XyWrite 4.0 Customization Guide

Published by The Technology Group in 1993

Customization Guide

C:\XY4

XyWrite™ 4.0

Customization Guide

XyWrite 4.0

First Edition (Revised), March 1993
Document Control No. 003506-02

The Technology Group
36 South Charles Street
Baltimore, MD 21201
410-576-2040

This document was produced using XyWrite 4.0.

Table of Contents

Preface

Custom Commands

Defaults

Keyboard File

Memory

Menu and Help Files

Printer Files

Sort File

Startup File

User Programming

Variables

Appendixes

NOTES

Preface

The *Customization Guide* describes XyWrite functions that are beyond the scope of the *Command Reference Guide*. It is intended for technical coordinators, office information technology managers, LAN administrators, and others with technical expertise.

The *Customization Guide* gives information and instructions on the following topics.

- Custom Commands. This section explains the procedure for developing a customized set of XyWrite commands and function calls.
- Defaults. This section explains SETTINGS.DFL, XyWrite's default file. It also documents all the defaults available, as well as color settings and customization tables.
- Keyboard File. This section describes the procedure for modifying the XyWrite keyboard file and lists the available function calls.
- Memory. This section describes the commands you use to load custom information into memory.
- Menu and Help Files. This section gives you the information you need to customize the standard XyWrite files (XY4.MNU and XY4.HLP) or create your own.
- Printer Files. This section describes the basic construction of a printer file. You can use this information in conjunction with the documentation provided by the printer manufacturer to customize or create a printer file.
- Sort Table. This section gives you the information you need to create a sort table so you can modify the order in which XyWrite sorts data.
- Startup File. This section describes the format and function of the XyWrite initialization file and explains how to modify it.
- User Programming. This section explains the procedure for using XyWrite Programming Language (XPL), and describes the commands and operators available to you.
- Variable Settings. This section describes the VA (Value of Variable) command, which obtains the current value of a XyWrite setting. It also includes a list of system settings for which you can obtain values.

There are also five appendixes:

- Appendix A. Advanced Keyboard Functions. This appendix contains a description of some keyboard shortcuts that are designed for the advanced user.

- Appendix B. Terminate-and-Stay-Resident Programs. This appendix provides some tips on using TSR programs with XyWrite.
- Appendix C. Foreign Language Support. This appendix describes two features of XyWrite that are useful for foreign language applications: the accent table and uppercase table.
- Appendix D. Reserved XyWrite Elements. This appendix lists ID numbers, keywords, and macros reserved for use by XyWrite.
- Appendix E. XyWrite Character Set. This appendix lists the entire character set.

For more information about XyWrite, refer to the following publications:

Command Reference Guide
Installation and Learning Guide
LAN Administrator's Guide

Custom Commands

PURPOSE

XyWrite's command override feature lets you create your own set of commands. You can redefine the functions associated with existing immediate commands (such as CALL and DIR) and function calls (such as CL and YD), or you can create new command names.

Custom command sets are stored in a separate customization file and loaded into memory with the LOAD command. Once the command set has been defined and loaded, you activate it with the OO (Override On) and OX (Override Off) commands.

ACTION

Creating a Custom Command Set

To create a custom command set:

1. Create the customization file. For example:

   Type: [F5]new command.set[↵]

2. On the first line, type the label that identifies this file as a command set. Be sure to type it in uppercase.

   Type: ;U2;

3. Create the first custom command. The format of each immediate command in the set is:

   {{5,^^*name*}} (or {{5,^*name*}} — see Note #1)
   ☻*code*☻

   where *name* is the name of the immediate command or function call you are redefining or the name of the immediate command you are creating and *code* is the XPL (XyWrite Programming Language) instruction you want the command to execute.

   Let's take a look at a simple example (a more complex example is presented at the end of this section). We'll create a new command named GET that calls a frequently referenced appointment file named DATES.

   a. Type the command label followed by a carriage return.

      Type: {{5,^^get}}←

   b. Type ASCII character 2.

      Type: [Alt][Ctrl]2 (on the numeric keypad)

c. Write the program for command you want to execute. (Refer to "User Programming" for instructions on writing XPL programs.) We recommend that you start *all* programs with instructions to disable command override. For our example, the program would look like this:

```
BX oxQ2 BX call c:\xy4\docs\datesQ2 BX ooQ2 « ex»
```

Note that the last two commands turn on command override and exit the program.

d. Type an ASCII character 2 to mark the end of the code for this command.

4. Repeat step 3 for every command you want to redefine or create.

5. Store the file.

ACTION

Using the Custom Command Set

To use the custom command set:

1. Load it into memory. For example:

   Type: [F5]load command.set[↵]

2. Turn on command override.

   Type: [F5]oo[↵]

3. Issue any of the commands in the custom set. If you redefined a standard XyWrite immediate command or function call, the new definition will be applied.

4. To restore the standard XyWrite functions and commands, turn off command override.

   Type: [F5]ox[↵]

NOTE #1 **Command Labels.** If you are redefining or creating a new immediate command, type two carets (^^) before *name*. If you are redefining a function call, type one caret before *name*.

