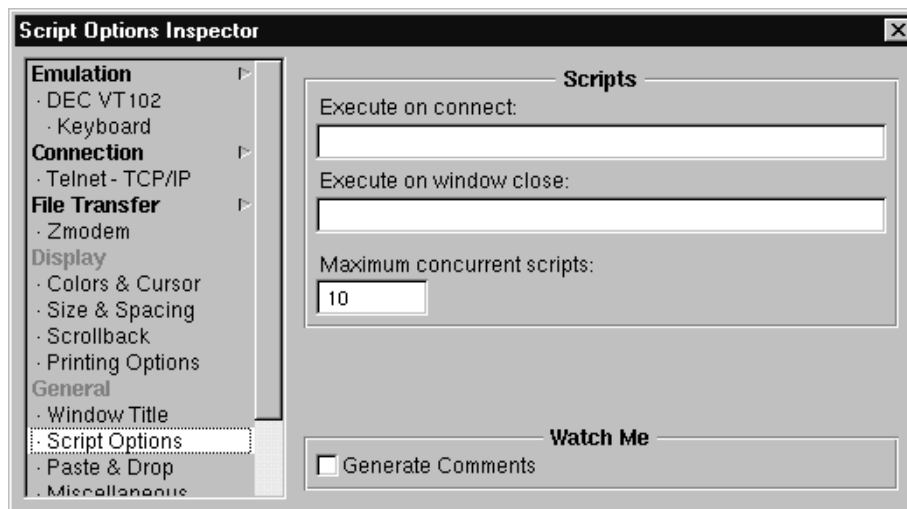
Note that there are a number of buttons under the *Main* and *Mini* title text fields. When you click on one of these buttons, the inspector expands the text representation for the corresponding pre-defined variable at the insertion point in the selected title text field. The following table describes each of the pre-defined variables:

**Table 1:**

Button	Variable	Description
Emulation	[emulation]	terminal model name specified in the <i>Emulation Inspector</i>
Width	[width]	width of screen in columns
Height	[height]	height of screen in rows
Status	[status]	<i>Connection Inspector</i> status Connected:No value Not connected:DISCONNECTED Spooling to printer:SPOOLING File Transferring:TRANSFERRING
Device	[device]	<i>Connection Inspector</i> device <i>Shell /Command</i> :Pseudo-terminal device name <i>Serial Port</i> :Asynchronous device name <i>Telnet - TCP/IP</i> :<tcp/ip>
Command	[cmd]	<i>Connection Inspector-&gt;Shell/Command</i> only basename of argument 0 in the <i>Command</i> text field
Command Line	[cmd-line]	<i>Connection Inspector-&gt;Shell/Command</i> only entire <i>Command</i> text field
Directory	[directory]	<i>Connection Inspector-&gt;Shell/Command</i> only <i>Directory</i> text field
exit(value)	[exit-value]	<i>Connection Inspector-&gt;Shell/Command</i> only exit status value

## Script Options

Select *General/Script Options* from the *Inspector* directory to display the *Script Options Inspector*.



*Execute on connect* allows you to specify what Copilot script Cables executes after it connects the terminal.

*Execute on window close* allows you to specify what Copilot script Cables executes before it closes the window.

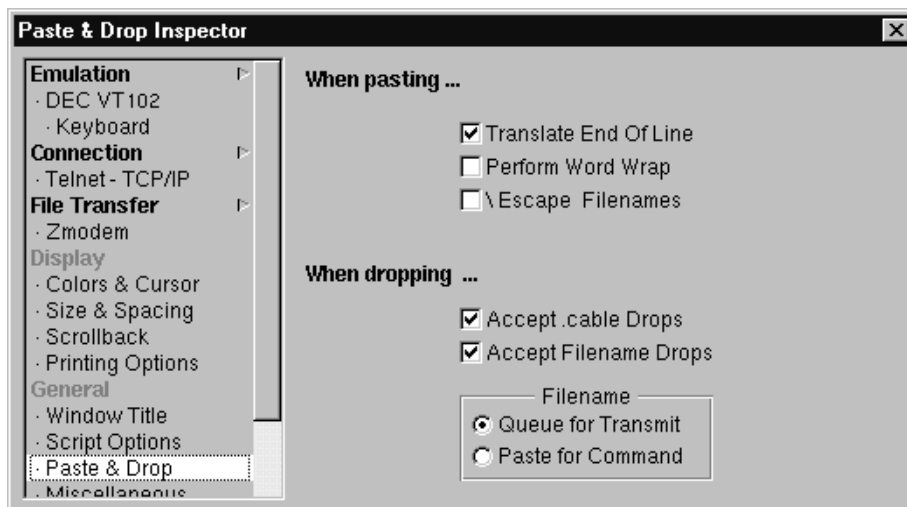
You can specify the maximum number of concurrently executing scripts the terminal allows. You can ensure that only one script at a time will be executed successfully by setting *Maximum concurrent scripts* to 1.

By unclicking *Generate Comments* you stop *Watch Me* (*Tools->Scripts...*) from generating comments when it records the script. In most cases the script will be reasonably understandable so you will not have to generate comments. If, however, you are recording a script on an IBM 3270 terminal, you will probably want to enable the *Generate Comments* options for improved readability.

The Copilot scripting facility and Copilot programming examples are discussed in separate chapters.

## Paste & Drop Options

Select *General/Paste & Drop* from the *Inspector* directory to display the *Paste & Drop Inspector*.



This inspector lets you determine what Cables does when you cut and paste a selection, or when you drag and drop a file.

### When pasting ...

#### Translate End of Line

If you select the *End of Line* option, then Cables will, when you paste text into the terminal window, translate the end of line character code into your Return key character code.

See *Tools->Inspector->Emulation->Keyboard*.

If you do not select the *End of Line* option, then Cables will not translate the end of line character code in pasted text.

#### Perform Word Wrap

If you select the *Word Wrap* option, then Cables will, when you paste text into the terminal window, insert the Return key character code before the first word that can't fit on the current line.

If you do not select *Word Wrap*, then Cables pastes text as is, without inserting the Return key character codes.

#### \ Escape Filenames

If you select the *\ Escape Filenames* option and you paste a filename, then Cables will insert the \ escape character before each unprintable character in the filename. This allows you, for example, to paste filenames that contain delimiters such as spaces and tabs.

### When dropping...

#### Accept .cable Drops

If you check *Accept .cable Drops*, then Cables allows you to reconfigure a session by dragging and dropping a .cable file into its session window.

### Accept Filename Drops

If you check *Accept Filename Drops*, then Cables allows you to drag and drop a file into a session window.

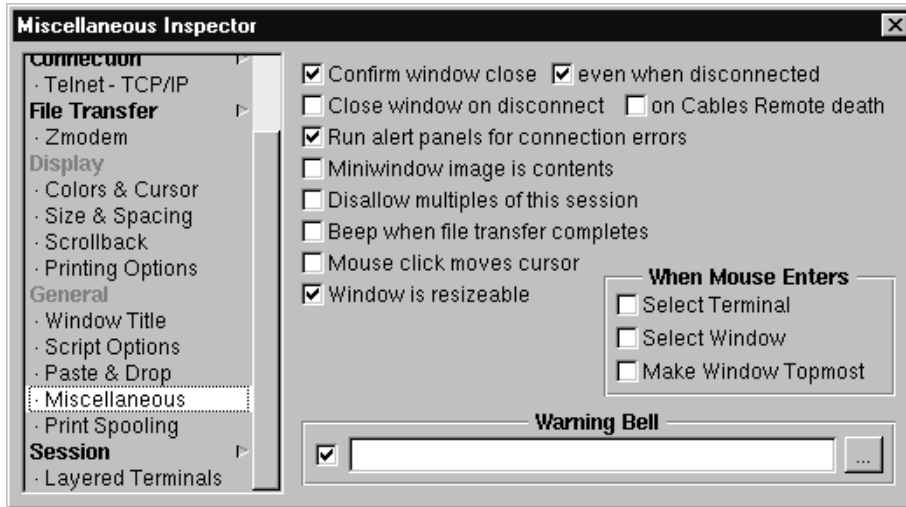
### Filename

The following options have an effect only if you've checked *Accept Filename Drops*:

- *Queue for Transmit* Whenever you drag and drop a file into the terminal, Cables will queue the file for transmission and bring up the *File Transfer Status* inspector.
- *Paste for Command* If you select *Paste for Command*, then, whenever you drag and drop a file into the terminal, Cables will paste the filename and append a space at the end.

## Miscellaneous Options

Select *General/Miscellaneous* from the *Inspector* directory to display the *Miscellaneous Inspector*.



### When Mouse Enters

#### Select Terminal

Cables makes the terminal of a multi-session configuration the key terminal whenever you place the cursor on it.

#### Select Window

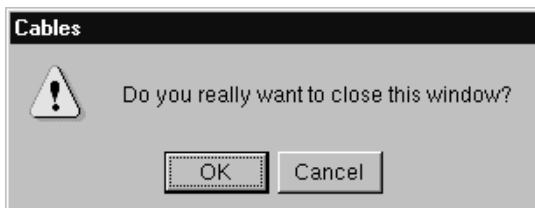
Cables makes a session window the key window whenever you place the cursor on it.

#### Make Window Topmost

Whenever you place the cursor on a session window, Cables brings the session window to the top without making it the key window.

### Confirm window close (even when disconnected)

If you check the *Confirm window close* box, Cables will prompt you with the following *Alert* panel whenever you close a connected session ( or a disconnected session if the second box is also checked) of this configuration:





## Close window on disconnect (on Cables Remote death)

Select *Close Window on Disconnect* if you want to close the session window when you disconnect the terminal. Select *on Cables Remote death* if you want to close the session when the remote connection dies. These attributes have no effect on a terminal in a multi-terminal session.

## Run alert panels for connection errors

If you check *Run alert panel for connection errors*, Cables will pop up an alert panel with an error message. For example, if you check *Close window on disconnect* when the disconnect occurs, an alert panel will tell you that the session is closed, and the window will close only after you have clicked *OK* on the alert panel.

## Beep when file transfer completes

If you check *Beep when file transfer completes*, Cables will signal when file transfer is complete.

## Mouse click moves cursor

If you check *Mouse click moves cursor*, Cables will attempt to move the cursor to the position clicked.

## Disallow multiples of this session

If you check *Disallow multiples of this session*, Cables will allow only one session for this .cable file.

## Window is resizable

If you uncheck *Window is resizable*, then the window size is fixed.

## Warning bell



Checking the *Warning Bell* box enables the warning bell. You can specify a sound file in the text field. (Sound files have the .snd extension.) The ... button brings up the *Sound File* panel. Using this directory browser, you can select the .snd file to use as the warning bell for this terminal. Note: In Windows NT, you cannot specify a particular sound file, the warning bell is the one defined by Windows NT.

Unchecking *Warning Bell*, disables the warning bell.

## Print Spooling

Select *General/Print Spooling* from the *Inspector* directory to display the *Print Spooling Inspector*.



Some terminals support special commands to direct and control screen output to an attached printer. This inspector allows you to change how Cables simulates the behavior of a terminal with an attached printer.

Note that this behavior is additional to Cables' implementation of the standard *Print* command which allows you to print any or all of the terminal's scrollback buffer at any time.

### Spooling Enabled

To enable spooling to the simulated attached printer, check the *Spooling Enabled* box.

If you enable spooling, then you have one of two options:

#### Save to File/Pipe to Command

*Save to File* causes Cables to write data destined for the attached printer to a file. When spooling ends, Cables will bring up a save panel so that you can specify the file's pathname.

*Pipe to Command* causes Cables to pipe data destined for the attached printer directly to the specified *Command*. If the *Command* is, for example, **lpr**, then data spooled to it will not get printed until the pipe is closed to force printing.

There are three ways of closing the pipe to force printing: the application on the host can send the terminal an escape sequence to stop spooling, you can press a key mapped to the **<spool:end>** macro, or you can close the session.

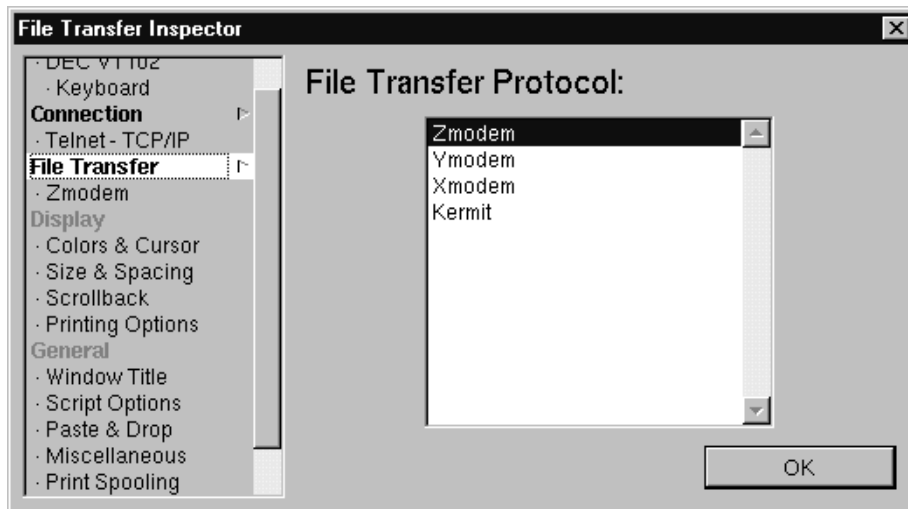
To change the command Cables spools to, click on the *Command* text field and enter the command. When you've edited the command, press the Return key.

Some terminals like the TI 931 write each line to the attached printer. If, for example, your Mac OS X Server machine has a printer on the parallel port **/dev/pp0**, you could more closely simulate TI 931 behavior by specifying the following *Command*: **cat > /dev/pp0**. This would cause every line piped to the **cat** *Command* to be printed.

An alternative to using the **cat** command with the TI 931 would be to set *Command* to **lpr** and map the macro **\<spool:end>** to a key. Whenever you wanted to send the spooled data to the printer, you could press this key.

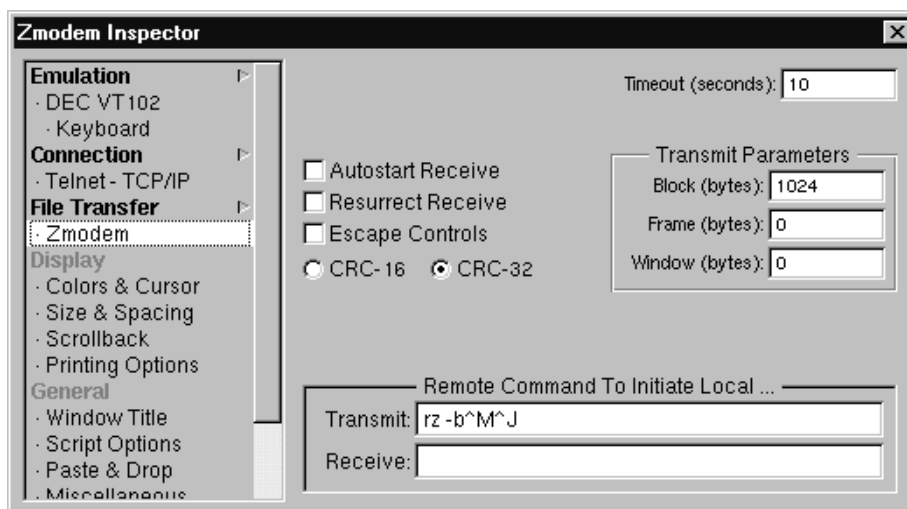
# File Transfer Protocols

Select *File Transfer* from the *Inspector* directory to display the *File Transfer Inspector*:



The *File Transfer Inspector* lists the file transfer protocols available to the terminal.

## Zmodem



### Timeout

Specify the timeout value in seconds.

### Autostart Receive

If you select *Autostart Receive* and invoke the **sz** command on the remote host, then Cables automatically receives the file without your first having to click the *Start* button in the *Zmodem Transfer Status* inspector (*Tools->Transfer...*).

### Resurrect Receive

If you select *Resurrect Receive* and only partially receive a file, then, when you try to receive the file again, Cables will append the untransmitted portion of the source file to the end of the destination file. This saves time when transferring very large files.

If you do not select *Resurrect Receive* and only partially receive a file, then, when you try to receive the file again, Cables resends the entire file.

### Escape Controls

If you select *Escape Controls*, then all control characters in the transmitted data will be escaped (transmitted as data instead of control).

### CRC-16/CRC-32

You can specify either a 16- or a 32-bit Cyclic Redundancy Check.

### Transmit Parameters

Specify the *Block*, *Frame*, and *Window* sizes in bytes.

- Window size is the number of bytes to send before waiting for an ACK. The default window size is the size of the file.
- Frame size is the number of bytes to send before inserting a header. The default frame size is the size of the file.
- Block size is the number of bytes checked by CRC. The default block size is 1024 bytes and cannot exceed the frame size.

You might want to reduce the window and frame sizes if you have a particularly noisy connection.

## Remote Command to Initiate Local ...

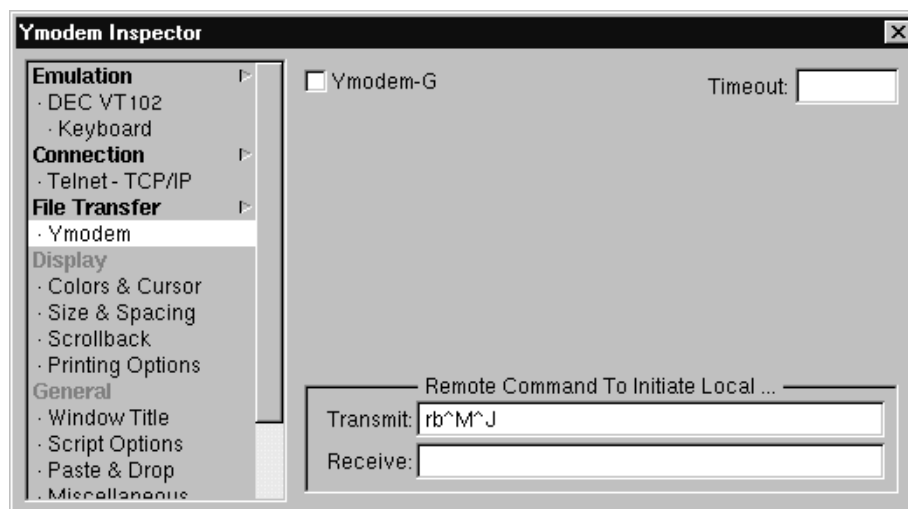
### Transmit

Specify the **rz** command line used on the remote host to receive the transmitted files.

### Receive

Specify the **sz** command line used on the remote host to send the files.

## Ymodem



### Timeout

Specify the timeout value in seconds.

### Ymodem-G

If you select *Ymodem-G* and an error occurs, Cables aborts the transfer.

### Initiate

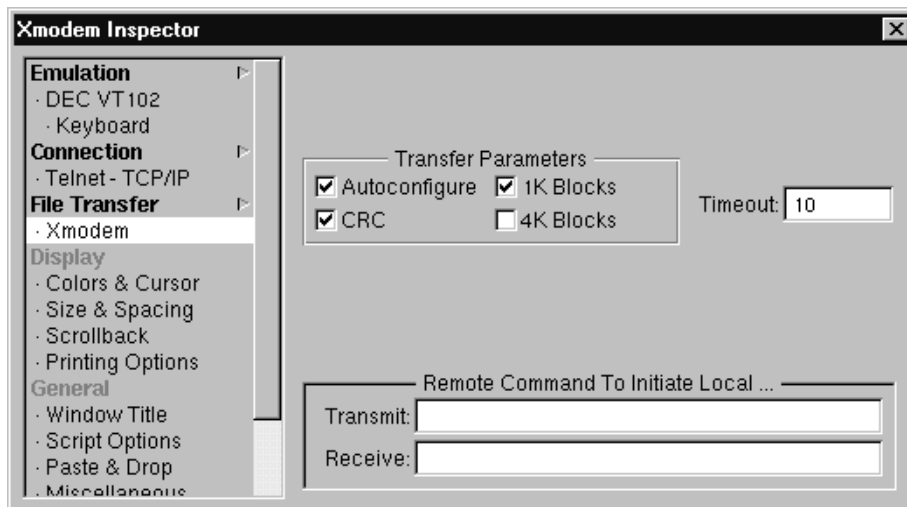
#### Transmit

Specify the **rb** command line used on the remote host to receive the transmission.