NOTE #2 **Function Calls.** You should use caution when redefining function calls or you may not get the results you expect. We recommend that you avoid redefining the function calls BC, BX, Q2, and XC.

NOTE #3 **VA Command.** You can check the status of command override with VASOO. A value of 0 means it is off; a value of 1 means it is on.

NOTE #4

Command Line. If you need the contents of the command line, your XPL program should include a command to save it before you execute any other BX or BC functions. For example, to save the contents of the command line to macro 01, insert the following instructions at the beginning of the XPL code that defines your custom command:

```
«sx01,«va$cl»»
```

EXAMPLE

```
{{5,GetArg}}
▣«sv01,^^»«XS40,01,02,03,04»«SX10,«VA$CL»»«XS10,04,05,06,07»
«SX40,«IS07»»▣
←
{{5,^^open,^^o}}
▣«SX40,«VA$FR»»JM (2.GetArg) BX oxQ2 BX call«PV40»Q2 BX ooQ2 ▣
←
{{5,^^write,^^w}}
▣«SX40,«VA$FR»»JM (2.GetArg) BX oxQ2 BX save«PV40»Q2 BX ooQ2 ▣
```

This example defines two new commands: Open (abbreviated "O") and Write (abbreviated "W"). In both commands, VA$FR captures the name of the command override routine that was accessed (^^open, ^^o, ^^write, or ^^w) and stores it to macro 40.

The program then jumps to a routine called GetArg, which captures any argument that was typed after the custom command, including any switches, such as /NV. Notice that the JM function call uses "2." at the beginning of the jump. The "2." indicates that "GetArg" is in the U2 customization file.

GetArg parses the ^^ from the command, storing the result in macro 04. It then uses VA$CL to capture the text from the entire command line in macro 10. Next, macro 04 is parsed out of macro 10, and the results are saved in macro 07. For example, if you type the command:

```
w/nv  newfile.doc
```

the string "/nv newfile.doc" is saved in macro 07. Finally, macro 07 is saved back to macro 40.

In the original routine, macro 40 is put out as part of the XyWrite command. In the example, W/NV NEWFILE.DOC is executed as SAVE/NV NEWFILE.DOC.

NOTES

Defaults

INTRO

When XyWrite first starts up, it provides you with its own set of defaults. You can use the DEFAULT and DF commands to change the values of these settings, either temporarily or permanently. Permanent changes are stored in a customization file called SETTINGS.DFL, which is loaded every time you start XyWrite.

This section also describes the MD command, which lets you customize the display colors of text, and the SE, PP and CS tables, which specify the separators, port/printer file combinations and counter strings XyWrite uses.

CONTENTS

Default File

PURPOSE

The default file, SETTINGS.DFL, contains the commands and tables that override XyWrite's built-in values for anything from page layout settings to screen colors. You can add to SETTINGS.DFL either manually or via the menu.

The information in SETTINGS.DFL can be grouped into three categories:

- Settings that can be defined with the DF (Default) command
- Display mode colors
- Customization tables, e.g., PP (Pick Printer) table

As in other customization files, the first line of SETTINGS.DFL contains a label. The label for a default file is the same as that for a printer file:

;PR;

This label must be typed in uppercase, and must appear at the beginning of the first line of the file for XyWrite to recognize it as a default file.

ACTION

Modifying SETTINGS.DFL

You edit SETTINGS.DFL just as you would any text file. Let's change the default settings for WD (Widows) and OP (Orphans) to 3 (the initial value for each is 2).

1. Call SETTINGS.DFL to the screen.
2. Search for the existing WD setting (see Note #2).
3. Change the number after the equals sign to a 3.
4. Repeat steps 2 and 3 for the OP setting.
5. Store the file.

Result: The next time you load the file, the value of WD and OP will be 3. (See the procedure "Loading SETTINGS.DFL.")

ACTION

Loading SETTINGS.DFL

SETTINGS.DFL is automatically loaded during startup, but if you make a manual change to it, you must reload it in order for the change to take effect. To load SETTINGS.DFL into memory:

Type: F5load settings.dfl↵

Result: Any new values are loaded into memory. Existing default values are not changed unless they are explicitly overwritten.

NOTE #1 **Determining the Current Value.** If you want to know what the current value of a default setting is, use the VA (Value) command, which is described later in this guide.

NOTE #2 **Adding a Value to SETTINGS.DFL.** If the default setting you want to modify is not already specified in SETTINGS.DFL, we recommend adding it to the bottom of the file. The format is:

```
df mn=#←
```

Be sure to include the unit of measure, if applicable. For more information, see "Default Command" later in this section.

NOTE #3 **Duplicate Settings.** If you inadvertently define the same setting twice, the second setting overrides the first one.

NOTE #4 **Priority of Default Settings.** A command embedded in text takes priority over the same command entered with a default command anywhere else — on the command line, in the SETTINGS.DFL file, or in the printer file. For example, RM 8 embedded into the document takes precedence over a default setting of DF RM=7IN entered in SETTINGS.DFL or a printer file and over DEFAULT RM=7 entered on the command line.