#### Receive

Specify the **sb** command line used on the remote host to send the files.

## Xmodem



### Transfer Parameters

#### Autoconfigure

If you select *Autoconfigure*, then Xmodem tries to automatically match CRC type and block size.

#### CRC

If you select *CRC*, Xmodem uses a 16-bit CRC, otherwise Xmodem uses an 8-bit checksum.

#### 1K Blocks

If you select *1K Blocks*, Xmodem can handle 1024 byte as well as 128 byte blocks.

#### 4K Blocks

If you select *4K Blocks*, Xmodem can handle 4096 byte as well as 1024 byte and 128 byte blocks. If you select neither *4K Blocks* nor *1K Blocks*, then Xmodem can handle only 128 byte blocks.

### Timeout

Specify the timeout value in seconds.

### Remote Command To Initiate Local

#### Transmit

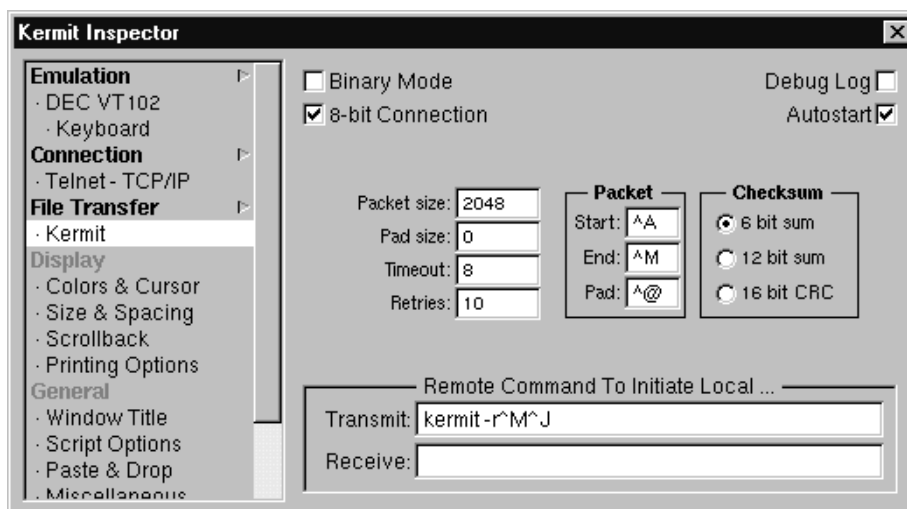
Specify the **rb** command line used on the remote host to receive the transmission.

#### Receive

Specify the **sb** command line used on the remote host to send the files.



## Kermit



### Binary Mode

Check *Binary Mode* to transfer binary files. Files are otherwise assumed to be text; the file will be converted to text file format on the target host.

### 8-bit Connection

Check *8-bit Connection* when you're sure you have an 8 bit link. Some remote Kermits may only allow 8-bit data transfers via an 8-bit connection. If you're using **rlogin** to connect, remember to use the **-8** option.

### Debug Log

Checking *Debug Log* causes Kermit to log debugging information to the file `/tmp/CablesKermitLog` which you can include in a problem report.

### Autostart

Checking *Autostart* causes Kermit to automatically start to receive files sent from a remote host without your first having to click the *Start* button on the *Kermit Transfer Status* inspector (*Tools->Transfer...*).

### Packet Size

Enter the packet size in bytes.

### Pad Size

Enter the pad size in bytes.

### Timeout

Enter the timeout value in seconds.

## Retries

Enter the number of retries before the transfer aborts.

## Packet

You can change the control character Kermit uses to identify the start, end, and padding of packets.

## Checksum

Choose the type of checksum Kermit uses.

- *6 bit sum*
- *12 bit sum*
- *16 bit CRC*

## Remote Command to Initiate Local . . .

### Transmit

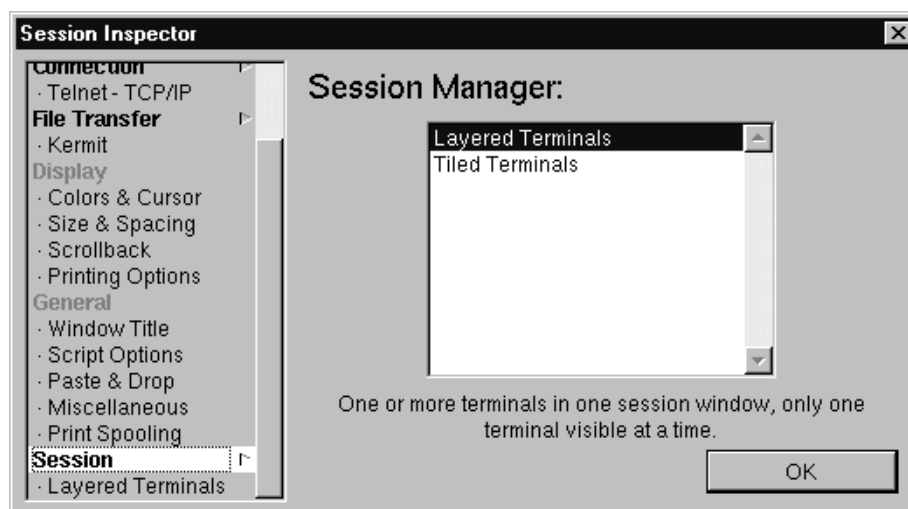
Specify the **kermit** command line used on the remote host to transmit files.

### Receive

Specify the **kermit** command line used on the remote host to receive files.

## Session Manager

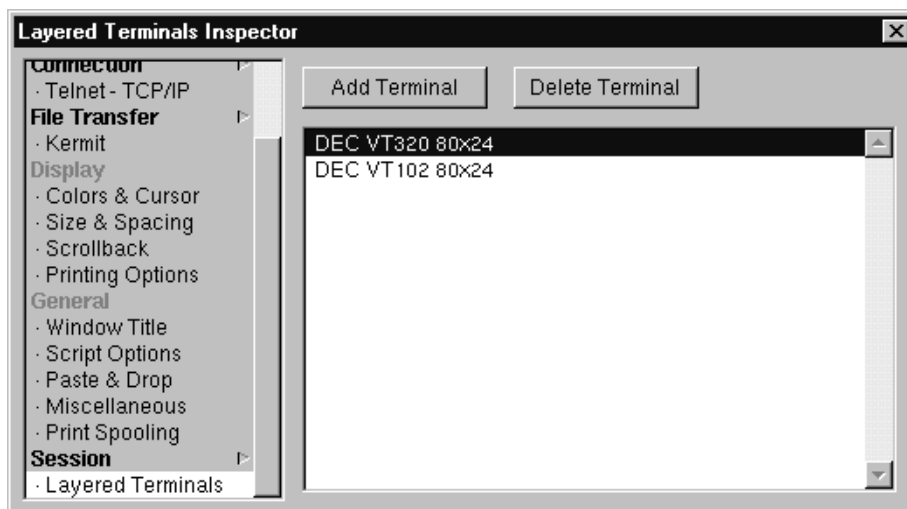
Select *Session* from the *Inspector* directory to display the *Session Manager Inspector*.



This inspector allows you determine how Cables displays sessions having more than one terminal. Double-click on *Layered Terminals* in the *Session Manager Inspector* to bring up the *Layered Terminals Inspector*.

**NOTE:** When you open a multi-terminal session, Cables always makes the session's first terminal the key terminal.

## Layered Terminals



The *Layered Terminals* option causes Cables to layer multiple terminals at the same position on the screen. Cables displays only the selected terminal of the layered session.

The above inspector shows that there are two terminals in the session window, a VT102 terminal and a VT320 terminal. When you select the VT320 terminal as above, Cables will display the VT320 terminal in the session window.

### Add Terminal

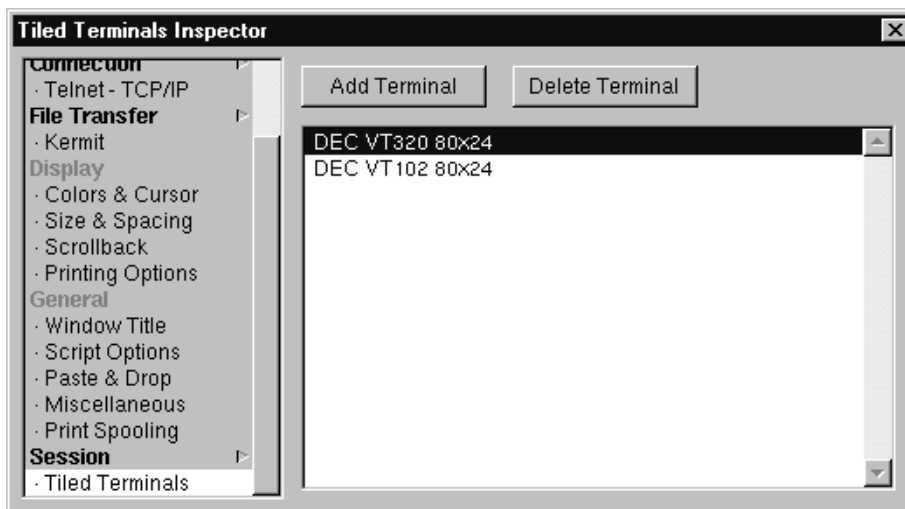
Click this button to add another terminal to your session. After you add the session, it will be disconnected. You can use the *Connection Inspector* to set the appropriate connection.

### Delete Terminal

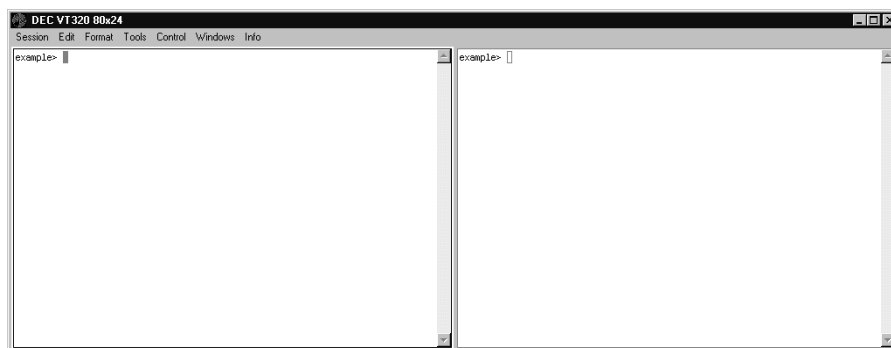
To delete a terminal, select one in the *Layered Terminal Inspector*, and click on *Delete Terminal*.

## Tiled Terminals

Double-click on *Tiled Terminals* in the *Manager Inspector* to display the *Tiled Terminals Inspector*:



The *Tiled Terminals* option causes Cables to tile multiple terminals in the same session window.



The above inspector shows that there are two terminals in the session window, a VT102 terminal and a VT320 terminal. When you select the VT320 terminal as above, Cables makes the VT320 the key terminal in the session window. Note that the Window Title changes to reflect which terminal you selected. You can save a multiple terminal session as you would save any other session by selecting *Session->Save* or *Save As...*

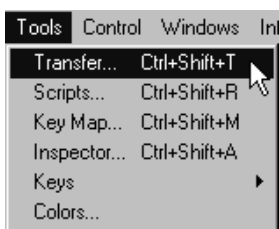
### Add Terminal

Click this button to add another terminal to your session. After you add the session, it will be disconnected. You can use the *Connection Inspector* to set the appropriate connection. Cables initially tiles the terminals horizontally. You can rearrange them vertically by dragging down the bottom edge of the session window far enough to accommodate the longest terminal to the right.

### Delete Terminal

To delete a terminal, select one in the *Tiled Terminal Inspector*, and click on *Delete Terminal*.

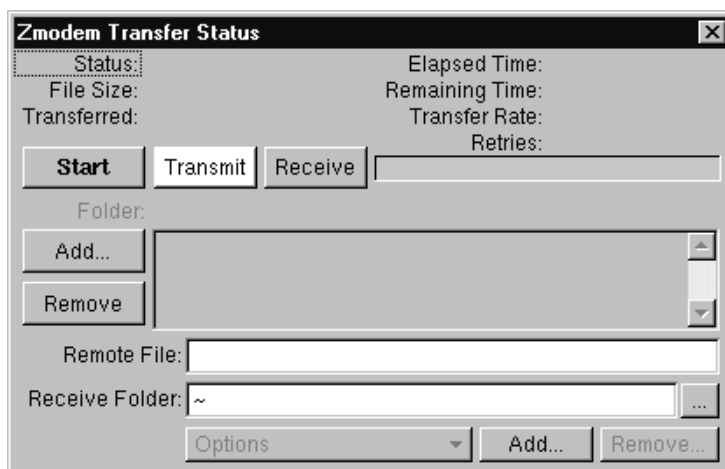
# Transfer



Selecting *Tools->Transfer...* brings up the *Transfer Status* inspector. The *Transfer Status* inspector allows you to transfer files between your computer and the remote host the terminal is connected to.

**NOTE:** The Cables file transfer protocol you selected (*Tools->Inspector->File Transfer*) must also be supported on the remote host.

## Transmit

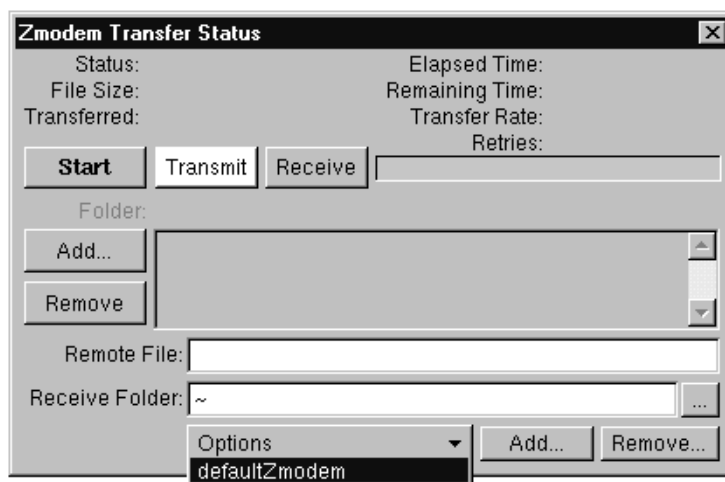


The *Transfer Status* inspector title reflects the file transfer protocol you selected for the key terminal using the *File Transfer Inspector* (*Tools->Inspector->File Transfer*). The above *Transfer Status* inspector indicates that you selected Zmodem.

You can specify the files you want to transfer either by pressing the *Add...* button and using the *Transmit Files* file browser, or by dragging file icons from the *File Viewer* and dropping them into the terminal window. In either case, the files will appear in the transmit list. You can specify the remote file name either in the *Transmit Files* file browser or by selecting the file entry in the scrollview and editing the *Remote File* text field.

To remove files from the transmit list, select the files and click the *Remove* button.

You can save the parameters you've set up in *Tools->Inspector->File Transfer->Zmodem* by adding a file transfer option. Click the *Add* button on the last row of the *Transfer Status* inspector to bring up *Add Options Template* panel. Enter the name with which you want to associate the current Zmodem parameters. Now, when you pull down the *Options* list in this configuration, you'll see that the new option is there:



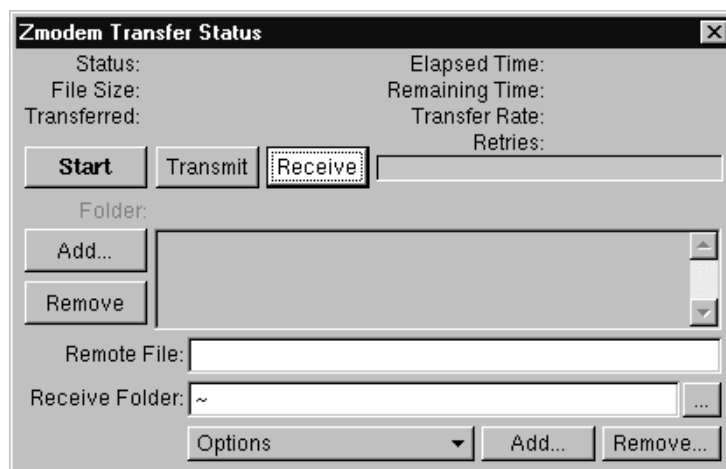
Any time you want to use these parameters to transfer a file from this terminal or another terminal configured for Zmodem, you can select the **defaultZmodem** option from the *Options* pull-down list.

**NOTE:** You cannot change the file transfer protocol for the terminal by selecting an option. For example, if you've changed the file transfer protocol for this terminal to Xmodem, selecting the file transfer option **defaultZmodem** does nothing. You will have to explicitly change the file transfer protocol back to Zmodem before selecting **defaultZmodem** initializes the Zmodem parameters.

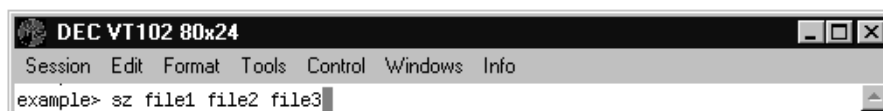
To start the file transfer, click on the *Start* button. Cables will transfer the files and list the statistics at the top of the panel.



## Receive



In order to receive files from the remote host, click the *Receive* button, click on the remote terminal window, and, if you're using ZModem, for example, type:



Press Return. Now in the *Transfer Status Inspector*, click *Start*. Cables will transfer the files and list the statistics at the top of the panel.

## Key Map



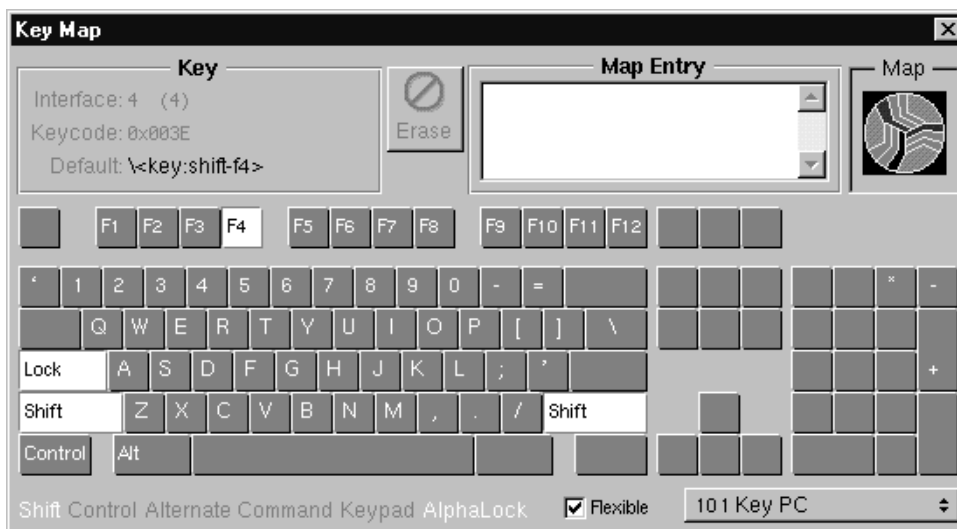
Selecting *Tools->Key Map...* brings up the *Key Map* inspector and the *Key Palette*. The *Key Map* inspector allows you to map keys from the emulation's keyboard to keys on the actual keyboard of your machine.

## Mapping Keys

The *Key Map* inspector displays the keyboard of the target machine.

**NOTE:** Cables allows you to prepare a .cable file for a target machine with a different keyboard from your current one. Drag down the keyboard option list at the lower right to select the target keyboard. In most cases, however, you will be configuring sessions for your current computer. To restore the target keyboard to your current one, select *Currently Attached* from the keyboard option list.