NOTE #5 **Unit of Measure.** When you issue a command from the command line, XyWrite automatically appends the default unit of measure if you don't supply one. When you enter a DF command in a default file, you must include the appropriate unit of measure. Otherwise, XyWrite will assume horizontal values to be in tenths of an inch and vertical values to be in sixths of an inch.

Default Command — DF, DEFAULT

FORMAT

C-XY DEFAULT/nv *nn*=# (on command line or in STARTUP.INT file)
DF *nn*=# (in SETTINGS.DFL or printer file)

/nv is an optional switch that preserves upper- and lowercase letters and spaces in the command.
nn is the two-letter name of the command.
= (equal sign) separates the name from the value.
# is the value of the command.

MENU

Advanced Preferences Defaults...

PURPOSE

The DEFAULT and DF commands change the value of XyWrite default settings. Both commands can be used to change any default setting, and both affect *all* files — not only newly created files.

There are several differences between the two commands:

- The DEFAULT command is issued from the command line. The DF command is stored in SETTINGS.DFL, a printer file, or other default file. The stored settings are executed when you load the file that contains them.
- The DEFAULT command adds the default unit of measure to format settings. The DF command requires you to include the unit of measure. For example, the following commands are equivalent:

  ```
  [F5]default rm=7↵
  df rm=7IN↵
  ```

- The DEFAULT command ignores upper- and lowercase and spaces within a default setting unless you include the /NV switch. The DF command honors both case and spaces without the switch. The following commands are equivalent:

  ```
  [F5]default/nv mg=Happy Birthday!↵
  df mg=Happy Birthday!↵
  ```

ACTION

Entering Defaults into SETTINGS.DFL

The procedure for entering defaults into SETTINGS.DFL is described in the previous section "Default File."

ACTION

Entering Defaults from the Command Line
When you want to change a default temporarily (only for the current editing session), the appropriate method is to change it from the command line. For example, to set the right margin at 7 inches:

Type: Fs default rm=7↵

Result: These changes take effect immediately — whenever you create a new document or call an existing one, until you quit XyWrite or change the defaults.

ACTION

Entering Defaults into STARTUP.INT
In general, we recommend that you specify permanent default settings in SETTINGS.DFL, but there are exceptions. For example, the DR (Drive) default setting is specified on the first line of STARTUP.INT because it establishes information that XyWrite may need to complete loading.

STARTUP.INT is a program file. To enter a default command into it, you must start the line with the BC (Blank Command Line) function call. The procedure is:

1. Call STARTUP.INT.
2. Move the cursor to the point where you want to enter the new default.
3. Enter the BC function call.

   Press: Ctrl Alt Shift B
4. Enter the DEFAULT command. For example:

   Type: default nd=e,f,z
5. Store STARTUP.INT

Result: The next time you start XyWrite or run STARTUP.INT, the default command that establishes network drives will be automatically executed. (Refer to "Startup File" for more information about modifying STARTUP.INT.)

NOTE #1

Defaults in the Printer File. The procedure for putting a default setting in a printer file is identical to the procedure for modifying SETTINGS.DFL.

NOTE #2

Priority of Default Settings. Commands embedded in text take priority over all other settings — those entered in the SETTINGS.DFL file, the STARTUP.INT file, the command line, or in the printer file. In other words, RM 8 embedded into the document takes precedence over a default setting of RM=7IN in the SETTINGS.DFL file, the STARTUP.INT file, the printer file, or on the command line.

Default Settings

PURPOSE

Default settings let you change the value of XyWrite settings. Default settings can be divided into four categories:

- **Format Settings** — Set the initial format conditions at the start of every file. For example, you might use it to set the right margin at 7 inches and the type size at 11 points.
- **System Settings** — Set controls specific to your system, such as adjusting the level of audible signals and establishing error prompts.
- **Display Settings** — Set controls specific to your monitor. Also sets your preferences for colors and display styles. For a list of available colors, refer to "Color Settings" later in this section.
- **Printer Settings** — Set controls specific to your printer. (Some printer settings are defined in printer files, not the default file.)

The following table lists the XyWrite default settings, along with the initial default value. Note that the initial value of some of the system, printer, and display defaults varies, depending on the setup you have. For example, the SI (Screen Resolution) default is different for EGA and VGA. XyWrite establishes the values that are appropriate for your setup based on the choices you make during installation.

Default	Description	Initial Default Value
Format Settings		
df al	Automatic Leading	1
df bl	Blank Lines	0
df bf	Bottom Footnote	1
df bt	Bottom Margin	1in,1in,1.25in,.75in
df cf	Change Footnote Separator	0
df dy	Color	0,0
df dz	Date Format	Mmmm d, yyyy
df ee	Element End	0
df ef	Special Effect	0
df et	Element Top	0
df fc	Flush Center	FL
df fd	Form Depth	11in
df fl	Flush Left	FL
df fr	Flush Right	FL
df ft	Footnote Transition	0
df gu	Gutter	.2in,0
df hy	Hyphenation	0
df ip	Indent Paragraph	0,0,0
df ju	Justification	NJ
df ll	Line Leading	0,0
df lm	Left Margin	1in

Default	Description	Initial Default Value
Format Settings (cont'd)		
df lq	Letter Quality	0
df ls	Line Space	1li
df md	Display Mode	NM
df nj	No Justification	NJ
df of	Offset	1in,1in
df op	Orphan	2
df or	Orientation	0
df pw	Page Width	8.5in
df rm	Right Margin	11in
df rt	Relative Tabs	1
df sc	Superscript Numbers	NM
df sz	Size	12 point
df tp	Top Margin	1in,1in
df ts	Tab Set	.5in,1.5in,2.5in..
df uf	Use Typeface	Standard
df ul	Underline Style	1
df wd	Widow	2
System Settings		
df 1a	End of File	0
df ah	Allow Hyphenation	0
df aop	Autosave Path	C:\XY4\DOCS
df aot	Autosave Time	0,0
df az	Numbering Style	0
df bk	Backup	0
df cb	Correction Beep	512,4096
df ck	Spelling Checker	0
df cv	Change Verification	0
df d1	Delete Stack	30,3
df db	Debugging Mode	0
df dh	Discretionary Hyphen	-
df dl	Directory Settings	1,6,0
df dp	Decimal Point	.
df dr	Default Drive	C:\XY4\
df eb	Error Beep	1792,36864
df eh	Error Help	0
df ep	Error Prompt	1,1,1,1,0,0,1
df es	Enable Scoping	0
df eu	European Punctuation	[illegible]
df fu	Footnote Unit	3,5

Default	Description	Initial Default Value
System Settings (cont'd)		
df fx	Field Separator	○
df fz	Format Date	mm-dd-yy
df gp	Graphics Path	C:\XY4\PICTURES\
df hi	Header Item	%f au %d cm
df ht	Header Title	Filename Author Saved Comment
df hv	Hyphenation Values	6,3,3
df io	Information On/Off	1
df iu	Information Menu	1
df jb	Job Begin	
df kc	Key Click	0,0
df ks	Keyboard Speed	0,1
df lh	Low-High	50,100,22
df lz	Redlining Date	MM/DD
df ma	Find Match	40
df mf	Mode for Forms	NM
df mg	Message	—
df mt	Military Time	0
df nc	Normal Carriage Return	1
df nd	Network Drives	—
df nl	Network Login Path	C:\XY4\
df nw	New Window	3
df o1	Options	0
df ob	Overstrike Beep	0,0
df oe	Open Editor	0
df ol	Outline Fonts Path	C:\BTFONTS\
df pd	Pad Spaces	0
df r2	Mouse Double Click	8
df rb	Reverse Button	0
df rl	Repetition Interval	5
df rn	Round Off Numbers	0
df rs	Record Separator	←
df rx	Ratio for X Direction	15
df ry	Ratio for Y Direction	15
df rz	Record Size	3500
df sg	Save/Get	—
df sk	Sort Key	1,80
df sm	Show Menus	1
df so	Sort Setting	F1
df uh	Unit, Horizontal	in
df un	Untitled File	0
df ur	Use Pointer	1
df uv	Unit, Vertical	in

Default	Description	Initial Default Value
System Settings (cont'd)		
df vf	Variable Forms	0
df wa	Wait Time	18
df wo	Word Overstrike	0
df ww	Filters Path	C:\XY4\FILTERS
df xd	Read-Only Directory	1
df xf	Extract Fields	
df xl	Selective Directory	
df xt	Extract Records	
df xt	Expand Triangle	1
df xy	Screen Fonts Path	C:\BTFONTS\
df zs	Point Sizes	8,9,10,12,14,16,18,24,36,48,72,96
Display Settings		
df bw	Black/White	1
df bx	Window Border Colors	15,7,7...7,15,15,7,240
df cr	Cursor Type	0,199,102,4
df dd	Define Display	119
df de	Soft carriage return	▼, ◄
df dt	Display Type	1
df eg	EGA Control	0
df ga	Graphics Adapter	MD
df it	Insert Cursor Type	519,1543,8193,264,1543,8193,0,1
df l0	Action Bar Color	7,12,16,7
df l1	Command Line Color	15,112,112,7
df l2	Prompt Line Color	112,7,15,7
df l3	Ruler Line Color	112,15.15,7
df l4	Pull-Down Menus Color	15,112,112,7,112
df lc	Line End Character	←
df mr	Metric Ruler	10
df mw	Maximize Windows	1
df od	Offset Display	0
df p.	Truncated Path Name	14
df pk	Page Break Color	23
df px	Page Break Character	196
df rd	Redline Data	1,14,9,0,249
df rl	Ruler Markers	▶◀▼⌂└┐░├┤•·
df sl	Screen Resolution	96,72
df sl	Screen Length	25
df st	Show Tabs	1

Default Settings (cont'd)

Default	Description	Initial Default Value
Display Settings (cont'd)		
df sw	Screen Width	80
df um	Unhide Modes	0
df wb	Window Border Chars.	┌ ┐ ─ └ ┘ │ ↑ ↓ → ← ░ ▓ ░ ▓
df wf	Wrap to Fit	1
df xm	Transpose Messages	•PL •FI •TI
df zm	Zoom	75
Printer Settings		
df bs	Backspace	1
df ej	Eject Last Page	1
df ff	Form Feed	1
df im	Image Mode Printing	0
df ne	No Errors from Printer	0
df sq	Sequential Pages	0
df tb	Tab Character	0
df tf	Top Margin	0
df ws	Whole Space Justification	0

DETAIL

This section contains a description of most of the default settings listed above. These descriptions are not categorized, but instead are listed in alphabetical order. The Format settings are described in the *Command Reference Guide*, and the information is not repeated here.