Assume that the *Key Map* inspector below is inspecting a VT320 terminal and the currently attached keyboard is a 101 key PC. You can see what the current mapping is for a key by either pressing the key on the actual keyboard or clicking on the key on the inspector keyboard. Cables will light up the selected key and will display the corresponding keycode and value:



When you inspect the default key mappings, you'll find that most of the emulation specific keys, i.e. the appropriate function and keypad keys, have been mapped in as defaults. You might, however, want to map keys in differently from the default mappings or map often used key sequences to special keys.

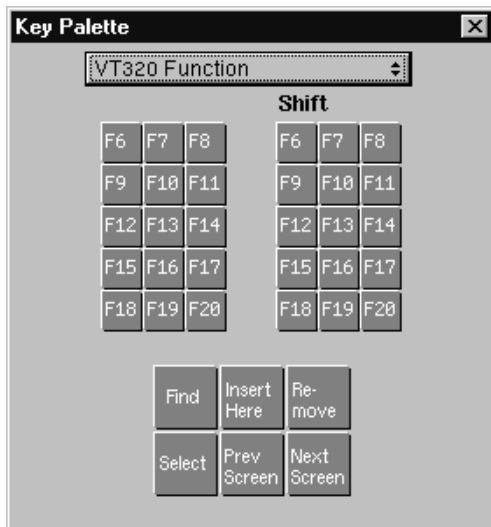
To map a key, first specify the target key. You can either press the key on the actual keyboard or click on the key in the inspector keyboard. You can in the same way select any of the control keys such as Shift, Control, Alternate, and Command. Now, in the *Key Palette*, double-click on the key you want to map into the selected target key. This key will now be mapped into the selected target key in the target keyboard.

For example, assume you want to map the Find key from the VT320 Function Keys palette to the selected key:

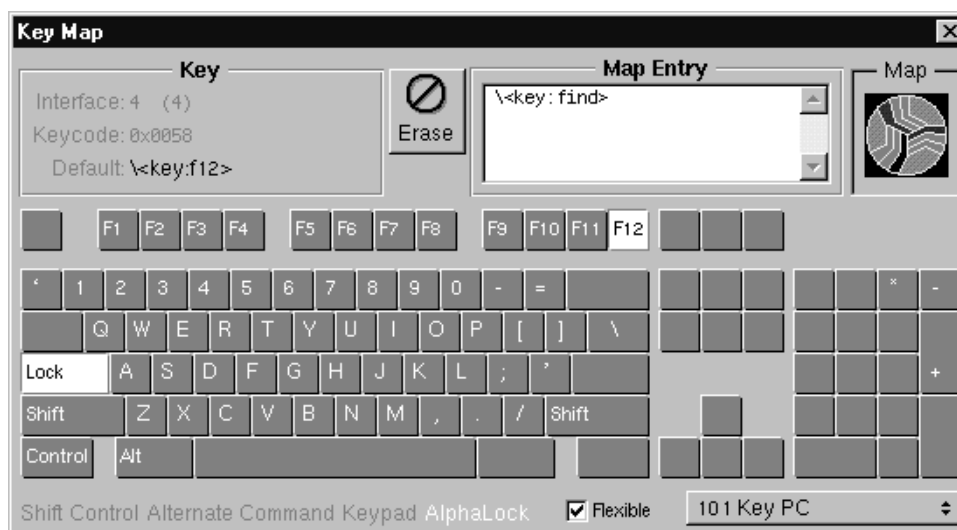


**NOTE:** The keys Control, (Shift) Lock, Alternate, and Command have no key mappings of their own; they modify the key mappings of other keys. If you use these modifier keys, a key that can be mapped has 16 possible mappings.

Drag the Find key in the *Key Palette* to the F12 Key on the KeyMap Inspector::



The Find key will now be mapped to the selected (shifted) key.



You can also remap keys by clicking in the *Map Entry* text box and typing in the text of the map entry.

## Key

The *Key* box in the upper left corner contains the following information:

- *Interface* identifies the major and minor number of the displayed keyboard. The major and minor numbers of the currently attached keyboard are in parentheses.
- *Keycode* is the hexadecimal code the selected key sends to the application. This is emulation dependent.
- *Default* is the default key mapping of the selected key.

## Erase

Clicking the *Erase* button (upper middle) causes Cables to erase the *Map Entry* for the selected key; this restores the default key mapping.

## Map Entry

You can map any key to output a string of text, control characters, other characters, and macros.

Enter the text string in the *Map Entry* box terminating the entry by pressing either the Return or Enter key.

The text string can be one or more of the following types:

- printable characters
- control characters

Control characters begin with ^ followed by the control character, e.g:

^[ is Escape or control-[

^C is control-C

^Z is control-Z

- special \ sequences

\a- bell

\b- backspace

\f- form feed

\n- newline

\r- return

\t- tab

\\- \ character

\^ - ^ character

\? - ? character

- ASCII literals

Octal ASCII literals begin with \0; hexadecimal ASCII literals begin with \x (lowercase).

For example, both \0376 and \xfe represent decimal 254.

- macros

Macros have the syntax: <*type:name*> where *type* is the type of macro and *name* is the name of the macro. Macros allow Cables to handle key mappings as symbols; entities whose values are configureable. Note that the drag and drop mapping is just a quick way of filling in the *Map Entry* with a macro.

Cables supports the following types of macros:

1. **key** macros expand to the keycodes for most keys on *Key Palette*. The following **key** macros are defined on *Key Palette->VT Series Keypad*:  
 pf1, pf2, pf3, pf4  
 kp0, kp1, kp2, kp3, kp4, kp5, kp6, kp7, kp8, kp9  
 kp-minus, kp-plus, kp-period, kp-enter  
 up, down, right, left
2. **link** macros expand to operations which affect the current connection.  
 There is currently one **link** macro defined on *Key Palette->General Keys* on the Break key:

<link:break>Causes Cables to send the break signal to the modem.

3. **application** macros control the application.

\<application:hide> Hides Cables.app.  
 \<application:unhide> Unhides Cables.app.

4. **session** macros expand to operations which control the session window.

<b>miniaturize</b>	Miniaturize the session window.
<b>deminaturize</b>	Maximize the session window.
<b>close</b>	Close the session.
<b>previous</b>	Makes the previous session's window the key window.
<b>next</b>	Makes the next session's window the key window.
<b>make-key</b>	Makes the current session the key window.
<b>order-front</b>	Orders the current session window to the front.

5. **terminal** macros allow you to manipulate the terminal. The following **terminal** macros are defined on *Key Palette->General Keys*:

<b>next</b>	Focus on the next terminal in the session.
<b>previous</b>	Focus on the previous terminal in the session.
<b>page-up</b>	Scrolls the screen up a page.
<b>page-down</b>	Scrolls the screen down a page.
<b>line-up</b>	Scrolls the screen up a line.
<b>line-down</b>	Scrolls the screen down a line.

In addition, there are the following **terminal** macros:

<b>kill-all-copilots</b>	Kills all Copilot scripts running on this terminal.
<b>copy</b>	Copies the selected text into the pasteboard.
<b>paste</b>	Pastes the selected text from the pasteboard into the terminal window.
<b>select</b>	Selects the specified text. The <b>select</b> macro has a special syntax:

**select row<sub>0</sub>,column<sub>0</sub> to row<sub>1</sub>,column<sub>1</sub>**

For example, entering the following in the *Map Entry* text window of the *Key Map* inspector causes the mapped key to select the first 15 lines and the first 10 characters of the 16th line of the screen:

\<terminal:select 0,0 to 15,9>

6. **spool** macros expand to operations which control spooling to attached printers. There is currently one **spool** macro defined on *Key Palette->General Keys* on the EndSpool key.

\<spool:end> causes Cables to print all spooled data.

7. **copilot** macros

Cables supports the *Key Map* macro type **copilot**.

You can use copilot macros to map Copilot scripts to keys. The Copilot script you specify in the copilot macro should not be defined as local and should not require arguments.

The *Map Entry* form of a copilot macro that invokes a script called **pascal** is:

\<copilot:pascal>

## 8. Copilot-related macros

\<terminal-raise:exception-name>	Raises the exception <i>exception-name</i> for all scripts running on this terminal
\<session-raise:exception-name>	Raises the exception <i>exception-name</i> for all scripts running on all terminals in this session.
\<application-raise:exception-name>	Raises the exception <i>exception-name</i> for all scripts running on all terminals in all sessions in the Cables application.

## 9. file **transfer** macros

You can control file transfers with the macro type **transfer**.

\<transfer:receive>	Sets the file transfer direction to receive.
\<transfer:transmit>	Sets the file transfer direction to transmit and initiates the transfer.
\<transfer:initiates>	Initiates the file transfer.

## 10. Cables Remote API macros

\<notify:notification-name>Sends the notification name to the Remote API clients registered to receive it.

The **notify** macro is a means for a Copilot script to communicate with Remote API clients. Complete documentation for the Cables Remote API is included in the YrridDeveloper package.

# Map

In the *Map* box is a Cable icon representing the terminal's key map configuration. You can drag and drop this *Key* icon onto another terminal window. This replaces the target terminal's key map configuration with a copy of the current terminal's.

# Flexible

If you check *Flexible* (the default), then any keys you map to the current target keyboard will, if possible, be mapped to keyboards with the same major number. For example, if the current terminal's keyboard is a *101 Key PC* (3/0), then most keys mapped on it will also apply to the following keyboard types *102 Key PC* (3/1), the *AX Desktop* (3/17), the *IBM 5576-A01 OADG* (3/16), and the *Toshiba J3100* (3/18).

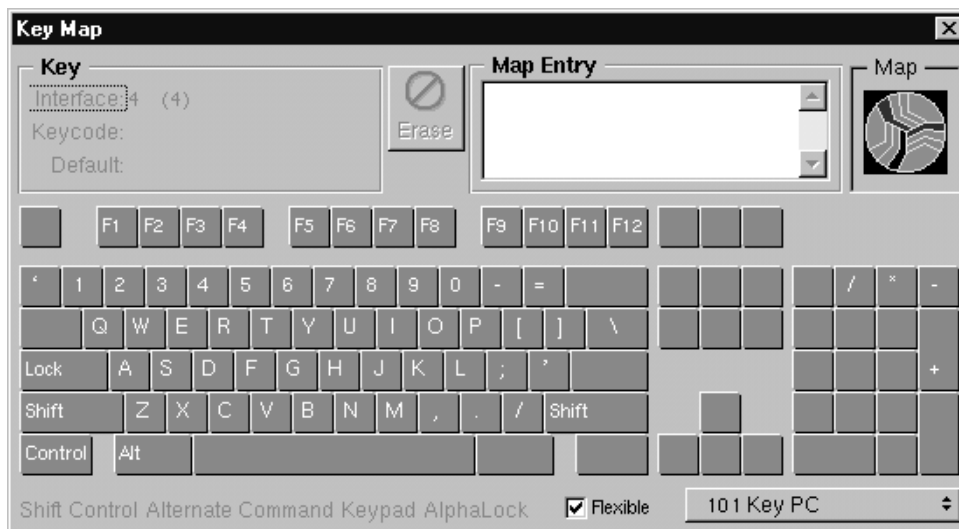
If you uncheck the *Flexible* box, then none of the keys that you map to the current target keyboard will be mapped to keyboards with the same major number. You generally do not want to uncheck *Flexible*, but there are cases where you might want to have uniquely mapped keyboards with the same major number.



## Key Map listings

Cables allows you to generate and save a list of the Key Map entries to an .rtf file.

You can bring up the Key Map inspector by selecting *Tools -> Key Map* from the main menu.

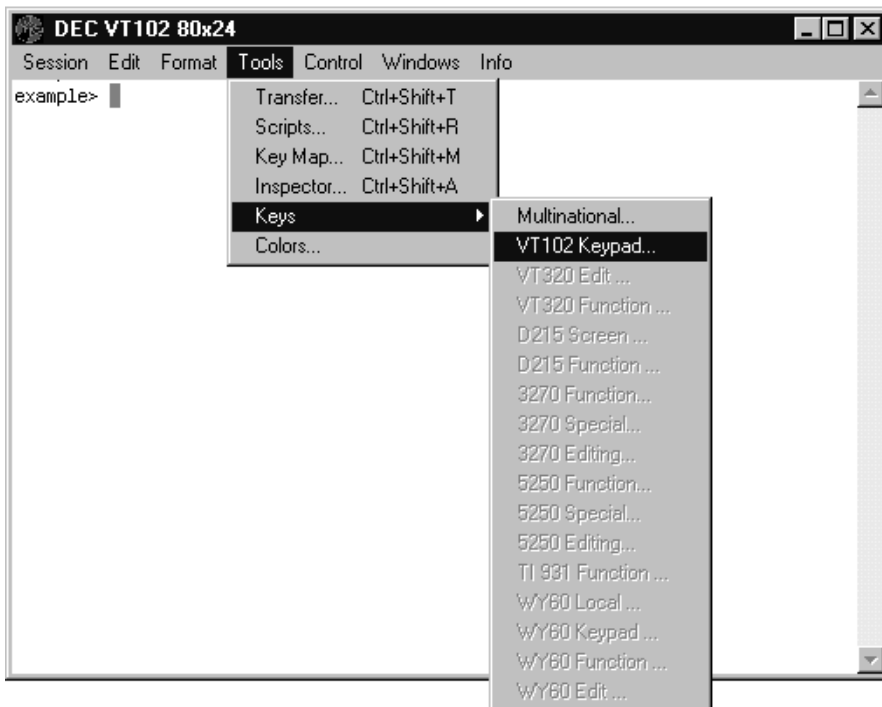


To create a *Key Map* listing, first open an .rtf window. This window can be a window in TextEdit, for example. Click on the Cables icon in the upper right-hand corner of the *Key Map* inspector and ctrl-drag the icon into the .rtf window. The *Key Map* listing will appear in the window. The re-mapped keys appear in bold on the left with their mapping listed on the right.

The *Key Map* listing, the .cable file, and the Cables.defaults file together describe completely all the user definable aspects of a Cables configuration.

**NOTE:** Creating a *Key Map* listing has no side-effects. If you have changed the *Key Mapping*, and you want to save it, you still have to save the configuration explicitly using *Session->Save As* and choosing a .cable filename.

# Keys



Selecting *Tools->Keys* displays a menu of key panels you can associate with the key terminal. Key panels are small windows which provide sets of useful, mostly emulation-specific keys. When you click on keys on the key panel, Cables sends the appropriate keycodes to the host. You can modify these keycodes with Shift and Control.

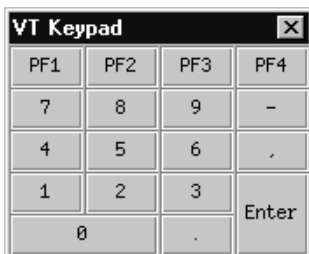
Using key panels can free up default mappings of keypad and function keys, thus allowing you greater flexibility when remapping your keyboard. Click on the key panel name you want to add to your terminal. You can then adjust the position of the key panel relative to the terminal window. Whenever you move the terminal window, the key panels associated with it will move with it. To remove a key panel from a terminal, click the key panel's close button.

Once you've associated key panels with a terminal and saved it, then every time you open that configuration Cables brings up the key panels along with the terminal window.

**NOTE:** In each emulation chapter is a list of the key panels specific to that emulation.

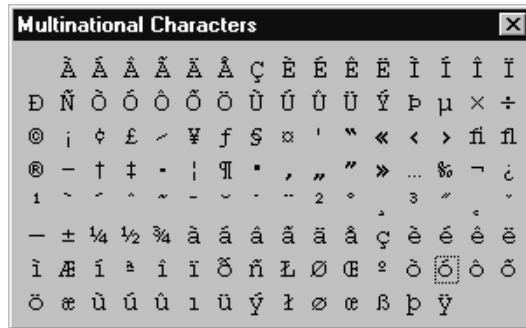
## VT102 Keypad

You can associate the VT102 keypad with your current VT series terminal:



## Multinational Characters

The *Multinational Characters* key panel provides the following multinational characters:





---

## 7 *Advanced Cables Configuration*

---

This chapter addresses advanced Cables system administration topics not covered elsewhere in the guide. If you are not familiar with Cables, we recommend you first install it, launch it, and try walking through the example described in the *Getting Started* chapter of the *Cables System Administrator's Guide* before reading the following sections.

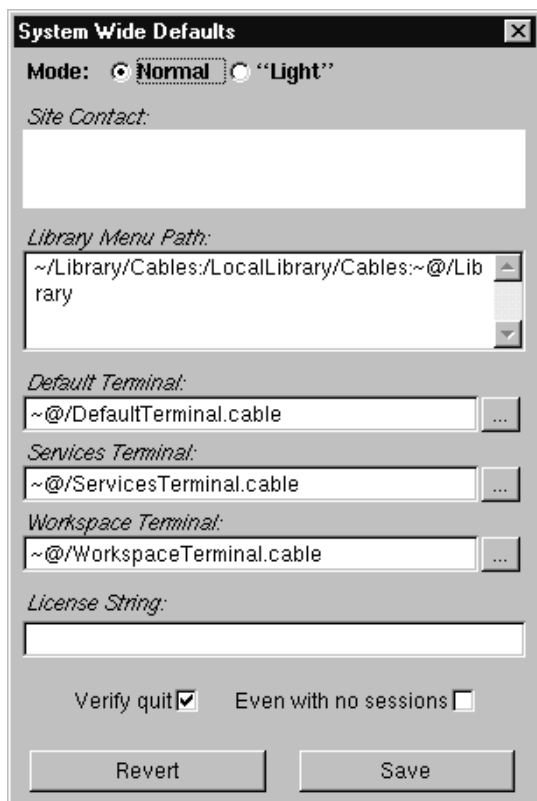
### **Configuring Cables defaults**

When you launch Cables, it initializes its defaults from two sources in the following order:

1. Cables.defaults or Cables-winnt.defaults, a text file in the Cables.app directory. In Windows NT, Cables-winnt.defaults is in C:\Yrrid\Cables.app\Resources.
2. your user defaults database.

A default defined in the user defaults database will, therefore, override a default of the same name in Cables.defaults.

There are certain system-wide defaults (Cables.defaults) that you can set only if you launch Cables.app with **root** or **Administrator** privileges. The *Info* panel will now contain a new item called *Admin...* Selecting *Info->Admin* brings up the following *System Wide Defaults* panel:



You can use this panel to set the following system-wide defaults:

## Mode

- *Normal* selects the normal, user-configureable mode for Cables. This sets the CablesNib default in the **Cables.defaults** or **Cables-winnt.defaults** file to Cables:

**“CablesNib”=“Cables”;**

- *Light* selects the simplified, non-user-configureable mode for Cables. This sets the CablesNib default in the **Cables.defaults** or **Cables-winnt.defaults** file to CablesLight:

**“CablesNib”=“CablesLight”;**

## Site Contact

Specify in the *Site Contact* text box the person you want users to contact in case of problems.

The *Site Contact* text box is similar to other text boxes except you must press alt-Return instead of just Return to go to the next line. After you’ve entered the *Site Contact* text, press Return.

## Library Menu Path

*Library Menu Path* is the list of directories Cables searches to find the configurations (.cable files) to put in the *Session->Library* menu. *Library Menu Path* also determines which Copilot scripts (.copilot files) Cables displays in the *Copilot Script Library* panel (*Tools->Scripts*) and in the *Control->Scripts* menu.

*Library Menu Path* is the initial value of the *TerminalLibrary* default in the **Cables.defaults** or **Cables-winnt.defaults** file. Each directory pathname in the *TerminalLibrary* search list is separated by a colon (:).

## Default Terminal

*Default Terminal* specifies the default terminal configuration that the *Session->New* command uses to create a new session.

*Default Terminal* is the initial value of the *DefaultTerminal* default in the **Cables.defaults** or **Cables-winnt.defaults** file.

## Services Terminal

*Services Terminal* is the default terminal configuration that the *Services* command uses.

*Services Terminal* is the initial value of the *ServicesTerminal* default in the **Cables.defaults** or **Cables-winnt.defaults** file.