NOTE #1

DEFAULT vs DF. The following descriptions use the DF command. However, all the defaults can also be specified with the DEFAULT command.

NOTE #2

Color Settings. Many of the default settings specify display colors. For a list of choices, refer to the color table in "Display Mode Colors" later in this section. You can also display the available colors by selecting "Color Choices" from the Help index.

DF 1A

Ignore End of File Marker — Enables XyWrite to read past the end of file marker (1A). (The initial default is 0.)

df 1A=0 Causes XyWrite to honor the end of file marker.

df 1A=1 Causes XyWrite to ignore the end of file marker. You would use this if you are including, merging or appending a file from another source that has a 1A character in it. (Do not use this setting if you have Document Info turned on.)

DF AH

Allow Hyphenation — Enables XyWrite to hyphenate a word that contains a hard hyphen (e.g., self-evident). (The initial default is 0.)

df ah=0 Hyphenates words that contain hard hyphens.

df ah=1 Does not hyphenate words that contain hard hyphens.

DF AOP

Autosave Path — Specifies the path for the Autosave files. (The initial default is the directory you establish as your working XyWrite directory.)

df aop=*d:\path* Format of AOP setting, where *d:* is the drive letter and *path* is the directory where you want XyWrite to put Autosave files. If you just specify a drive, XyWrite assumes the root directory. Directory must exist!

df aop=c:\xy4 Example of AOP setting.

DF AOT

Autosave Timer — Specifies the amount of time you want to elapse between autosaves. (The initial default is 0,0.) = no autosave

df aot=*m,n* Format of the AOT setting, where *m* is the minimum length of time (0 to 55 minutes) and *n* (optional) is the maximum length of time (0 to 60 minutes) XyWrite should wait before saving the file. If you do not specify a maximum time, the default is the minimum time plus 5 minutes. XyWrite starts counting from the time you make an edit. After the minimum time has passed, XyWrite checks to see if you are typing. If you aren't, XyWrite saves the current file in AUTOSAV*n*.TMP (where *n* is the current window number). If you are typing, XyWrite waits until you pause or until the maximum amount of time has elapsed before saving.

df aot=10,12 Example of AOT setting.

DF AZ

Numbering Style — Changes the sequence style of counters when you are using upper- and lowercase letters and have more than 26 items for that counter. (The initial default is 0.)

df az=0 Restarts the letter sequence at the 27th entry as AA, BB, CC, etc.
df az=1 Restarts the letter sequence at the 27th entry as AA, AB, AC, etc.

DF BK

Backup of Files — Turns the backup of files on or off. (The initial default is 0.)

df bk=0 *Does not* keep backup copies. (This saves room on your disk.)
df bk=1 *Keeps* backup copies. With this setting, XyWrite makes a backup at every SAVE. Whenever the SAVE command is executed, XyWrite renames the previously saved version to *filename*.BAK, overwriting any previous backup, then saves the displayed version. Note that backup does not happen if you execute SAVE with an argument — such as to a drive (SAVE A:) or filename (SAVE TEST).

DF BS

Backspace Control — Enables your printer to print a backspace, whether or not your printer can recognize a backspace character. For more information on this setting, refer to the section on "Printer Files" elsewhere in this manual.

DF BW

Black and White — Specifies a fast screen or a clean screen for systems with CGAs (Color Graphic Adapters). (The initial default is 1.)

df bw=0 Creates a clean screen, with no flicker, but is slower. Recommended for systems with CGAs.
df bw=1 Causes XyWrite to update the display faster. This setting provides improved speed when scrolling text, switching between windows, and running programs. The cost is a noticeable flicker each time the screen is updated.

DF BX

Window Border Colors — Defines the colors of the window borders, markers, and graphic view. (The initial default for a monochrome monitor is 15,7,7,7,7,7,7,7,7,7,15,15,7,240.)

df bx=*a,w1,w2,...,w9,l,m,t,g* Format for the BX setting, where *a* sets the color of the active window; *w1,w2,...,w9* set the color of windows 1 through 9; *l* sets the color of labels; *m* sets the color of command markers; *t* sets the color of tab markers; and *g* sets the color of graphic view.
df bx=31,23,23,23,23,23,23,23,23,23,32,30,23,240 Example of BX setting.

DF CB

Correction Beep — The CB (Correction Beep) setting allows you to define the tone and duration of the beep you hear whenever the Spelling Checker automatically corrects a misspelling for you. (The initial setting is 512,4096.)

df cb=*t,d* Format for the CB setting, where *t* defines the tone of the beep and *d* defines the duration of the beep. Both are numbers between 0 and 65534.

df cb=0,0 Example of the CB setting. This command turns the correction beep off.