## Workspace Terminal

*Workspace Terminal* is the default terminal configuration that ProjectBuilder and Workspace use.

*Workspace Terminal* is the initial value of the *WorkspaceTerminal* default in the **Cables.defaults** or **Cables-winnt.defaults** file.

## License String

If you have an Evaluation/Demo version of Cables, filling in the *License String* allows you to run Cables in Eval mode. If you do not fill in the License String, Cables will run in Demo mode, which allows you to run Cables for 20 minutes.

## Verify quit

Selecting *Verify quit* causes Cables to bring up a *Really Quit?* Alert panel if you try to *Quit* while there are still sessions open.

*Verify quit* is the initial value of the *VerifyQuit* default in the **Cables.defaults** or **Cables-winnt.defaults** file.

## Even with no sessions

Selecting *Even with no sessions* brings up the *Really Quit?* Alert panel if you try to *Quit* even if there are no open sessions.

*Even with no sessions* is the initial value of the *EvenWithNoSessions* default in the **Cables.defaults** or **Cables-winnt.defaults** file.

## Save

Press the *Save* button when you want the changes you have made to be saved to the **Cables.defaults** or **Cables-winnt.defaults** file.

## Revert

Pressing the *Revert* button causes the settings in the *System Wide Defaults* panel to revert to the values as they were last saved in the **Cables.defaults** or **Cables-winnt.defaults** file.

**NOTE:** You can either use the graphical user interface Cables provides to configure its defaults or directly edit the defaults in the **Cables.defaults** or **Cables-winnt.defaults** file. If you edit **Cables.defaults** or **Cables-winnt.defaults** directly, remember that the defaults names are case sensitive, e.g. *TerminalLibrary* is not the same as *terminallibrary*.

# Partial configuration files

Because .cable files are ASCII text files, you can configure, save, and edit a configuration file to contain a subset of the complete configuration. Cables allows you to drag and drop a partial .cable file icon into a running terminal session to change the specified attributes.

If, for example, you wanted to change the title of the main window to include the host name *read-and-black* and change the foreground and background colors to black and red before connecting to host *read-and-black*, you would:

1. Open a configuration,
2. Select *Tools->Inspector->General->Window Title* and change the main title to include the remote host name *read-and-black*,
3. Select *Tools->Inspector->Display-> Colors & Attributes* and change the foreground to black and background to red,
4. Save the configuration as *read-and-black*,
5. Use a text editor to delete all lines from *read-and-black.cable* except:

```
setup-version = 1.1
text-color-fore = Gray:[ 0.000000 ]
text-color-back = RGB:[ 0.858821 0.027451 0.086276 ]
window-main-title = "read-and-black [emulation] [width]x[height] [exit-value]"
window-mini-title = "red&black"
```

6. Drag and drop the *read-and-black.cable* icon into an open session.

**NOTE:**

If the partial .cable file contains a **pty-command** line, Cables will bring up the *Reconnect Session?* Alert panel, allowing you to optionally reconnect the session.

If you remove the **pty-command** line from the partial .cable file, Cables will reconfigure the session without your having to reconnect it.



---

## 8 *Cables Light*

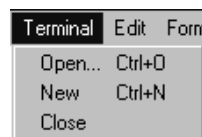
---

Many users of Cables will want to use preconfigured terminal sessions exclusively. You can configure Cables to run in a simplified mode of operation called Cables Light.

Cables Light differs from standard Cables in the following ways:



1. *Info* displays only the Info panel, i.e. there is no *Info->Preferences*.
2. Selecting *Sessions* brings up the menu of available sessions. *Sessions* is like the standard *Session->Library* menu. A user can start a preconfigured session by selecting a *Session* item.
3. *Terminal* replaces the standard *Session* menu item. It allows the user to *Open* any readable Cable file, open a *New* default session, and *Close* a session.



4. *Tools* is not in the user version; however, the standard *Tools->Keys* menu item is in Cables Light's main menu.

The rest of Cables Light, i.e. *Edit*, *Format*, *Control*, *Windows*, *Print*, *Services*, *Hide*, and *Quit* menu items, is identical to standard Cables.

---

# Configuring Cables Light

To configure Cables Light for your site, do the following:

1. Login as **root**.
2. Configure the sessions you want to appear in the Cables Light menu and save them in the appropriate directories.
3. Make sure that these .cable files are writable only by accounts you trust and are readable by the users who you want to open them.
4. Launch Cables and bring up the *System Wide Defaults* panel (*Info->Admin*).
5. Add the directories containing the Cables Light configurations you saved to the *Library Menu Path*.
6. Select *Mode*: “*Light*”.
7. Press the *Save* button.

Individual users who want to use standard Cables should execute the following command:

```
dwrite Cables CableNib Cables
```

This adds the line **Cables CablesNib Cables** to their user defaults. When the user launches Cables, it will come up as standard Cables.

Individual users can override the *Library Menu Path* (the value of *TerminalLibrary* in **Cables.defaults** or **Cables-winnt.defaults**) by setting it in their own defaults, e.g.:

```
dwrite Cables TerminalLibrary “~/Library/Cables:/LocalLibrary/Cables:./Library”
```

---

## 12 *Copilot Language*

---

Cables has a general scripting facility called Copilot. It provides the means of automating much of what you most commonly do in a terminal emulation session: setting up an environment, changing a directory, and running commands, scripts, and programs. Copilot also allows you to select and copy from one terminal and paste into another.

Selecting Tools ▶ Scripts... brings up the *Copilot Script Library* panel which allows you to create, save, and run Copilot scripts. You can also:

- use the Tools ▶ Inspector ▶ General/Script Options Inspector to specify what Copilot script Cables runs after it connects the terminal,
- use the Tools ▶ Key Map inspector to map Copilot scripts to keys,

Cables saves Copilot scripts in ordinary text files with the .copilot extension. When you launch Cables, it searches the directory path specified in the *TerminalLibrary* default. Cables reads in all the .copilot files it finds in these directories and lists them in the *File* column of the *Copilot* browser. (*TerminalLibrary* is the directory path Cables also uses to find the configurations (.cable files) to put in the Session ▶ Library menu).

# Copilot Script Library Panel

## Watch Me

Watch Me provides a way of automatically generating a Copilot script by processing your keyboard input and the application's responses.

The easiest way to create a Copilot script is to open a session, bring up the *Copilot Script Library* panel, click the *Start* button in the *Watch Me* box and type. Click the *Pause* button whenever you want to suspend *Watch Me* script recording and the *Resume* button whenever you want to resume recording.

When you're done, click *Stop*; a *Save/Append WatchMe* panel will appear. Specify the name of the .copilot file you want to save the script in and the name of the script. If the .copilot file already exists and you have not selected the *Append to File* option, the script will overwrite the contents of the existing .copilot file.

To edit a .copilot file, double-click on the .copilot entry in the *File* column of the *Copilot Script Library* browser. This will launch Edit.app. If you change the name of a procedure, save the .copilot file. When you click on the *Copilot* panel, Cables will redisplay the new procedure names in the middle column of the *Copilot* browser.

If there is another procedure with the same name in the file or in another .copilot file, Cables will display error messages in the bottom text window similar to:

```
Redefinition of function `newprocedure' in file ~/Library/Cables/new.copilot, previous one in ~/Library/Cables/old.copilot.
```

## New

To create a new Copilot script from scratch, click the *New* button. A *Save* panel will appear. Enter the name of the new Copilot file in the *Name:* text field and click *OK*. This creates a new empty file with the specified name with a .copilot extension. Cables then opens the new .copilot file in Edit.app so that you can enter Copilot code.

## Start

In order to run a script in the current session, select the procedure in the .copilot file you want and click the *Start* button. The script should require no arguments. As long as the script is running, Cables will display an instance number starting at 0 in the right column of the *Copilot* browser. This column shows you how many scripts are still running concurrently.

If you command-click on *Start*, Cables will start *Watch Me*, then reconnect the session. This is helpful if, for example, you want to connect the session to a **/usr/ucb/telnet** command and are using *Watch Me* to record the initial *Execute on connect* script (*Tools ▶ Inspector ▶ General ▶ Script Options*). If you connected to **telnet** first, then started *Watch Me*, *Watch Me* would not record the first password prompt. Your script, therefore, would not wait for the password prompt, and would immediately write the password to **telnet** before it was ready to read it.

## Stop

Click *Stop* to stop the selected executing script.

## Copilot Script Syntax

In the following syntax description the convention *name...* means none, one, or a list of names separated by commas. Keywords are in **bold**. Comments begin with # and run to the end of the line.

<b>procedure:</b>	procedure-name statement procedure-name ( arg-name... ) statement <b>local</b> / procedure-name statement <b>local</b> / procedure-name ( arg-name... ) statement
<b>statement:</b>	value ! conditional <b>case</b> conditional ... <b>end</b> <b>when</b> conditional ... <b>end</b> <b>break</b> <b>cycle</b> <b>catch</b> <b>return</b> value <b>repeat</b> statement exception-name <b>raise</b> procedure-name ( value... ) copilot : procedure-name value : value [ statement... ]
<b>conditional:</b>	condition statement
<b>condition:</b>	exception-name ~ value ? value <b>second</b> value <b>seconds</b> value <b>minute</b> value <b>minutes</b> value <b>hour</b> value <b>hours</b> value <b>byte</b> value <b>bytes</b>
<b>value:</b>	<b>any</b> <b>forever</b> integer floating-point string name

## Defining a procedure

A procedure definition consists of a slash (/), the procedure name, an option formal parameter list and a statement or statement list:

```
/ls[ "ls\r" ! ]
```

Or with one formal parameter:

```
/lsx(x) ["ls " ! x! "\r"]
```

Or with multiple formal parameters, with formal parameter names separated by commas:

```
/sxyz(x,y,z) ["ls " ! x ! " " ! y ! " " ! z ! "\r" !]
```

As soon as you save the .copilot file in which you've just defined a procedure, the new procedure will appear in the middle column of the *Copilot* browser. You can make a procedure local to its .copilot file by prepending the key-word `local` to its definition:

```
local /lsx(x) ["ls " ! x ! "\r" !]
```

The *Copilot* browser displays local procedure names in *italics*.

# Statements

## procedure calls

The syntax for a procedure call that executes serially is:

```
procedure-name ( value... )
```

Even if a procedure is defined with no formal parameters, it must be called with an empty argument list:

```
/ls-then-lsx [  
  ls()  
  lsx("/tmp")  
]
```

The syntax for a procedure call that executes in parallel is:

```
copilot:procedure-name
```

Suppose you've written the following script that brings up an alert panel whenever the terminal receives the string **"sos"**.

```
/sos-alert [  
  when  
    "sos" ? alert-panel("A ship is sinking")  
  end  
  
  never ~ []  
]
```

In order to start **sos-alert** from a script and have it run concurrently, you'd invoke it in the following way:

```
...  
copilot:sos-alert  
...
```

## send

The **send** statement sends the specified value to the terminal.

**"ls\r" !**

Sends the string **"ls\r"** to the terminal.

**x !**

If **x** is a formal parameter, sends the value of **x** to the terminal. If **x** is not a formal parameter, sends the string **"x"**.

**123 !**

Sends the string **"123"** to the terminal.

You can **send** the following characters:

- printable characters
- control characters begin with the control symbol ^ followed by the character, e.g:

<b>^[</b>	escape, control-[
<b>^C</b>	control-C
<b>^Z</b>	control-Z

- special \ sequences:

<b>\a</b>	bell
<b>\b</b>	backspace
<b>\f</b>	formfeed
<b>\n</b>	newline
<b>\r</b>	return
<b>\t</b>	tab
<b>\\</b>	backslash
<b>\^</b>	caret
<b>\?</b>	question mark

- ASCII literals  
Octal ASCII literals begin with **\0**  
Hexadecimal ASCII literals begin with **\x** (lowercase).  
For example, both **\0376** and **\xfe** represent decimal **254**.
- macros



## condition statements

A condition statement waits for a specific event to happen.

**"host> " ? []**

Waits for the terminal to receive the string **"host> "**, then does nothing.

**"host> " ? "ps\r"**

Waits for the terminal to receive the string **"host> "**, then sends the string **"ps\r"**.

**10 seconds "\n" !**

Waits for 10 seconds, then sends a newline character.

**1 minute "^D" !**

Waits for a minute, then sends a **ctrl-D**.

**20 bytes "\<key:up>" !**

Waits for the terminal to receive 20 bytes, then sends the keycode for the **up** key.

**wakeup ~ "I'm up" !**

Waits for the wakeup exception, then sends the string **"I'm up"**.

**"row=0,col=0" @ "Back home again" !**

Waits for the cursor to be in the home position, then sends the string **"Back home again"**. The following rules apply to the value preceding the @ cursor wait condition:

- There are three keywords: **row**, **column**, and **length**. Minimum uniqueness applies, e.g.: **r** can be used for **row**, **col** or **c** for **column**, **len** or **l** for **length**.
- **row**, **column** must have a (zero-based), decimal value within physical limits of the terminal. For example, if the terminal has 24 rows and 80 columns, **row** must be within the range 0-23 and **column** must be within 0-79.
- **length** can only be specified if both **row** and **column** are specified. **length** is the length of the screen field starting in the specified **row** and **column**.
- Specifying **row=x** without a **column** is shorthand specifying for the entire row x.
- Specifying **column=y** without a **row** is shorthand specifying for the entire column y.

The following are more examples of the @ cursor condition:

**"r=1,column=5,length=5" @ "00000"**

Waits for the cursor to be in the 5 character field beginning at row 1/column 5, then sends the string **"00000"**.

**"r=1" @ "In row one"**

Waits for the cursor to be in any column in row 1, then sends the string **"In row one"**.

**"col=60" @ "^G"**

Waits for the cursor to be in column 60, then sends a **control-G**.

## repeat

**repeat** executes its statement the specified number of times. If the **repeat** value is 0 or less, or not an integer, then **repeat** does not execute the statement.

Send "**Good morning!\r**" 10 times:

```
10 repeat "Good morning!\r" !
```

Send "**hi\r**" and wait for "**lo**" 10 times:

```
10 repeat [
  "hi\r" !
  "lo" ? []
]
```

Send "**echo testing\r**" every 7 seconds until you receive "**uncle**":

```
local /say_uncle [
  when
    "uncle" ? return
  end

  forever repeat [
    7 seconds "echo testing\r" !
  ]
]
```

The following **repeat** statement sends the **key** macro **pf1** 10 times:

```
10 repeat [
  key:pf1
]
```

This is equivalent to:

```
10 repeat [
  \<key:pf1>" !
]
```

## case

The **case** statement waits for a set of conditions, executes the first condition met and then executes the statement following the **case** statement.

The following **case** statement waits for "yes" or "no" and replies to the contrary.

```
case
  "yes" ? "no" !# wait for "yes", send "no"
  "no" ? "yes" !# wait for "no", send "yes"
end
```

## when

The when statement is similar to the case statement; it does not, however, pend until one of its conditions is met. When a when statement first executes, it proceeds to the next executable statement. Whenever one of the when statements conditions is met, Cables performs the specified action.

You can use the **when** statement to handle asynchronous events while continuing execution of the rest of the script.

The following script lists the man pages for ls, ps and who, initiating the next man page after the host prompt, and replying to all "--More--" prompts with a space.

```
local /man3(prompt) [
  when
    "--More--" ? " " !
  end

  prompt ? "man ls\r" !
  prompt ? "man ps\r" !
  prompt ? "man who\r" !
  prompt ? []
]

/host_man3 [
  "\r" !
  man3("host> ")
]
```

Whenever a **when** statement exits normally (without executing a **catch**, **break**, **cycle**, or a **return**), execution resumes at the point where the **when** condition was met.

A **when** statement is cleared when its scope is exited. A **when** statement's scope is the code segment in which it resides that is bounded by the innermost pair of square brackets. **when** statements within a **repeat** statement are restarted at each iteration.

## catch

You can execute the **catch** statement only from within a **when** statement. Executing a **catch** statement causes all executing code after the **when** statement (within its scope) to be unwound. Execution resumes at the statement following the **when** statement.

```
when
  "catch" ? catch
end
"echo catch gets us here" !
. . .
forever repeat [
  7 seconds []
  "echo 7 seconds are up\r" !
]
```

## break

A **break** statement can be used in a **case** statement to exit a **repeat** statement. When a **break** executes, it transfers execution to the statement following the **repeat** statement.

```
forever repeat [
  case
    "break" ? break
    "go" ? []
  end
  "echo this is where go gets us\r" !
]
"echo this is where break gets us\r" !
```

If a **repeat** statement is executing when a previously established **when** condition is met and the **when** statement executes a **break**, then execution resumes at the statement following the **repeat** statement.

```
when
  "break" ? break
end

forever repeat [
  7 seconds []
  "echo 7 seconds are up\r" !
]
"echo this is where break gets us\r" !
```

## cycle

A **cycle** statement can be used in a **case** statement to cause the next iteration of the enclosing **repeat** statement.

```
forever repeat [  
  "echo cycle gets us here\r" !  
  case  
    "cycle" ? cycle  
    "go" ? []  
  end  
  "echo cycle won't get us here\r" !  
]
```

If a **repeat** statement is executing when a previously established **when** condition is met and the **when** statement executes a **cycle**, then the **repeat** statement performs another iteration.

```
when  
  "cycle" ? cycle  
end  
  
forever repeat [  
  "echo cycle gets us here\r" !  
  7 seconds []  
  "echo cycle won't get us here\r" !  
]
```

## return

Executing a **return** from a **when** or a **case** statement causes the procedure containing the **when** or the **case** to return.

---

## Built-in operations

**alert-panel**(*arg...*) takes a variable number of arguments. It brings up an alert panel with a message composed of the concatenated arguments.

**notify**(*notification-name*) posts the specified notification.

**notify**(*notification-name* , *aString*) posts the specified notification with *aString* as its data. Access *aString* with the method **notificationData**.

**open-file**(*filename*) opens the specified filename.

**system**(*string...*) takes a variable number of arguments. It calls the system(3) function with the concatenated strings as its argument. Output from the system(3) function call is written to Cables stdout which is the Workspace console.

**transfer-add-file**(*filename*) adds *filename* to the *File Transfer Status* queue as the local filename. The remote filename defaults to the name the remote subsystem defines.

**transfer-add-file**(*local* , *remote*) adds the *local* and *remote* filenames to *File Transfer Status* queue.

**transfer-set-direction**(*direction*) sets the *direction* for file transfers to "**transmit**" or "**receive**".

**transfer-set-options**(*options-name*) changes the file transfer options for the current file transfer protocol to those defined for the specified *options-name* from the [\*Tools ▶ Transfer Status Inspector\*](#) options list.

**transfer-set-receive-folder**(*directory-path*) sets the destination folder on a receive to *directory-path*.

# Exceptions

Copilot supports a system of named exceptions to handle error conditions and inter-terminal communication. When an exception is raised, Copilot will look for the most recent handler for that exception. If no handler is found, Copilot will generate an uncaught exception error.

Copilot supplies one built-in exception, **exit**, used to exit a script.

The syntax for creating a handler is:

```
when
  exception-name ~ exception-statement
end
```

The syntax for raising an exception is:

```
exception-name raise
```

In the following example, the procedure *xyz* calls the *send* procedure 3 times to send the string "xyz" one character at a time. If the *send* procedure receives the string "ERROR" within the 10 second timeout, it raises an exception named "error". The **error** exception handler catches the exception, displays an alert panel with an error message and returns.

```
/send(val) [
  val !
  case
    "ERROR" ? error raise
    10 seconds []
  end
]

/xyz [
  when
    error ~ [
      alert-panel("There was an error")
      return
    ]
  end

  send("x")
  send("y")
  send("z")
]
```

You can also use exceptions to communicate with terminals in the same session or with terminals in other sessions. This is especially useful if one terminal wants to signal to another that it has just copied text to the pasteboard and that the other terminal can paste it into its application.

The statement

```
terminal-raise:ok_to_paste
```

raises the **ok\_to\_paste** exception for all scripts executing in the key terminal.

The statement

```
session-raise:ok_to_paste
```

raises the **ok\_to\_paste** exception for all terminals in the session.

The statement

`application-raise:ok_to_paste`

raises the **ok\_to\_paste** exception for all terminals in all sessions in the Cables application.



# Macro Syntax

Macros allow Cables to handle key mappings as symbols; entities whose values are configurable.

Macros have the syntax:

```
\<macrotype:macroname>
```

Copilot supports the following shorthand for sending *Key Map* inspector macros:

```
macrotype:macroname
```

The following pairs of Copilot statements are equivalent:

```
"\<key:enter>" !  
key:enter
```

```
"\<copilot:pascal>" !  
copilot:pascal
```

```
"\<link:break>" !  
link:break
```

Cables supports the following types of macros:

- 1 **key** macros expand to the keycodes for most keys on Key Palette. The following key macros are defined on *Key Palette ▶ VT Series Keypad*:

```
pf1, pf2, pf3, pf4  
kp0, kp1, kp2, kp3, kp4, kp5, kp6, kp7, kp8, kp9  
kp-minus, kp-plus, kp-period, kp-enter  
up, down, right, left
```

- 2 **link** macros expand to operations which affect the current connection. There is currently one link macro defined on *Key Palette ▶ General Keys* on the **Break** key:

```
\<link:break>Causes Cables to send the break signal to the modem.
```

- 3 **application** macros control the application.

```
\<application:hide>Hides Cables.app.  
\<application:unhide>Unhides Cables.app.
```

- 4 **session** macros expand to operations which control the session window.:

<b>miniaturize</b>	Miniaturizes the session window.
<b>deminaturize</b>	Maximizes the session window.
<b>close</b>	Closes the session.
<b>previous</b>	Makes the previous session's window the key window.
<b>next</b>	Makes the next session's window the key window.
<b>make-key</b>	Makes the current session the key window.
<b>order-front</b>	Orders the current session window to the front.

- 5 The following **terminal** macros are defined on *Key Palette ▶ General Keys*:

<b>next</b>	Focus on the next terminal in the session.
<b>previous</b>	Focus on the previous terminal in the session.

<b>page-up</b>	Scrolls the terminal's screen up a page.
<b>page-down</b>	Scrolls the terminal's screen down a page.
<b>line-up</b>	Scrolls the terminal's screen up a line.
<b>line-down</b>	Scrolls the terminal's screen down a line.

- 6 In addition, there are the following **terminal** macros (not defined on any *Key Palette*):

<b>kill-all-copilots</b>	Kills all Copilot scripts running on this terminal.
<b>copy</b>	Copies the selected text into the pasteboard.
<b>paste</b>	Pastes the selected text from the pasteboard into the terminal window.
<b>select row0 , column0 to row1 , column1</b>	Selects the specified text.

The **select** macro has a special syntax. The following macro causes the mapped key to select the first 15 lines and the first 10 characters of the 16th line of the screen:

```
\<terminal:select 0,0 to 15,9>
```

- 7 **spool** macros expand to operations which control spooling to attached printers. There is currently one spool macro defined on *Key Palette* ▶ *General Keys* on the **EndSpool** key.

```
\<spool:end>
causes Cables to print all spooled data.
```

- 8 **copilot** macros

Cables supports the *Key Map* macro type **copilot**. You can use **copilot** macros to map Copilot scripts to keys. The Copilot script you specify in the **copilot** macro should not be defined as local and should not require arguments. The *Map Entry* form of a copilot macro that invokes a script called **pascal** is:

```
\<copilot:pascal>
```

- 9 Copilot-related macros

```
\<terminal-raise:exception-name>
Raises the exception exception-name for all scripts running on this terminal
```

```
\<session-raise:exception-name>
Raises the exception exception-name for all scripts running on all terminals in this session.
```

```
\<application-raise:exception-name>
Raises the exception exception-name for all scripts running on all terminals in all sessions in the Cables application.
```

- 10 file **transfer** macros

You can control file transfers with the macro type **transfer**.

```
\<transfer:receive>
Sets the file transfer direction to receive.
```

```
\<transfer:transmit>
Sets the file transfer direction to transmit and initiates the transfer.
```

**\<transfer:initiates>**

Initiates the file transfer.

11 Cables Remote API macros

**\<notify:notification-name>**

Sends the notification name to the Remote API clients registered to receive it.

The **notify** macro is a means for a Copilot script to communicate with Remote API clients.

# IBM 3270 Key Macros

## IBM 3270 Edit

The following key macros are defined in the *IBM 3270 Edit Key Palette* ([Tools ▶ Key Map](#)):

**tab**  
**backtab**  
**left2**  
**left**  
**right**  
**right2**  
**dup**  
**fieldmark**  
**up2**  
**up**  
**down**  
**down2**  
**home**  
**newline**  
**backspace**  
**reset**  
**clear**  
**insert**  
**delete**  
**cursor-select**  
**field-end**  
**erase-to-eof**  
**erase-input**  
**erase-field**

Additionally the following key macros are defined:

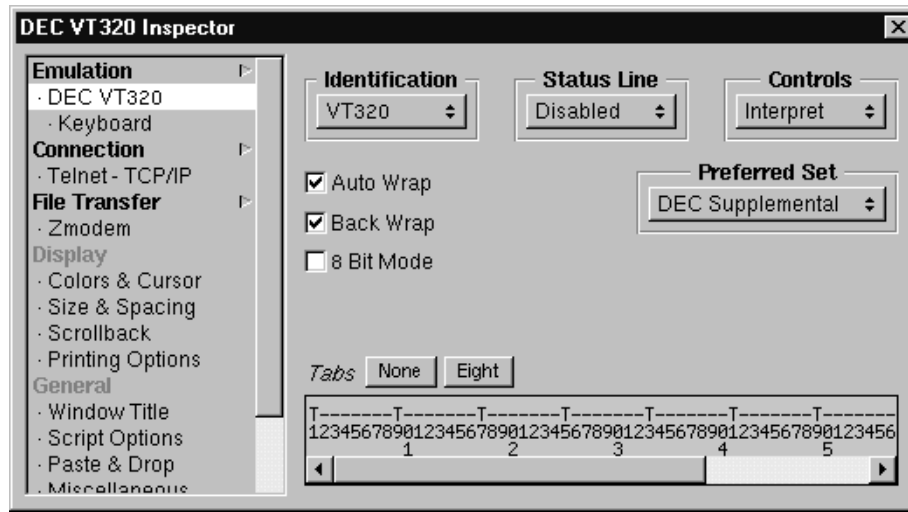
**attn**  
**print-screen**

## IBM 3270 Function

The following key macros are defined in the *IBM 3270 Function* key palette ([Tools ▶ Key Map](#)):

**pf1, pf2, pf3, pf4, pf5, pf6, pf7, pf8, pf9, pf10, pf11, pf12,**  
**pf13, pf14, pf15, pf16, pf17, pf18, pf19, pf20, pf21, pf22, pf23, pf24,**  
**pf25, pf26, pf27, pf28, pf29, pf30, pf31, pf32, pf33, pf34, pf25, pf36**  
  
**enter**  
**pa1, pa2, pa3**  
**sysreq**

## Emulation Attributes



### Identification

Drag and release the *Identification* option list to select the terminal identification. The VT320 emulation can identify itself as either a VT320, VT220, VT102, VT101, or VT100.

### Status Line

- *Disabled* disables the status line. (Restores the original number of rows.)
- *Empty* enables the bottom row of the screen as an empty status line. This reserves an unwritable status line.
- *Indicator* enables the bottom row of the screen as a status line displaying the cursor position and printer status.
- *Writable* enables the bottom row of the screen as a writable status line.

Enabling a status line causes the screen to become shorter by one row.

### Controls

- *Interpret* causes the terminal to interpret unprintable characters normally.
- *Display* causes the terminal to display unprintable characters as printable strings, e.g. the CR character is displayed as <CR>.

## Preferred Set

- *DEC Supplemental* causes the terminal to map the keycodes in the range 128-255 to the *DEC Supplemental* character set.
- *ISO Latin 1* causes the terminal to map the keycodes in the range 128-255 to the *ISO Latin 1* character set.

## Auto Wrap

Selecting *Auto Wrap* causes text entered past the last column to wrap onto the next row.

If you don't check *Auto Wrap*, then, when text is displayed past the last column of the current row, the cursor will stop at the last column; each new character will overwrite the last one.

## Back Wrap

If you check the *Back Wrap* option and use the Delete key to delete text that has wrapped over multiple lines, Cables will backtrack the cursor over the wrapped line. If you do not check *Back Wrap*, then the Delete key will still delete text, but the cursor will stop at the first column of the current line.

Note that *Back Wrap* only affects the behavior of backspace (BS) keycode.

## 8 Bit Mode

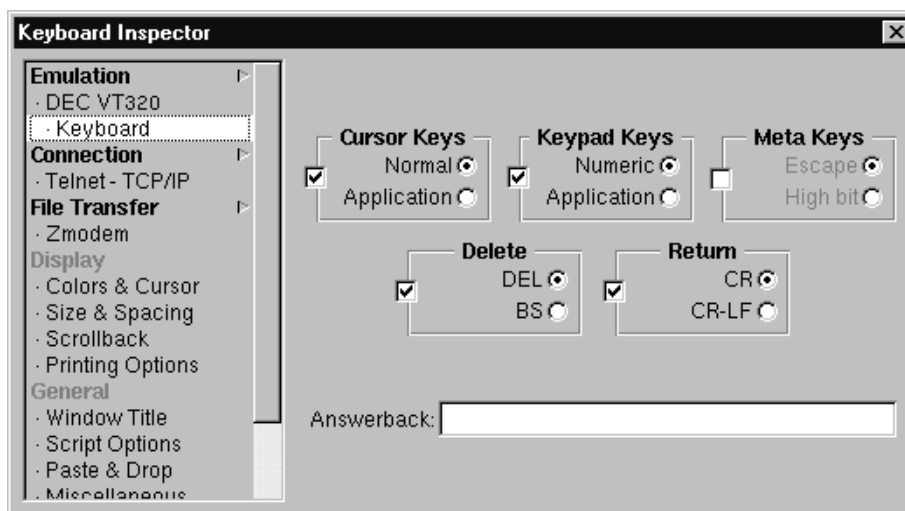
Check the *8 Bit Mode* box to allow the session to display and transmit multinational characters in the upper 128 characters.

## Tabs

- *None* removes all tabs.
- *Eight* puts a tab every eight columns.

You can also insert a tab by positioning the cursor in the tab ruler and clicking. To delete a tab, position the cursor over it and click.

## Keyboard



Select *Emulation->Keyboard* in the Inspector directory to display the *Keyboard Inspector*. This inspector allows you to change the mapping of the Cursor keys, Keypad keys, the Delete, Return, and Meta Keys, and to edit the answerback string.

### Cursor Keys

Unchecking the *Cursor Keys* box passes the cursor keycodes on to the application *without* translating to a VT series escape sequence.

Checking the *Cursor Keys* box enables one of the following options:

- *Normal* converts the cursor keycodes to the VT series “normal” escape sequence.
- *Application* converts the cursor keycodes to the VT series “application” escape sequence.

### Keypad Keys

Unchecking the *Keypad Keys* box passes the keypad keycodes on to the application without translating to a VT series escape sequence. **NOTE:** If you have a new style NeXT keyboard, you may want to uncheck the *Keypad Keys* box to be able to access the pipe (“|”) and backslash (“\”) keys.

Checking the *Keypad Keys* box enables one of the following options:

- *Numeric* converts the keypad keycodes to the VT series “numeric” representation.
- *Application* converts the keypad keycodes to the VT series “application” escape sequence.

### Meta Keys

Unchecking the *Meta Keys* box makes the Alternate key function normally.

Checking the *Meta Keys* box enables one of the following options:

- *Escape* causes the Alternate key to function as the EMACS meta key. If you hold down the Alternate key and press, for example, the **a** key, Cables will send the ESC (escape) character (0x1B) and then the **a** character (0x61).

- *High Bit* causes the Alternate key to set the high bit in the character code. If you hold down the Alternate key and press, for example, the **a** key, Cables will send the code 0xE1.

## Delete

Unchecking the *Delete* box maps the Delete key to its value in the Keyboard mapping.

Checking the *Delete* box maps the Delete key to:

- *DEL* the DEL (delete) character (0x7F).
- *BS* the BS (backspace) character (0x08).

## Return

Unchecking the *Return* box maps the Return key to its value in the Keyboard mapping.

Checking the *Return* box maps the Return key to:

- *CR* the CR (carriage return) character (0x0D).
- *CR-LF* both the CR and the LF (line feed) characters (0x0D and 0x0A).

## Answerback

The DEC VT series terminals send the answerback string back to the host in response to the host sending the ^E (control-E) command.

Click in the answerback text field to add or edit the answerback string.

# Key Macros

**NOTE:** You can also use macros defined on other VT series key palettes.

## VT320 Function

The following key macros are defined in the *VT320 Function* key palette (*Tools->Key Map*):

f6, f7, f8, f9, f10, f11, f12, f13, f14, f15, f16, f17, f18, f19, f20

shift-f6, shift-f7, shift-f8, shift-f9, shift-f10, shift-f11, shift-f12,  
shift-f13, shift-f14, shift-f15, shift-f16, shift-f17, shift-f18, shift-f19, shift-f20

find  
insert  
remove  
select  
prev  
next

Additionally the following key macros are defined:



f1, f2, f3, f4, f5

shift-f1, shift-f2, shift-f3, shift-f4, shift-f5

left

up

right

down

pf1, pf2, pf3, pf4

kp0, kp1, kp2, kp3, kp4, kp5, kp6, kp7, kp8, kp9

kp-minus, kp-comma, kp-enter, kp-period, kp-plus

help

do

print-screen

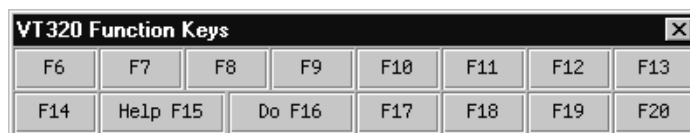
## Key Panels

The following *Key Panels* are defined for DEC VT320 emulations:

### VT320 Edit



### VT320 Function



---

## References

The following manuals on the VT series are available from DEC:

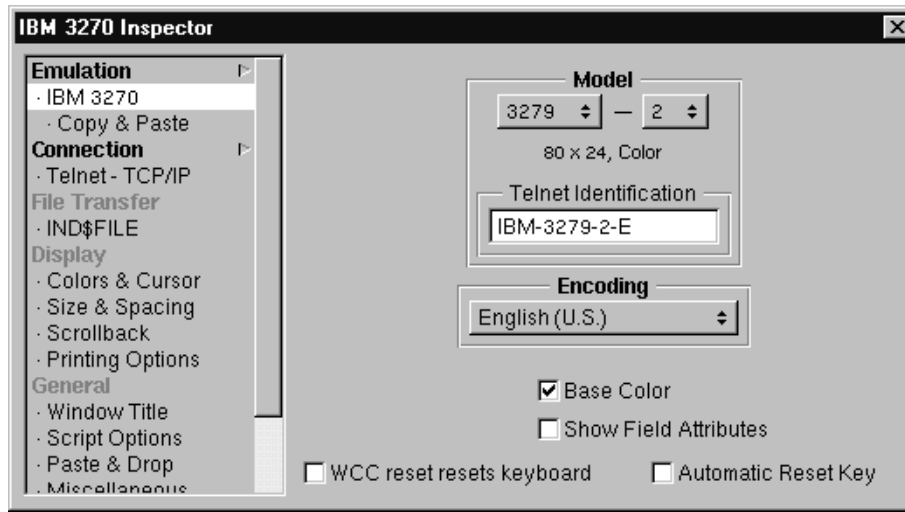
VT220 Owner's Manual

VT220 Programmer's Manual

VT320 Owner's Manual

VT320 Programmer's Manual

## Emulation Attributes



### Model

Select the following IBM 3270 terminal models by dragging and releasing the two option lists in the *Model* box.

The following table lists the attributes of the various IBM 3270 models:

**Table 2:**

Model	Cols	Rows	Color/Mono
3279-2	80	24	color
3279-3	80	32	color
3279-4	80	43	color
3279-5	132	27	color
3278-2	80	24	monochrome
3278-3	80	32	monochrome
3278-4	80	43	monochrome
3278-5	132	27	monochrome

---

## Base Color

Checking the *Base Color* box allows Cables to display attributed text from monochrome applications in color. The following table lists which colors Cables displays for which attributes:

**Table 3:**

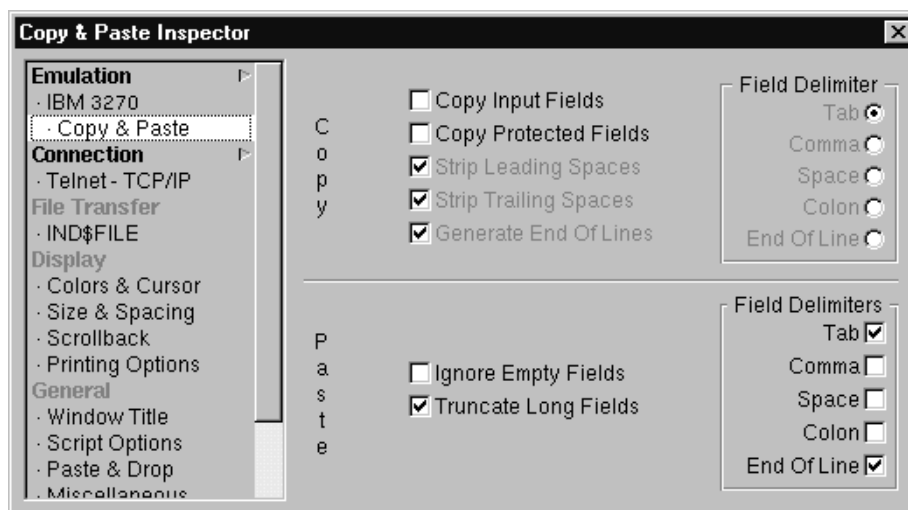
Bold	Protected	Color
		green
	X	blue
X		red
X	X	white

## Ident

*Ident* is the text string that the IBM 3270 session uses to identify itself to the mainframe.

# Copy & Paste Options

Click on *Copy & Paste* in the *Inspector* directory to display the *Copy & Paste Inspector*. This inspector allows you to specify field-based copy and paste behavior for the IBM 3270 emulation.



## Copy Options

### Copy Input Fields & Copy Protected Fields

If you check *Copy Input Fields* and do not check *Copy Protected Fields*, you can copy input fields, but cannot copy protected fields.

If you check *Copy Protected Fields* and do not check *Copy Input Fields*, you can copy protected fields, but cannot copy input fields.

If you check both *Copy Input Fields* and *Copy Protected Fields*, you can copy both input and protected fields.

If you check neither *Copy Input Fields* nor *Copy Protected Fields*, you can still copy input and protected fields, but the following copy formatting options do not apply.

### Strip Leading Spaces

Strips the leading space from each field as it is copied from the terminal window.

### Strip Trailing Spaces

Strips the trailing space from each field as it is copied from the terminal window.

### Generate End of Lines

If you want a newline generated at the end of each entire row you copy, check *Generate End Of Lines*.

## Field Delimiters

Select the field delimiter you want to be inserted between fields:

- *Tab*
- *Comma*
- *Space*
- *Colon*
- *End of Line.*

## Paste Options

### Ignore Empty Fields

Select *Ignore Empty Fields* to cause the paste operation to ignore empty fields in the pasteboard.

For example, if you've selected comma (",") as one of your field delimiters that the paste operation recognizes and the pasteboard contains:

**first,,,second,,third,**

it will be equivalent to:

**first,second,third,**

### Truncate Long Fields

If you check *Truncate Long Fields*, then, when a pasteboard field is longer than the 3270 input field you're pasting it into, the paste operation ignores the remaining characters in the pasteboard field. It then pastes the next pasteboard field into the next 3270 input field.