DF CF

Change Footnote Separator — Defines which footnote separator is printed on a page. XyWrite supports three independent sets of footnotes, and you can define a separate style and format for each set. Whether you define one, two, or three styles of footnote separators, the first thing XyWrite prints at the bottom of any page that contains footnotes is the separator associated with footnote set 1. This is true even if there are no footnotes from set 1 on the page, and results in a uniform appearance of the pages in your document. To print only the separators associated with footnote sets that actually appear on the page, change the CF default setting. (The initial default is 0.)

df cf=0 Causes footnote separator 1 to print at the bottom of any page that contains footnotes.

df cf=1 Causes only the footnote separators that are associated with footnotes that appear on the page to print.

DF CK

Spelling Checker — Turns the checking of words that contain numbers on and off. (The initial default is 0.)

df ck=0 Causes the Spelling Checker to check words that contain numbers and letters (e.g., 12th).

df ck=1 Causes the Spelling Checker to ignore all words that contain a number.

DF CR

Cursor Type — Changes the text cursor to be non-blinking and defines the color of the text cursor or mouse pointer. (The initial default is 0,199,102,4.)

df cr=*a,b,c,d* Format of the CR setting, where *a* is the number of the display mode for the cursor normally, when you are editing text; *b* is the number of the display mode for the cursor when a dialog box is displayed (XyWrite uses this value only if the first value is 0); *c* is the number of the display mode for the mouse pointer in the text; and *d* is a number (0-5) that specifies the width of the cursor in graphic view (use 0 if you want the cursor to be the width of the character under it; use 1 to make the cursor 1 pixel wide; 2 to make the cursor 2 pixels wide, etc.).

df cr=0,112,88,0 Example of the CR setting.

DF CV

Change Verification Prompt — Enables the prompt "Cannot recover changes--proceed anyway?" whenever you execute the CH (Change) command; this message reduces the risk of inadvertently making changes. (The initial default is 0.)

df cv=0 Turns off the prompt.

df cv=1 Enables the prompt.

Regardless of your choice of setting, the changes are not permanent until you save or store the file.

DF D1

Delete Stack - Defines the number of entries XyWrite keeps in the delete stack and the minimum number of characters it recognizes as a delete unit. The initial default is 30,3.

df d1=*e,d* Format of the D1 setting, where *e* is the number of entries in the delete stack and *d* is the minimum number of characters recognized as a delete unit.

df d1=24,7 Example of the D1 setting.

DF DB

Debug a Program — Enables you to test parts of your XPL (XyWrite Programming Language) program so you can verify that it is doing what you want. Also enables you to create programs that are formatted with carriage returns and tab settings.

df db=*x,y* Format of the DB setting, where *x* is one of the following values:

0 Turn off debugging.

1 Stop on IF statements.

2 Stop on labels.

4 Stop on JM (Jump to Menu) commands.

8 Stop on carriage returns.

16 Stop on commands that don't execute properly.

y (optional) is one of the following:

1 Ignore carriage returns when running the program.

2 Ignore tabs when running the program.

3 Ignore both.

You can add two or more values to stop the program at more than one type of entry. For example, DF DB=3 stops the program after IF statements *and* labels.

DF DD

Define Display — Changes the appearance of defined blocks. This setting is useful for low-contrast screens such as those found in laptops. XyWrite takes the number that you specify and compares it, in binary form, to the number of the text that is defined. The actual display is based on the binary difference between those two numbers. The display will differ from mode to mode; the appearance of the command triangles will also be affected. We suggest that you choose a number for DD based on the appearance of normal mode text and the triangles. (The initial default is 119.)

df dd=*n* Format of the DD setting, where *n* is the number on which XyWrite bases its calculations to determine how to display defined text.

df dd=40 Example of the DD setting.

DF DE

Soft Carriage Return — Defines the character you want to use as the soft carriage return. (The default characters are ♥,◄.)

df de=*m,n* Format of DE setting, where *m* is the character you assigned to a key and *n* is the character you want to display as the soft carriage return.

df de=<,< Example of DE setting.

Note: If you don't want a character to display, define a space for *n* by pressing Ctrl Alt 32.

DF DH

Discretionary Hyphen — Defines the character you want displayed as a soft hyphen. (The default character is –.)

df dh=*d* Format of DH setting, where *d* is the character you want displayed as a soft hyphen.

df dh=^ Example of the DH setting.

DF DI

Directory Settings — Modifies the way a long directory (DIRL) is displayed. (The initial default is 1,6,0.)

df di=*k,l,p* Format of the DI setting, where *k* is the number the file size is to be divided by; *l* is the approximate number of lines of text displayed; and *p* packs the text (1 removes carriage returns; 0 keeps them.)

df di=1024,4,1 Example of the DI setting.

DF DP

Decimal Point — Defines the character used for number alignment in decimal tabs. The default value is a period (.).

df dp=*d* Format of DP setting, where *d* is the character that will be used for number alignment.

df dp=) Example of the DP setting.

Default Settings (cont'd)

DF DR — **Default Drive for Temporary Files** — Specifies the primary and secondary default drives and paths where you want to save temporary files. The secondary drive is used only when the primary drive is full. If you specify a RAM drive as the primary drive and the hard disk as a secondary drive, you can take advantage of the added speed of a RAM drive without being handicapped by its smaller size.

df dr=*d1:\path,d2:\path* Format of the DR setting, where *d1:\path* is the primary drive where TMP fies are saved and *d2:\path* is the secondary drive where TMP files are saved.

df dr=e:\,c:\sw Example of the DR setting.

DF DT — **Display Type** — Specifies how documents are to be displayed when first called to the screen. Some of the settings hide the markers—the line-end symbol (←) and embedded command markers. DF DT affects all windows. (The initial default is 1.)

df dt=0 Causes the file to display in expanded view.

df dt=1 Causes the file to display in draft view with page breaks off, markers visible.

df dt=2 Causes the file to display in draft view with page breaks on, markers visible.

df dt=4 Causes the file to display in graphic view with markers visible.

df dt=9 Causes the file to display in draft view with page breaks off, markers hidden.

df dt=10 Causes the file to display in draft view with page breaks on, markers hidden.

df dt=12 Causes the file to display in graphic view with markers hidden.

DF DZ — **Date Format** — Modifies the way the date is displayed by the DA and TODAY commands. (The initial default is Mmmm d, yyyy for Month, Day, Year.)

df dz=*date* Format of the DZ setting, where *date* is any combination of the following:

MMMM	Complete name of month. To make the letters of the month lowercase, use m instead of M.
MMM	Three-letter abbreviation
MM	Two-digit month (01-12)
M	Months 1-12
ddd	Days with leading space for days 1-9
dd	Days with leading zero for days 1-9
d	Days with no leading zero or space
yyyy	Four-digit year
yy	Two-digit year

DF EB

Error Beep — Defines the tone and duration of the beep you hear whenever XyWrite detects an error. The speed of the hardware will affect the duration of the beep, so if you have a faster machine you should increase the duration setting accordingly. We recommend multiplying the duration by 4 on 386 machines. (The initial setting is 1792,36864.)

df eb=*t,d* Format of the EB setting where *t* defines the tone of the beep and *d* defines the duration of the beep. Both are numbers between 0 and 65534.

df eb=768,6144 Example of EB setting.

DF EG

EGA Support — Specifies 25 screen lines or 43 screen lines if you have an IBM Enhanced Graphics Adapter (EGA). (The initial default is 0.)

df eg=0 Produces 25 screen lines.
df eg=1 Produces 43 screen lines on a black & white monitor.
df eg=2 Produces 43 screen lines on a color monitor.

DF EH

Display Error Help Frame — Enables an error screen to appear (from the help file) whenever an error occurs. (The initial default is 0.)

df eh=1 The error/help frames are displayed whenever an error occurs. This feature is useful when you are learning, because you can display information about how to continue. However, it can slow performance considerably.

df eh=0 The error/help frames are disabled.

DF EJ

Eject Last Page — Enables XyWrite to eject the very last page of a document. For more information on this setting, refer to the section on "Printer Files" elsewhere in this manual.

DF EP

Error Prompt — Enables a variety of prompts designed to give you a chance to reconsider actions that may have undesirable consequences. Each prompt is turned on with 1 and off with 0. (The initial default is 1,1,1,1,0,0,0.)

df ep=*a,b,c,d,e,f,g* Format of the EP setting, where:

a controls the message "Do you want to erase? Y/N" when you issue a DELETE command.

b controls the message "Print defined block? (Y/N)" when you issue one of the PRINT commands while you have defined text on the current screen.

c controls the message "Print directory? (Y/N)" when you issue one of the PRINT commands when you have a directory displayed and the cursor is on the command line rather than on a file name.

d controls the message "File modified, abort anyway? (Y/N)" when you issue the ABORT command after you have modified the displayed file.

e controls the message "OK to save? (Y/N)" when you use Ctrl S to save from the keyboard or when you use the function call SA.

f controls the message "Delete command marker? (Y/N)" when you attempt to delete a formatting marker.

g controls the message "Screen/printer font mismatch" when you are in graphic view and the screen representation of the fonts is different from what will print.

df ep=0 This setting turns off all prompts.

DF ES

Enable Scoping — Specifies the boundaries, or scope, of formatting commands. (The initial default is 0.)

df es=0 Applies formatting commands to text from the current cursor position to the end of the document or until it is overridden.

df es=1 Paragraph Forward. Formatting commands take effect at the beginning of the current paragraph, and override any other occurrences of the command within the paragraph. The new command remains in effect until the end of the document or until it is overridden by another occurrence of the same command in a subsequent paragraph.

df es=2 Previous Command Forward. When you issue a formatting command, XyWrite searches back through the file for another occurrence of the command. If it finds one, it changes the command to the new value; if it does not find a previous occurrence of the command, it inserts the new command at the current cursor location.