If you uncheck *Truncate Long Fields*, then, when a pasteboard field is longer than the 3270 input field you're pasting it into, the paste operation pastes the remaining characters into the next 3270 input fields. It then pastes the next pasteboard field into the next 3270 input field.

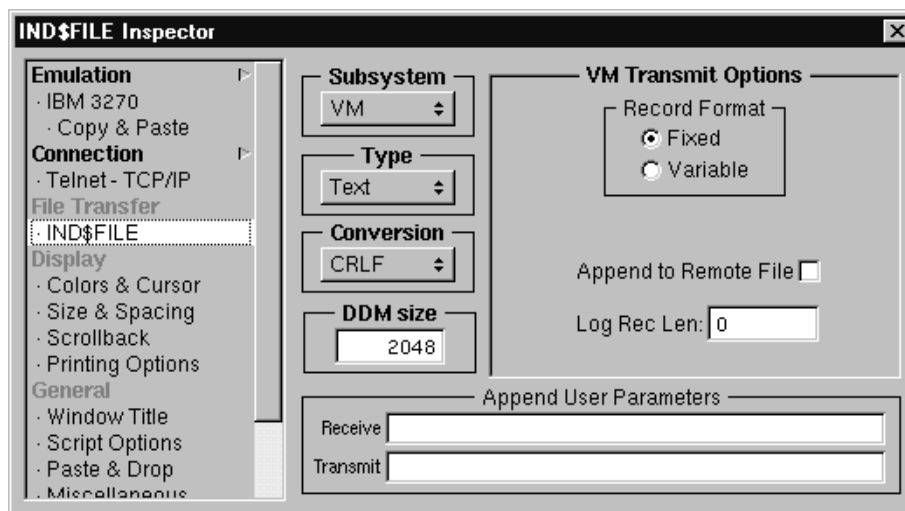
## Field Delimiters

Specify the characters in the pasteboard you want the paste operation to recognize as field delimiters:

- *Tab*
- *Comma*
- *Space*
- *Colon*
- *End of Line*

# IND\$FILE Transfer

Click on *File Transfer/IND\$FILE* in the *Inspector* directory to display the *IND\$FILE Inspector*.



## Subsystem

Select the subsystem that the remote host is running:

- VM
- TSO
- CICS

## Type

- *Text* causes the remote subsystem to translate ASCII encoded text to its EBCDIC equivalent.
- *Binary* transfers files without translation.

## Conversion

- *CRLF* causes the remote subsystem to convert the carriage return/line feed sequence in the transmitted text file to the logical record end in the remote dataset.
- *None* transfers files without conversion.

## Append User Parameters

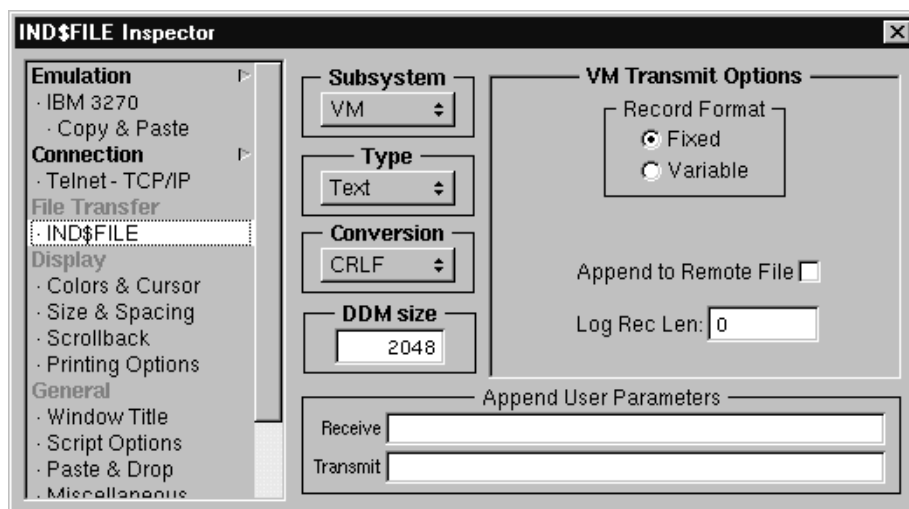
### Receive

Enter any additional parameters for receiving files in the *Receive* text field.

### Transmit

Enter any additional parameters for transmitting files in the *Transmit* text field.

## VM Transmit Options



### Record Format

Specify either fixed length or variable length records.

- *Fixed* sets the **RECFM F** option.
- *Variable* sets the **RECFM V** option.

### Logical Record Length

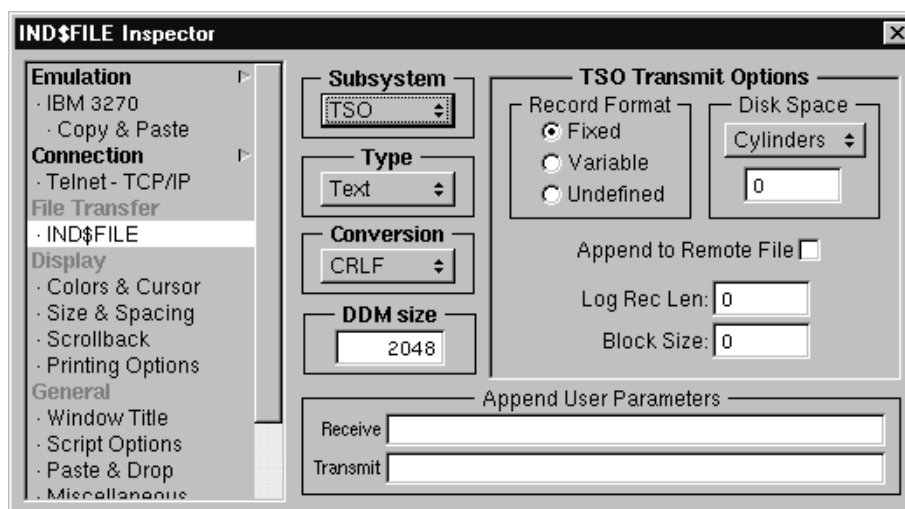
Specify the logical record length in bytes of the dataset on the remote host. This sets the **LRECL n** option.

### Append to remote filename

Check *Append to remote filename* to append the file to the end of the existing dataset on the remote host. Checking *Append to remote filename* selects the **APPEND** option; this causes the remote subsystem to use the existing dataset's attributes instead of the *Record Format* and *Logical Record Length*.



## TSO Transmit Options



### Record Format

Specify fixed length, variable length, or unknown length records.

- *Fixed* sets the **RECFM (F)** option.
- *Variable* sets the **RECFM (V)** option.
- *Unknown* sets the **RECFM (U)** option.

### Disk Space

Specify the disk space the subsystem allocates for the new dataset. Enter the number and type of disk units to allocate. This sets the **SPACE(n,type)** option.

### Logical Record Length

Specify the logical record length in bytes of the dataset on the remote host. This sets the **LRECL(n)** option.

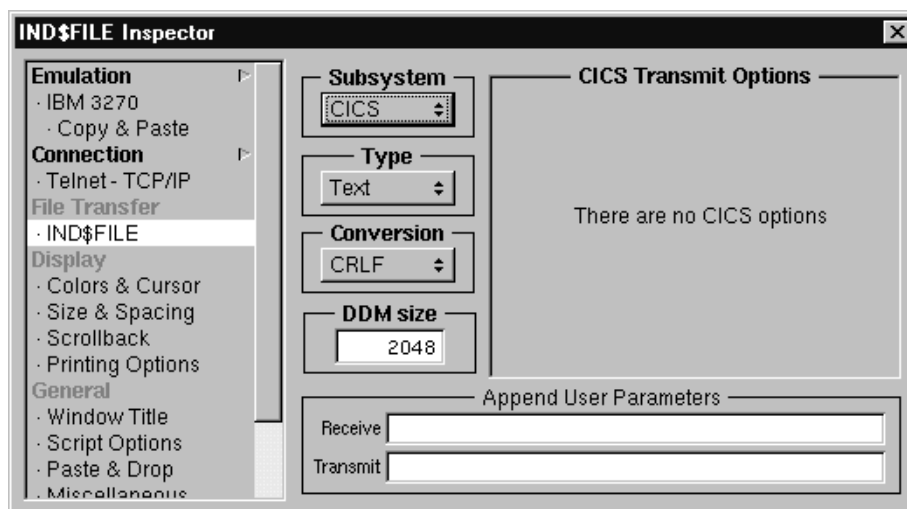
### Block Size

Specify the block size in bytes for the dataset on the remote host. This sets the **BLOCKSIZE(n)** option.

### Append to remote filename

Check *Append to remote filename* to append the file to the end of the existing dataset on the remote host. Checking *Append to remote filename* selects the **APPEND** option; this causes the remote subsystem to use the existing dataset's attributes instead of *Record Format*, *Logical Record Length*, and *Block Size*.

## CICS Transmit Options



There are no special options for the CICS subsystem.

## Key Macros

### IBM 3270 Edit

The following key macros are defined in the *IBM 3270 Edit Key Palette (Tools->Key Map)*:

- tab
- backtab
- left2
- left
- right
- right2
- dup
- fieldmark
- up2
- up
- down
- down2
- home
- newline
- backspace
- reset
- clear
- insert
- delete
- cursor-select
- field-end
- erase-to-eof

erase-input  
erase-field

Additionally the following key macros are defined:

attn  
print-screen

## IBM 3270 Function

The following key macros are defined in the *IBM 3270 Function* key palette (*Tools->Key Map*):

pf1, pf2, pf3, pf4, pf5, pf6, pf7, pf8, pf9, pf10, pf11, pf12,  
pf13, pf14, pf15, pf16, pf17, pf18, pf19, pf20, pf21, pf22, pf23, pf24,  
pf25, pf26, pf27, pf28, pf29, pf30, pf31, pf32, pf33, pf34, pf25, pf36

enter  
pa1, pa2, pa3  
sysreq

# Key Panels

The following *Key Panels* are defined for IBM 3270 emulations

## IBM 3270 Function

IBM 3270 PF Keys											
PF1	PF2	PF3	PF4	PF5	PF6	PF7	PF8	PF9	PF10	PF11	PF12
PF13	PF14	PF15	PF16	PF17	PF18	PF19	PF20	PF21	PF22	PF23	PF24

## IBM 3270 Special

3270 Special		
PA1	PA2	PA3
DUP	FM	␣
↵	Reset	TReq

## IBM 3270 Editing

3270 Editing		
Home	Clear	Newline
Insert	Delete	Select
Erase		
EOF	Input	Field

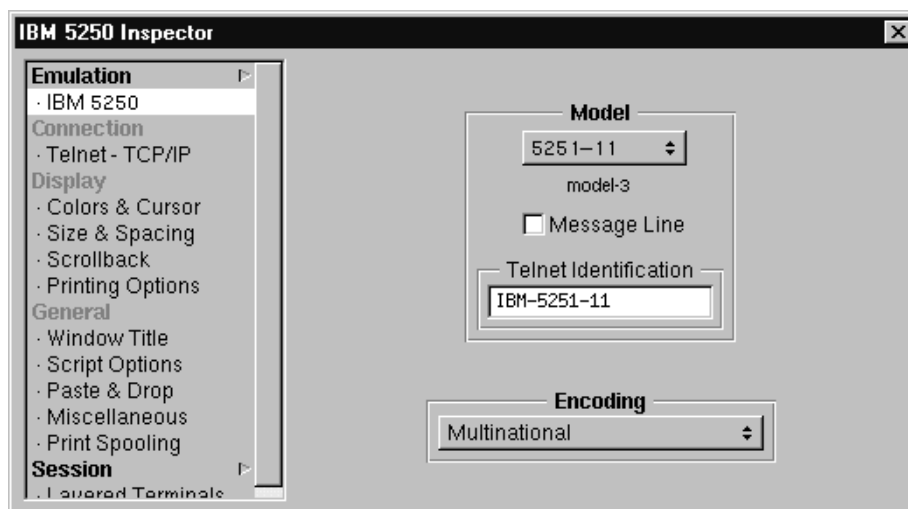
# References

The following manuals on the 3270 are available from IBM:

*3270 Information Display System  
Data Stream Programmer's Reference*  
GA23-0059-07

*3270 Information Display System  
D3274 Control Unit Description and Programmer's Guide*  
GA23-0061-2

## Emulation Attributes



### Model

Select the following IBM 5250 terminal models by dragging and releasing the option list in the *Model* box.

Check the *Message Line* box if you want to display the host's message line.

The following table lists the attributes of the various IBM 5250 models:

**Table 4:**

Model	Columns	Rows	Color/ Mono
3179-2	80	24	color
3180-2	132	27	mono- chrome
3180-2C	132	27	color
5251-11	80	24	mono- chrome
5291-1	80	24	mono- chrome
5292-2	80	24	color

## Ident

*Ident* is the text string that the IBM 5250 session uses to identify itself to the mainframe.

## Encoding

Select either *Multinational* or *U.S., Canada, Netherlands* encoding for the keyboard.

Key Macros

(EQ 1)

# IBM 5250 Edit

The following key macros are defined in the *IBM 5250 Edit* key palette (*Tools->Key Map*):

tab  
backtab  
left2  
left  
right  
right2  
dup  
up2  
up  
down  
down2  
home  
newline  
backspace  
reset  
clear  
insert  
delete  
cursor-select  
field-end  
erase-to-eof  
erase-input  
erase-field

## IBM 5250 Function

The following key macros are defined in the *IBM 5250 Function Key Palette* (*Tools->Key Map*):

f1, f2, f3, f4, f5, f6, f7, f8, f9, f10, f11, f12,  
f13, f14, f15, f16, f17, f18, f19, f20, f21, f22, f23, f24

**enter**  
**sysreq**  
**help**  
**attn**  
**roll-up**

roll-down  
print  
field-plus  
field-exit  
field-minus

## Key Panels

The following *Key Panels* are defined for IBM 5250 emulations:

### IBM 5250 Function

IBM 5250 Function Keys											
F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12
F13	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23	F24

### IBM 5250 Special

5250 Special	
Help	Print
Reset	Attn

### IBM 5250 Editing

5250 Editing		
Home	Roll Up	Clear
	Roll Down	
Field+	FieldExit	Field-
DUP	Insert	Delete
Erase		
EOF	Input	Field

## References

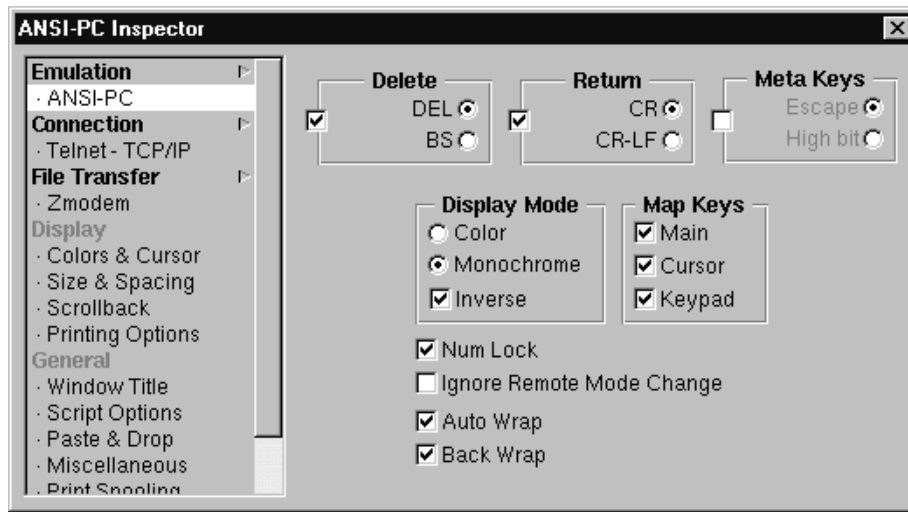
The following manuals on the 5250 are available from IBM:

*IBM 5250 Information Display System  
Functions Reference Manual*  
SA21-9247-6

*IBM 5291 Models 1 and 2 Display Station  
Operators Guide  
GA21-9409-2*



## Emulation Attributes



### Delete

Unchecking the *Delete* box causes the keycode value for the Delete key to be the value defined for it in *Tools - > Key Map*.

Checking the *Delete* box causes the keycode value for the Delete key to be one of the following:

- *DEL* the DEL (delete) character (0x7F).
- *BS* the BS (backspace) character (0x08).

### Return

Unchecking the *Return* box causes the keycode value for the Return key to be the value defined for it in *Tools - > Key Map*.

Checking the *Return* box causes the keycode value for the Return key to be one of the following values:

- *CR* the CR (carriage return) character (0x0D).
- *CR-LF* both the CR and the LF (line feed) characters (0x0D and 0x0A).

### Meta Keys

Unchecking the *Meta Keys* box makes the Alternate key function normally.

Checking the *Meta Keys* box enables one of the following options:

- *Escape* causes the Alternate key to function as the EMACS meta key. If you hold down the Alternate key and press, for example, the **a** key, Cables will send the ESC (escape) character (0x1B) and then the **a** character (0x61).
- *High Bit* causes the Alternate key to set the high bit in the character code. If you hold down the Alternate key and press, for example, the **a** key, Cables will send the code 0xE1.

## Display Mode

Select *Color* for a color display; select *Monochrome* for a monochrome display. If you select *Monochrome*, the terminal session will ignore all commands to set colors.

Check the *Inverse* box to display black as white and white as black. This is useful for viewing white-on-black applications as black-on-white.

## Map Keys

The *Map Keys* options allow you to control the key mapping defined in the *Key Map* inspector (*Tools->Key Map*) for the following keypads.

NOTE: Any values you've mapped to keys using the Key Map inspector override the effects of the following options.

### Main

Checking *Main* enables the default *Key Map* mapping for the keys in the main keypad.

Unchecking *Main* disables the default *Key Map* mapping for the keys in the main keypad enabling the default NEXTSTEP mapping.

### Cursor

Checking *Cursor* enables the default *Key Map* mapping for the keys in the cursor keypad.

Unchecking *Cursor* disables the default *Key Map* mapping for the keys in the cursor keypad enabling the default NEXTSTEP mapping.

### Keypad

Checking *Keypad* enables the default *Key Map* mapping for the keys in the numeric keypad.

Unchecking *Keypad* disables the default *Key Map* mapping for the keys in the numeric keypad enabling the default NEXTSTEP mapping.

## Num Lock

Checking *Num Lock* causes the numeric keys on the numeric keypad to return numeric values.

## Ignore Remote Mode Change

If you check *Ignore Remote Mode Change*, Cables will ignore escape sequences from the application to change attributes such as the number of columns and color/monochrome mode.

## Auto Wrap

Selecting *Auto Wrap* causes text entered past the last column to wrap onto the next row.

If you don't check *Auto Wrap*, then, when text is displayed past the last column of the current row, the cursor will stop at the last column; each new character will overwrite the last one.

## Back Wrap

If you check the *Back Wrap* option and use the Delete key to delete text that has wrapped over multiple rows, Cables will backtrack the cursor over the wrapped line. If you do not check *Back Wrap*, then the Delete key will still delete text, but the cursor will stop at the first column of the current row.

Cables selects *Back Wrap* as the default because it is a generally desirable feature (which might not have been supported in the emulated terminal).

Note that *Back Wrap* only affects the behavior of backspace (BS) keycode.

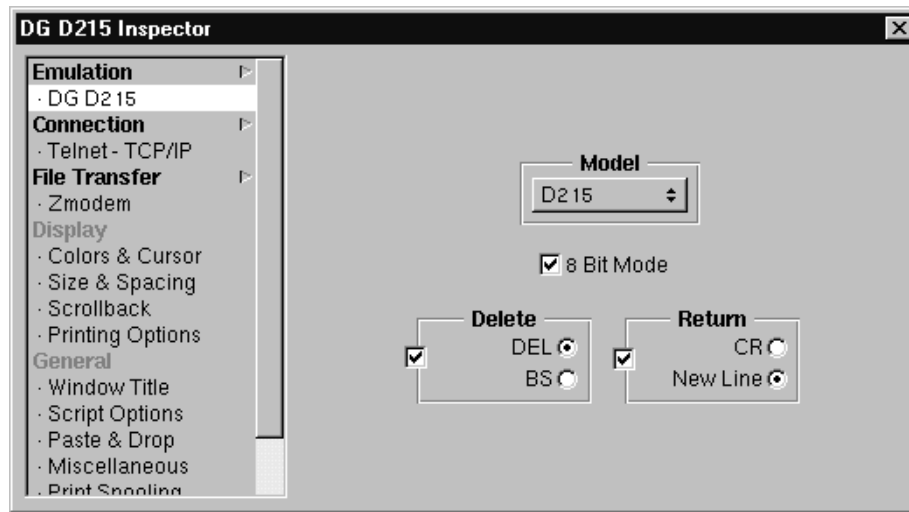
## References

For more information specific to the ANSI-PC terminal see:

Woodcock, Joanne. *MS-DOS 6.0 Companion*. Microsoft Press. ISBN 1-55615-550-6.



## Emulation Attributes



### Model

Dragging and releasing on a model id from the *Model* option list sets the model id the session returns in response to the *Read Model Id* command. Cables supports D210, D211, D214, and D215 model ids.

### 8 Bit Mode

Check the *8 Bit Mode* box to allow the session to display and transmit multinational characters in the upper 128 characters.

### Delete

Unchecking the *Delete* box causes the keycode value for the Delete key to be the value defined for it in *Tools->Key Map*.

Checking the *Delete* box causes the keycode value for the Delete key to be one of the following:

- *DEL* the DEL (delete) character (0x7F).
- *BS* the BS (backspace) character (0x08).

### Return

Unchecking the *Return* box causes the keycode value for the Return key to be the value defined for it in *Tools->Key Map*.

Checking the *Return* box causes the keycode value for the Return key to be one of the following:

- *CR*                      the CR (carriage return) character (0x0D).
- *New Line*              the LF (line feed) character (0x0A).

**NOTE:** Some modems require the CR key code when they're in command mode. If you have the Return key mapped to *New Line*, then you'll have to use **^M** or **shift-return** to generate a CR.

## Key Macros

### DG D215 Edit

The following key macros are defined in the *DG D215 Edit Key Palette (Tools->Key Map)*:

up  
left  
down  
right

shift-up  
shift-left  
shift-down  
shift-right

control-up  
control-left  
control-down  
control-right

c1  
c2  
c3  
c4

shift-c1  
shift-c2  
shift-c3  
shift-c4

### DG D215 Function

The following key macros are defined in the *DG D215 Function* key palette (*Tools->Key Map*):

f1, f2, f3, f4, f5, f6, f7, f8, f9, f10, f11, f12, f13, f14, f15

shift-f1, shift-f2, shift-f3, shift-f4, shift-f5, shift-f6, shift-f7, shift-f8,  
shift-f9, shift-f10, shift-f11, shift-f12, shift-f13, shift-f14, shift-f15

control-f1, control-f2, control-f3, control-f4, control-f5, control-f6, control-f7, control-f8,  
control-f9, control-f10, control-f11, control-f12, control-f13, control-f14, control-f15

shift-control-f1, shift-control-f2, shift-control-f3, shift-control-f4, shift-control-f5,  
shift-control-f6, shift-control-f7, shift-control-f8, shift-control-f9, shift-control-f10,  
shift-control-f11, shift-control-f12, shift-control-f13, shift-control-f14, shift-control-f15

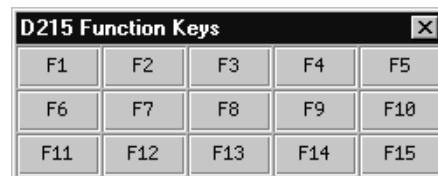
## Key Panels

The following *Key Panels* are defined for DG D215 emulations:

### D215 Screen



### D215 Function



## References

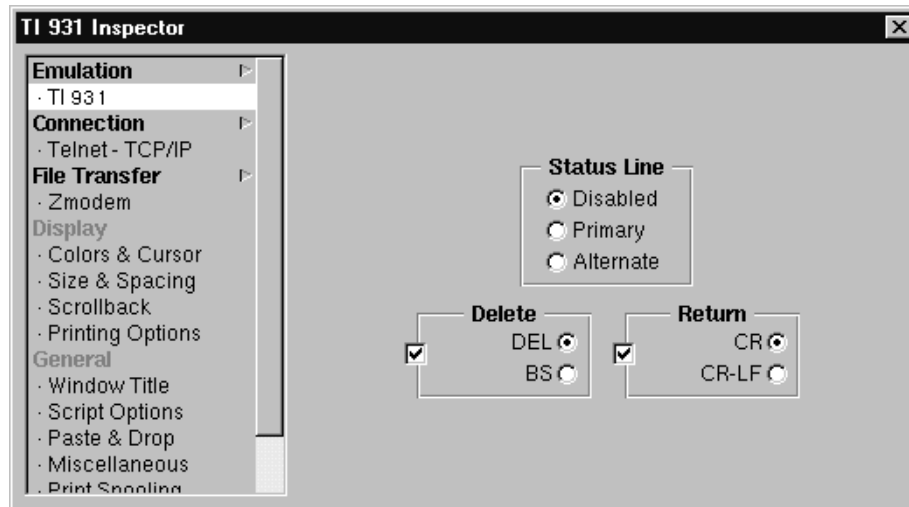
The following manual on the Dasher D200 series is available from Data General Corporation:

Dasher D210 and D211 Display Terminal User's Manual





## Emulation Attributes



### Status Line

- *Disabled* disables the status line. (Restores the original number of rows.)
- *Primary* enables the bottom row of the screen as the primary status line. The application can write to the left 40 characters of the primary status line; the terminal reserves the right 40 characters for status information.
- *Alternate* enables the bottom row of the screen as the alternate status line. The application can write to all 80 characters of the alternate status line.

Enabling a status line causes the screen to become shorter by one row.

### Delete

Unchecking the *Delete* box causes the keycode value for the Delete key to be the value defined for it in *Tools->Key Map*.

Checking the *Delete* box causes the keycode value for the Delete key to be one of the following:

- *DEL* the DEL (delete) character (0x7F).
- *BS* the BS (backspace) character (0x08).

## Return

Unchecking the *Return* box causes the keycode value for the Return key to be the value defined for it in *Tools->Key Map*.

Checking the *Return* box causes the keycode value for the Return key to be one of the following:

- *CR*                      the CR (carriage return) character (0x0D).
- *CR-LF*                  the CR and the LF (line feed) characters (0x0D and 0x0A) .

# Key Macros

## TI 931 Function

The following key macros are defined in the *TI 931 Function* key palette (*Tools->Key Map*):

f1, f2, f3, f4, f5, f6, f7, f8, f9, f10, f11, f12

cmd  
blank-orange  
blank-gray  
erase-field  
erase-input  
field  
del-char  
ins-char  
print

shift-f1, shift-f2, shift-f3, shift-f4, shift-f5, shift-f6,  
shift-f7, shift-f8, shift-f9, shift-f10, shift-f11, shift-f12

shift-cmd  
shift-blank-orange  
shift-blank-gray  
shift-erase-field  
shift-erase-input  
shift-field  
shift-del-char  
shift-ins-char  
shift-print

## TI 931 Keypad/Misc.

The following key macros are defined in the *TI 931 Keypad/Misc.* key palette (*Tools->Key Map*):

print-screen

left  
up

home  
skip  
enter  
down  
right  
  
shift-esc  
shift-kp-tab  
  
alt-cmd  
alt-blank-orange  
alt-erase-field  
alt-erase-input  
alt-skip  
  
alt-kp-space  
alt-kp-tab  
alt-3  
alt-kp-equal  
alt-kp-plus

## Key Panel

The following *Key Panel* is defined for TI 931 emulations:

### TI 931 Function

TI 931 FKeys					
F1	F2	F3	F4	F5	F6
F7	F8	F9	F10	F11	F12
CMD		Orange		Gray	
Del Char		Erase Field		Print	
Ins Char		Erase Input		Field	

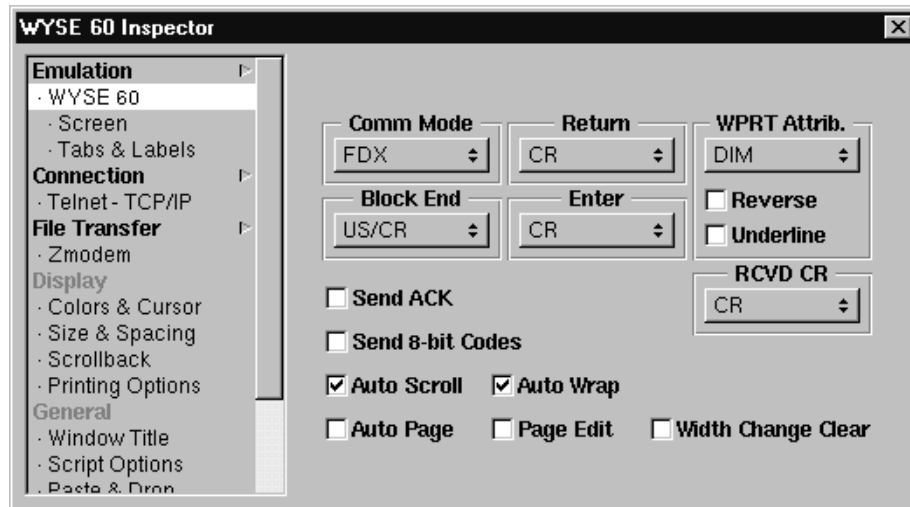
## References

The following Texas Instruments manual describes the TI 931:

Model 931 Video Display Terminal General Description



## Emulation Attributes



### Communication Mode

Drag and release the *Comm. Mode* button to select the terminal's communication mode.

- *FDX* Full Duplex mode.
- *BLK* Block mode.
- *HDX* Half duplex mode.
- *HALF BLK* Half duplex block mode.

### Return

Drag and release the *Return* button to select the character sent by the Return key.

- *CR* the CR (carriage return) character (0x0D).
- *CR-LF* the CR character and the LF (line feed) characters (0x0D and 0x0A).
- *TAB* the TAB character (0x09).

### Enter

Drag and release the *Enter* button to select the character sent by the Enter key.

- *CR* the CR (carriage return) character (0x0D).
- *CR-LF* the CR character and the LF (line feed) characters (0x0D and 0x0A).
- *TAB* the TAB character (0x09).

## Block End

Drag and release the *Block End* button to select the characters sent at the end of a Line/Block.

- *US/CR* the US character (0x1F) as the end of line marker and the CR (carriage return) character (0x0D) as the end of block marker.
- *CRLF/ETX* the CR (carriage return) and LF (line feed) characters (0x0D0A) as the end of line marker and the ETX character (0x03) as the end of block marker.

## Write PRotected Attributes

Drag and release the *WPRT Attrib.* button to select the attribute for the write protected fields.

- *DIM*
- *NORMAL*
- *INVISIBLE*

## Reverse

Check *Reverse* to select the reverse video attribute for write protected fields.

## Underline

Check *Underline* to select the underline attribute for write protected fields.

## ReCeived Carriage Return

- *CR* When the terminal receives a CR from the host, it moves the cursor to the beginning of the line.
- *CRLF* When the terminal receives a CR from the host, it moves the cursor to the beginning of the next line.

## Send ACK

Checking *Send ACK* makes the terminal send an ASCII ACK character after certain commands.

## 8-bit Codes

Checking *8-bit Codes* allows the terminal to send and receive 8-bit data.

## Auto Scroll

Check *Auto Scroll* to cause the terminal to scroll down one line when it reaches the bottom of the window. Otherwise, the terminal repositions the cursor at the top of the window.

## Auto Wrap

Check *Auto Wrap* to cause text entered past the last column to wrap onto the next row.

If you don't check *Auto Wrap*, then, when text is displayed past the last column of the current row, the cursor will stop at the last column; each new character will overwrite the last one.

**Auto Page**

Check *Auto Page* to cause the terminal to bring up the next page when the cursor reaches the bottom of a page.

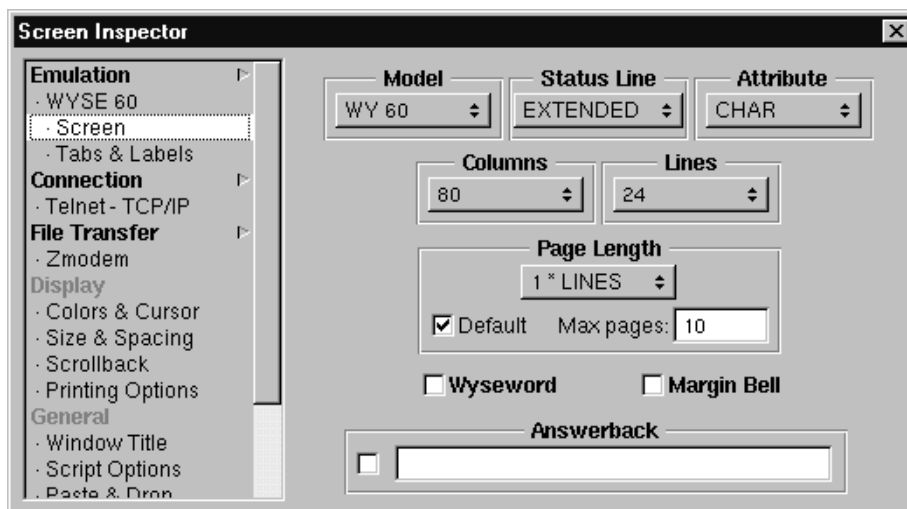
**Page Edit**

Check *Page Edit* to enable page edit mode.

**Width Change Clear**

Checking *Width Change Clear* causes the terminal to clear the screen when it receives a width change command.

# Screen



## Model

Select one the WYSE terminal models by dragging and releasing the *Model* button.

- *WY60* selects the WY-60 terminal.
- *WY50+* selects WY-50+ terminal (which implements a superset of WY-50 functions).

## Status Line

- *Standard* displays the standard status line.
- *Extended* displays the extended status line.
- *Off* displays no status line.

## Attribute

- *Char* the set attribute command applies to subsequent characters.
- *Line* Attribute applies to the line.
- *Page* Attribute applies to the page.



## Columns

- *80* 80 columns.
- *132* 132 columns.
- *Econ-80* The *Econ-80* option allows 80 columns with more pages.

## Lines

- *24* 24 data lines plus status line and label line.
- *42* 42 data lines plus status line and label line.
- *25* 25 data lines plus status line (no label line).
- *43* 43 data lines plus status line (no label line).

## Page Length

Select the length of a display memory page from the following options:

- *1 \* LINES* the number of lines selected in the *Lines* option.
- *2 \* LINES* twice the number of lines selected in the *Lines* option.
- *4 \* LINES* four times the number of lines selected in the *Lines* option.
- *\** the number of lines selected in the *Lines* option plus a second page containing the rest of the lines in memory.

## Answerback

Check *Answerback* to cause the terminal to transmit the answerback string when it receives an ASCII ENQ character.

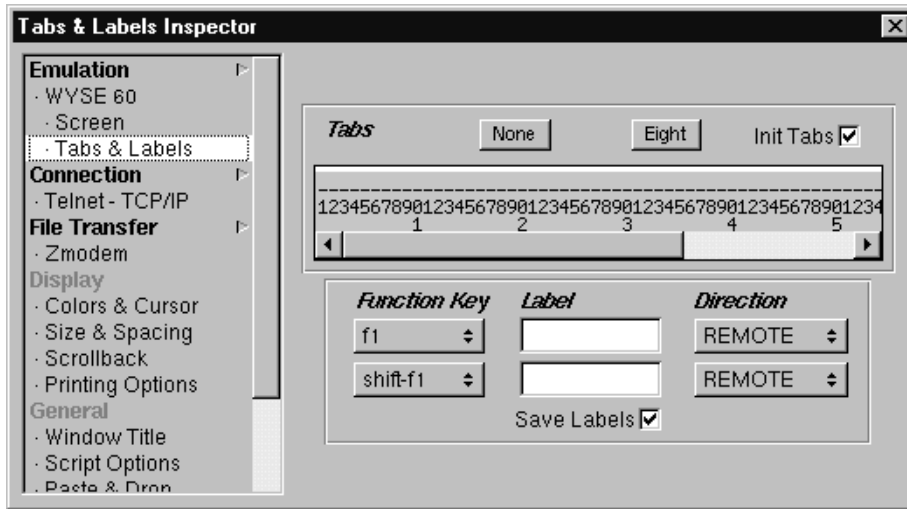
## Wyseword

Check *Wyseword* to allow certain keys to send WordStar-compatible codes.

## Margin Bell

Check *Margin Bell* to cause the margin bell to ring when cursor reaches the margin.

# Tabs & Labels



## Tabs

### None

Click *None* to clear all tab settings.

### Eight

Click *Eight* to set tabs every eighth column, i.e. at cols. 9, 17, 25, etc.

### Init Tabs

Check *Init Tabs* if you want Cables to load the tab settings from a .cable file.

You can insert a tab by positioning the cursor in the tab ruler and clicking. To delete an existing tab, position the cursor over it and click.

## Function Key

### Save Labels

Check *Save Labels* if you want Cables to save the Label settings when you save the configuration.

### Function Key

Select the function key you wish to label by dragging and releasing the options list.

### Label

Enter a label for the function key in the text box and press Return.

### Direction

Select where the terminal sends data when you press a function key.

- *REMOTE*      Data goes only to the host regardless of the terminal's communication mode.
- *LOCAL*        Data goes only to the terminal.
- *NORMAL*       Data goes to the host and the terminal depending on the terminal's communication mode.

# Key Macros

## WY60 Function

The following key macros are defined in the *WY60 Function Key* palette (*Tools->Key Map*):

**f1, f2, f3, f4, f5, f6, f7, f8, f9, f10, f11, f12, f13, f14, f15, f16**

**shift-f1, shift-f2, shift-f3, shift-f4, shift-f5, shift-f6, shift-f7, shift-f8,  
shift-f9, shift-f10, shift-f11, shift-f12, shift-f13, shift-f14, shift-f15, shift-f16**

**control-f1, control-f2, control-f3, control-f4,  
control-f5, control-f6, control-f7, control-f8,  
control-f9, control-f10, control-f11, control-f12,  
control-f13, control-f14, control-f15, control-f16**

**shift-control-f1, shift-control-f2, shift-control-f3, shift-control-f4,  
shift-control-f5, shift-control-f6, shift-control-f7, shift-control-f8,  
shift-control-f9, shift-control-f10, shift-control-f11, shift-control-f12,  
shift-control-f13, shift-control-f14, shift-control-f15, shift-control-f16**

## WY60 Edit

The following key macros are defined in the *WY60 Edit Key* palette (*Tools->Key Map*):

**esc  
tab  
backspace  
delete  
send  
prev-page  
return**

**shift-esc  
shift-tab  
shift-backspace  
shift-delete  
print  
next-page  
shift-return**

## WY60 Local

The following key macros are defined in the *WY60 Local Key* palette (*Tools->Key Map*):

**local-hold**  
**local-setup**  
**local-hard-reset**  
**local-toggle-blk**  
**local-data-port**  
**local-aux-print**  
**local-monitor**  
**local-wyseword**  
**local-clear-page**  
**local-split-screen**  
**local-raise-split**  
**local-lower-split**  
**local-roll-up**  
**local-roll-down**  
**local-print**  
**local-break**  
**local-page-0**  
**local-page-1**  
**local-page-2**  
**local-page-3**  
**local-page-4**  
**local-page-5**  
**local-page-6**  
**local-page-7**  
**local-page-8**  
**local-page-9**  
**local-page-next**  
**local-page-prev**  
**local-shft-label-line**

---

## WY60 Keypad

The following key macros are defined in the *WY60 Keypad* palette (*Tools->Key Map*):

**ins-line**  
**del-line**  
**clr-scrn**  
**insert**

**ins-char**  
**del-char**  
**clr-line**  
**replace**

**kp0, kp1, kp2, kp3, kp4, kp5, kp6, kp7, kp8, kp9, kp-minus, kp-period, kp-enter, kp-comma**

**shift-kp0, shift-kp1, shift-kp2, shift-kp3, shift-kp4,**  
**shift-kp5, shift-kp6, shift-kp7, shift-kp8, shift-kp9,**  
**shift-kp-minus, shift-kp-period, shift-kp-comma, shift-kp-enter**

**up, left, down, right, home**

**shift-up, shift-left, shift-down, shift-right, shift-home**

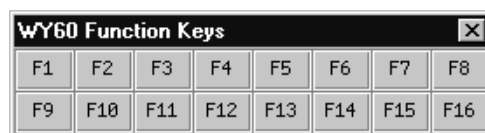
# Key Panels

The following *Key Panels* are defined for the WY-60/50+ emulations:

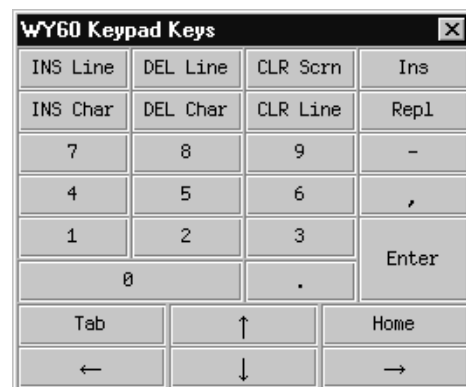
## WY60 Edit



## WY60 Function



## WY60 Keypad



# WY60 Local Keys

WY60 Local Keys			
Hold	SetUp	Reset	BLK
Data Port	Aux Print	Monitor	Wyseword
Clear Page	Split Screen	Raise Split	Lower Split
Roll Up	Roll Down	Print	Break
Page 0	Page 1	Page 2	Page 3
Page 4	Page 5	Page 6	Page 7
Page 8	Page 9	Page Next	Page Prev
Shifted Label Line			



## Assigning Line Attributes to Colors

The WY-60 terminal allows you to assign line attributes with the command **ESC G *lattr***. Some of these line attributes are represented in Cables as **colors**. The following line attributes are represented by these default colors:

**Table 5:**

lattr	Line Attribute	Default Color
G	Normal background	Blue
H	Bold background	Red
I	Invisible	White
J	Dim background	Yellow

You can remap these default colors. To do this, bring up the *Colors & Cursor* inspector and drag the *Options* list to either Blue, Red, Monochrome, or Yellow. Click on the border of the *Background* color well. Choose a color from the *Colors* palette and drag it into the *Background* color well. Now when you assign line attributes G, H, I or J, the background colors will be the colors you chose. In the same way, you can also change the foreground color displayed for these same line attribute.

## References

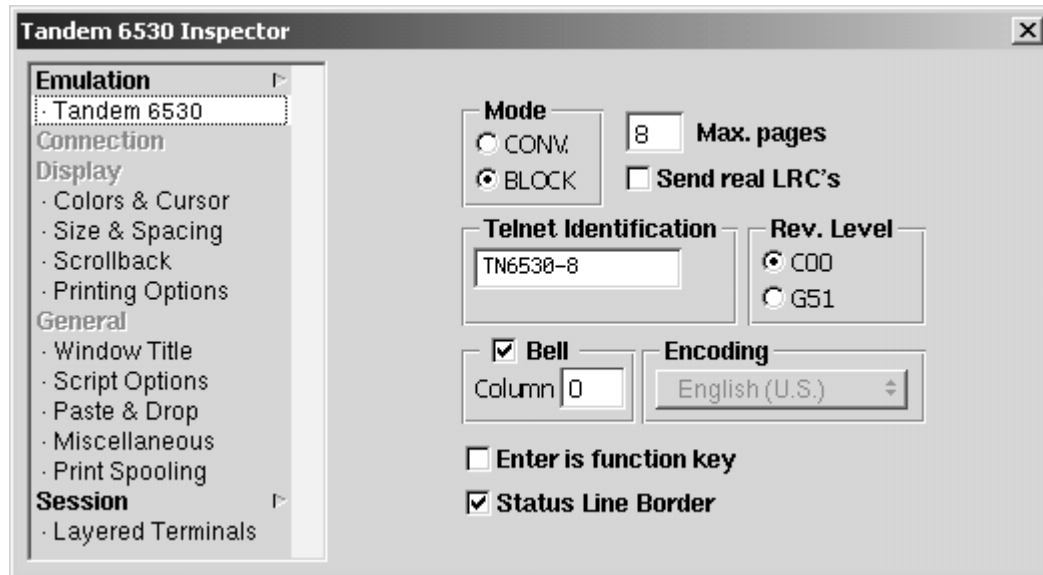
The following manuals on the WY-60/50+ terminals are available from Wyse Technology:

*WY-60 User's Guide* (880259-02 Rev. B)

*WY-60 Programmer's Guide* (880261-01 Rev. A)



## Emulation Attributes



### Mode

*Mode* is the initial mode of the terminal. Select

- *CONV.* if the host expects the terminal to start in Conversational mode (default)
- *BLOCK* if the host expects the terminal to start in Block mode

### Max. pages

*Max. pages* specifies the size of the terminal's memory in 24×80 character pages.

### Send real LRC's

If you enable *Send real LRC's*, the terminal calculates and sends real LRC's for all transmissions to the host. If you disable *Send real LRC's*, the terminal sends **NUL** characters (0x00) as LRC's for all transmissions to the host.

### Telnet Identification

*Telnet Identification* is the ASCII string the terminal uses to identify itself during the terminal type option negotiation with the host.

---

## Rev. Level

*Rev. Level* determines the full revision level that the terminal returns for the *Read Full Revision Level* ( **ESC \_** ) command and the major revision level for the *Read Full Terminal Status* ( **ESC ^** ). The Rev. Level does not otherwise affect the behavior of the terminal.

## Bell

Check the *Bell* box to enable the terminal's bell.

## Bell Column

The terminal beeps if, while typing, you move the cursor to the column specified in the *Bell Column* field (and the *Bell* is enabled).

Values in the range 1 to 80 are valid columns. A value of 0 disables the *Bell Column*.

## Enter is function key

If you enable *Enter is function key*, the terminal treats the **Enter** key (shifted as well as unshifted) as a function key.

## Status Line Border

If you enable *Status Line Border*, the terminal displays a line between the upper 24 rows and the status line.

## Encoding

*Encoding* is the terminal's character set encoding.

NOTE: The encoding is currently fixed to *English (U.S.)*.

# Key Macros

## Tandem 6530 Function

The following key macros are defined in the *Tandem 6530 Function* palette (*Tools->Key Map*):

<b>f1</b>	<b>shift-f1</b>		
<b>f2</b>	<b>shift-f2</b>		
<b>f3</b>	<b>shift-f3</b>		
<b>f4</b>	<b>shift-f4</b>		
<b>f5</b>	<b>shift-f5</b>		
<b>f6</b>	<b>shift-f6</b>		
<b>f7</b>	<b>shift-f7</b>		
<b>f8</b>	<b>shift-f8</b>		
<b>f9</b>	<b>shift-f9</b>		
<b>f10</b>	<b>shift-f10</b>		
<b>f11</b>	<b>shift-f11</b>		
<b>f12</b>	<b>shift-f12</b>		
<b>f13</b>	<b>shift-f13</b>		
<b>f14</b>	<b>shift-f14</b>		
<b>f15</b>	<b>shift-f15</b>		
<b>f16</b>	<b>shift-f16</b>		
<b>enter</b>	<b>shift-enter</b>		
<b>alt-up</b>	<b>shift-alt-up</b>		
<b>alt-down</b>	<b>shift-alt-down</b>		
<b>page-up</b>			
<b>alt-page-up</b>			
<b>ctrl-insert</b>			
<b>ctrl-delete</b>			
<b>page-down</b>			
<b>alt-page-down</b>			
<b>home</b>		<b>ctrl-home</b>	
<b>end</b>			
<b>left</b>			
<b>right</b>			
<b>up</b>			
<b>down</b>			
<b>enter</b>	<b>shift-enter</b>	<b>ctrl-enter</b>	
<b>insert</b>		<b>ctrl-insert</b>	<b>alt-insert</b>
<b>delete</b>		<b>ctrl-delete</b>	
<b>erase-to-eolf</b>			
<b>erase-to-eop</b>			

# Key Panels

## Function Keys

Tandem 6530 Function Keys																	X
	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	
Shift	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	

## Additional Function Keys

Tandem 6530 Additional Block Mode Function Keys										X
	Enter	Alt ↑	Alt ↓	PgUp	Alt-PgUp	Ctrl-Ins	Ctrl-Del	PgDn	Alt-PgDn	
Shift	Enter	Alt ↑	Alt ↓							

## Edit Keys

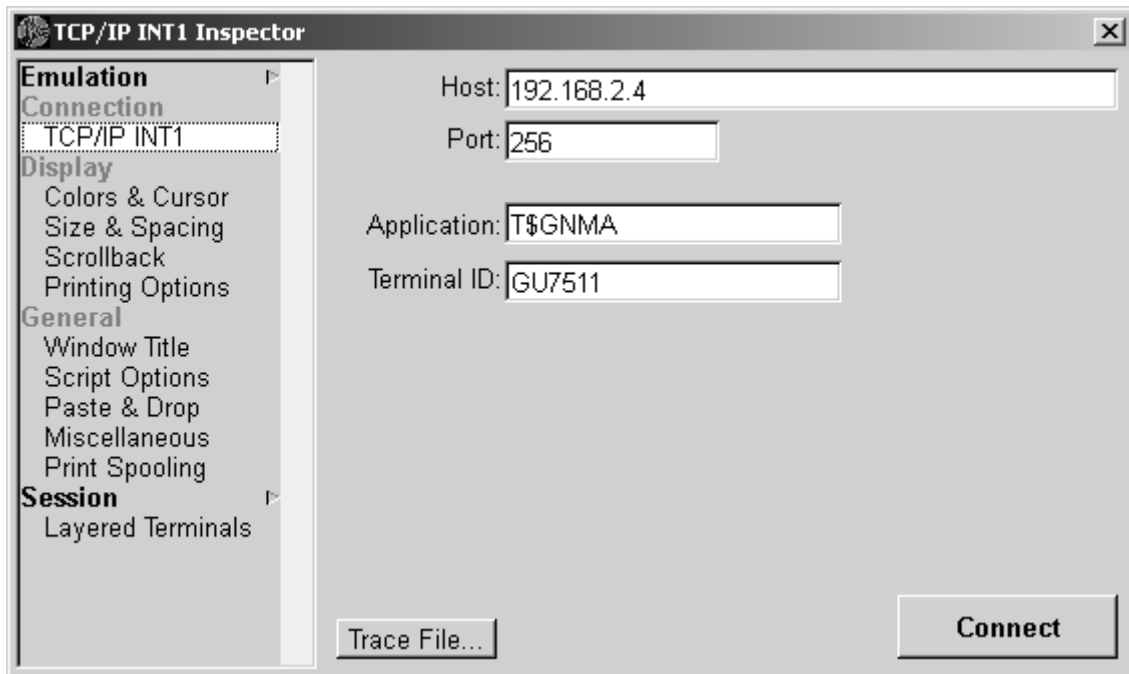
Tandem 6530 Edit Keys			X
→	↑	←	
←	↓	→	
Home	Line/Field Start	Line/Field End	
Enter	Last Line Start	Page End	
Enter Insert Mode	Insert Char	Insert Line	
	Delete Char	Delete Line	
	Erase Line/Field	Erase Page	
Set Tab	Clear Tab	Clear All Tabs	
	BREAK		

# References

The following manual is available from Compaq Computer Corporation:

*6530 Programmer's Guide* (131921 11/96)

## TCP/IP INT1 Connection Attributes



### Host

*Host* is name or IP address of the host (to which you want to connect).

### Port

*Port* is the port number on the *Host*.

### Application

*Application* is the name of the application.

### Terminal ID

*Terminal ID* is a valid terminal identifier for the connection.

---

# Key Macros

## Uniscope Edit

The following key macros are defined in the Uniscope Edit palette (Tools ▶ Key Map):

tab  
backtab  
left  
right  
up  
down  
enter  
message-wait  
backspace  
delete  
clear  
soe  
erase-to-eof  
erase-to-eol  
erase-to-eop

## Uniscope Function

The following key macros are defined in the Uniscope Function palette (Tools ▶ Key Map):

f1, f2, f3, f4, f5, f6, f7, f8, f9, f10, f11, f12, f13, f14, f15, f16, f17, f18, f19, f20, f21, f22

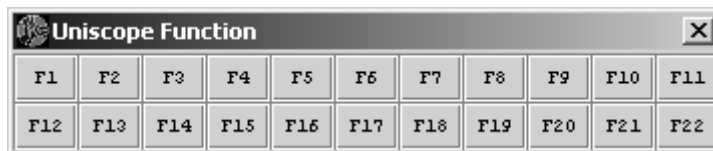


# Key Panels

## Edit Keys



## Function Keys





# INDEX

## Symbols

.cable files 6-1, 6-43

## Numerics

### 8 bit mode

DEC VT102 6-4

DEC VT320 A-2

DG D215 E-1

WYSE WY-60 G-2

## A

ANSI-PC D-1

### answerback

DEC VT102 6-7

DEC VT320 A-4

WYSE WY-60 G-4

application macros 6-46

### Auto Wrap

ANSI-PC D-3

DEC VT102 6-4

DEC VT320 A-2

WYSE WY-60 G-2

## B

### Base Color

IBM 3270 B-2

baud rate 6-9

## C

Cable files 6-1

Cables Light 8-1

Cables Remote API

macros 6-48

Clear (paste buffers) 5-5

### close

confirm 6-24

on disconnect 6-25

### color mode

ANSI-PC D-2

IBM 3270 B-1, C-1

inverse

ANSI-PC D-2

### colors

setting cursor colors 6-13

setting text attribute colors 6-12

### commands

Close 5-4

Edit 5-1

Format 5-1

- Library 5-3
- New 5-3
- Open 5-3
- Quit 5-1
- Revert to Saved 5-4
- Save 5-3, 5-4
- Save As 5-4
- Save To 5-4
- Services 5-1
- Windows 5-1
- configuration files 6-1
- connection trace file
  - rewind 6-10
  - start 6-10
  - stop 6-10
- Copilot
  - macros 6-47
  - related macros 6-48
- cursor keys
  - DEC VT320 A-3
- D
- DEC VT102 6-4
- DEC VT320 A-1, G-1
- Default Session 5-3
- Delete key
  - ANSI-PC D-1
  - DEC VT320 A-4
  - DG D215 E-1
  - TI 931 F-1
  - VT102 6-6
- DG D215 E-1
- Display mode
  - DEC VT320 A-1
- drop options
  - .cable files 6-22
  - filenames 6-23
- DTR
  - dropping 6-10
  - raising 6-10
- E
- EMACS A-3
- EndSpool key 6-47
- Enter key
  - WYSE WY-60 G-1
- F
- file transfer macros 6-48
- file transfer protocols
  - Kermit 6-33

- Xmodem 6-32
- Ymodem 6-31
- Zmodem 6-29
- filename
  - paste options 6-23
  - queue for transmit 6-23
- font
  - resize 6-15
- I
- IBM 3270 B-1
  - session 6-11
  - telnet client 6-11
- IBM 5250 C-1
- Ident string
  - IBM 3270 B-2
  - IBM 5250 C-2
- Ignore Remote Mode Change
  - ANSI-PC D-2
- IND\$FILE
  - transmit options
    - CICS B-8
    - TSO B-7
    - VM B-6
- IND\$FILE file transfer protocol B-5
- Info
  - Preferences
    - Default Session 5-3
- Interpret mode
  - DEC VT320 A-1
- K
- kermi
  - command 6-34
  - file transfer protocol 6-33
- key macros 6-46
  - DG D215
    - edit keys E-2
    - function keys E-2
  - IBM 3270
    - edit keys B-8
    - function keys B-9
  - IBM 5250
    - function keys C-2
  - IBM 5250 edit keys C-2
  - TI 931
    - function keys F-2
    - keypad/misc. keys F-2
  - VT Series Keypad 6-46
  - VT320
    - function keys A-4

- WYSE WY-60
  - edit keys G-6
  - function keys G-6
  - keypad G-7
  - local keys G-7
- key mapping 6-43, 6-44, 6-45, 6-48
  - ANSI-PC
    - cursor keypad D-2
    - main keypad D-2
    - numeric keypad D-2
- key panels 6-50
  - DEC VT102
    - multinational characters 6-51
    - VT102 keypad 6-50
  - DEC VT320
    - edit keys A-5
    - function keys A-5
  - DG D215
    - function keys E-3
    - screen keys E-3
  - IBM 3270
    - editing keys B-10
    - function keys B-10
    - special keys B-10
  - IBM 5250 editing keys C-3
  - IBM 5250 function keys C-3
  - IBM 5250 special keys C-3
  - TI 931
    - function keys F-3
  - WYSE WY-60
    - edit keys G-8
    - function keys G-8
    - keypad G-8
    - local keys G-8

- keypad keys
  - DEC VT320 A-3

## L

- link macros 6-46

- General Keys 6-46

- lpr 6-27

## M

- macros

- application 6-46
  - Cables Remote API 6-48
  - copilot 6-47
  - Copilot-related 6-48
  - file transfer 6-48
  - key 6-46
  - link 6-46

- session 6-47
- spool 6-47
- terminal 6-47
- main window title 6-19
- mapping keys 6-43, 6-44, 6-45, 6-48
- meta keys
  - DEC VT320 A-3
  - VT102 6-6
- miniwindow
  - title 6-19
- model identification
  - DEC VT 102 6-4
  - DEC VT320 A-1
  - DG D215 E-1
  - IBM 3270 B-1
  - IBM 5250 C-1
  - WYSE WY-60 G-3
- monochrome mode
  - ANSI-PC D-2
  - IBM 3270 B-1, C-1
- N
- numeric keypad lock
  - ANSI-PC D-2
- P
- parity 6-9
- paste
  - aborting large paste 5-5
- paste options 6-22
  - filename 6-23
- Preferred (character) Set
  - DEC VT320 A-2
- printers
  - attached 6-26
    - TI 931 6-26
  - pipe to command 6-26
  - save to file 6-26
  - spooling 6-26
- Prompt when Closing 5-4
- Q
- queue filename for transmit 6-23
- R
- rb command 6-31, 6-32
- Reconnect session command 5-5
- reference manuals
  - ANSI-PC D-3
  - DEC VT series A-6
  - DG D200 series E-3
  - IBM 3270 B-10

- IBM 5250 C-3
- TI 931 F-3
- WYSE WY-60 G-9

Reset Terminal command 5-5

resize

- font 6-15

Return key

- ANSI-PC D-1
- DEC VT102 6-6
- DEC VT320 A-4
- DG D215 E-1
- TI 931 F-2
- WYSE WY-60 G-1

rz command 6-30

S

sb command 6-31, 6-32

session macros 6-47

spool macros 6-47

- General Keys 6-47

status line

- DEC VT320 A-1
- TI 931 F-1
- WYSE WY-60 G-3

sz command 6-29, 6-30

T

tabs

- DEC VT320 A-2
- WYSE WY-60 G-5

telnet client

- IBM 3270 6-11

terminal macros 6-47

terminal reset 5-5

TI 931 F-1

- attached printer 6-26

title

- main window 6-19

tset 6-5

V

VT102, see DEC VT102

VT320, see DEC VT320

W

warning bell 6-25

WYSE WY-60 G-1

X

Xmodem file transfer protocol 6-32

XON/XOFF 6-9

XON/XOFF flow control 6-9



Y

Ymodem file transfer protocol 6-31

Z

Zmodem file transfer protocol 6-29





**Yrrid Incorporated • 507 Monroe Street • Chapel Hill • North Carolina 27516**

**Tel: (919) 968-7858 • Fax: (919) 968-7856 • EMail: [info@yrrid.com](mailto:info@yrrid.com)**