DF EU

European Punctuation — Defines the way XyWrite punctuates numbers for doing calculations. It also controls the argument separators used to issue XyWrite commands and the time separator. This setting does not affect decimal tabs. (The initial default is .,:,;)

df eu=*dslag* Format of the EU setting, where *d* (decimal point) defines the decimal point character; *s* (separator) defines the character inserted between 3-digit groups in numbers of one thousand or more (for example, the commas in 1,000,000); *t* (time) defines the character used to separate hours from minutes in all XyWrite-produced time displays; *a* (argument) defines the character you want to use to separate command arguments; and *g* (group) defines the character you want to use to separate groups of command arguments.

df eu=,.:¦; Example of the EU setting.

Notes:

- The EU default setting does not use commas between arguments.

- The argument and argument group separators are designed for international versions of XyWrite. They are not implemented in the U.S. version.
- You must specify unique characters for the decimal point, argument, and group separators.

DF FF

Form Feed Character — Enables XyWrite to insert a form feed character at the end of each page. For more information on this setting, refer to the section on "Printer Files" elsewhere in this manual.

DF FU

Footnote Unit — Defines the width of undisplayed footnote symbols or references. (The initial default is 3,5.)

df fu=*f,r* Format of the FU setting, where *f* is the width being allotted for footnote symbols and *r* is the width being allotted for cross-references.

df fu=3,7 Example of the FU setting.

DF FX

Field Separator — Specifies the character or string you want to use to separate fields in a data file. (The initial default is a tab character.)

df fx=*f* Format of the FX setting, where *f* is the character or string you want to use as a field separator.

df fx=^ Example of the FX setting.

DF FZ

File Date — Modifies the way the date is displayed in a directory. (The initial default is m-d-yy, for month, day, year.) The format choices are the same as those available for the DA command. For more information on the different settings, refer to the DZ setting above.

DF GA

Graphics Adapter — Defines the type of graphics adapter you are using.

df ga=md Indicates a monochrome adapter is in use; there is no graphics support.

df ga=cg Indicates a CGA adapter is in use. A CGA adapter normally displays monochrome graphics with a screen resolution of 640x200.

df ga=hg Indicates a Hercules Graphics adapter is in use. A Hercules Graphics adapter displays monochrome graphics with a screen resolution of 720x348.

df ga=eg Indicates an EGA adapter is in use. EGA supports color or monochrome graphics with a screen resolution of 640x350.

df ga=vg Indicates a VGA adapter is in use. VGA supports color or monochrome graphics with a screen resolution of 640x480.

DF GP

Graphics Path — Defines the location of the graphic files. XyWrite uses this information to create the list box in the Import Graphic dialog box. (The initial default is C:\XY4\PICTURES.)

df gp=*d:\path* Format of the GP setting, where *d:* is the drive that contains graphic files and *\path* is the subdirectory.

df gp=d:\xy4\picture Example of the GP setting.

DF HI

Header Items — Specifies the categories of information you want displayed when you issue the DIR/SU command. The default is %f au %d cm (filename, author, date last saved, comments).

df hi=*f1 f2* . . . Format of the HI setting, where *f1, f2* . . . are the information field mnemonics, spaced as you want them to appear in the directory. Available fields are:

%f	Filename (should always be the first field)
%s	File size
%d	Date the file was last saved
%t	Time the file was last saved
%p	Path of the file
au	Author (person who created the file)
lg	Logon of the person who last modified the file.
cd	Date the file was created.
cm	Comments stored with the file.
rv	Revision number of the file.
kv	Keywords in the file.
rp	Retention period of the file.
pj	Project number of the file.

df hi=%f au rv cd cm Example of HI setting.

See the description of the HT default setting for information about adding titles to the directory display.

DF HT

Header Title — Specifies the titles of the Document Info directory display. The default setting is: Filename Author Saved Comment.

df ht=*t1 t2*. . . Format of the HT setting, where *t1 t2* . . . are the titles, spaced as you want them to appear in the directory.

df ht=File Author Rev Creation Date Comments Example of HT setting.

DF HV

Hyphenation Values — Specifies the minimum size of words to be hyphenated and the fewest number of characters allowed before and after a hyphen. (The initial default is 6,3,3.)

df hv=*n1,n2,n3* Format of the HV setting, where *n1* is the minimum number of characters in a hyphenated word, *n2* is the minimum number of characters that can appear before a hyphen, and *n3* is the minimum number of characters that can appear after the hyphen.

df hv=5,2,3 Example of the HV setting.

DF IM

Image Mode Printing — Specifies whether image mode printing is on or off. In image mode, the printer rasterizes text and outputs it as images, producing a higher print quality. (The initial default is 0.)

df im=0 Turns off image mode printing.

df im=1 Turns on image mode printing for printers that support it. (Because of its impact on speed, this setting is not recommended for laser printers.)

DF IO

Information On/Off — Specifies whether the Document Info feature is on or off. (The initial default is 1.)

df io=0 Turns off the Document Info feature.

df io=1 Turns on the Document Info feature.

df io=2 Displays the Document Info dialog box when you store a file without first completing the comments section of Document Info.

DF IT

Insert Cursor Type — Defines cursor values, which are passed to the BIOS interrupt routine. This setting is required only on certain types of hardware, and is established during installation.

df it=*a,b,c,d,e,f,g,h* Format of the IT setting, where:

a-c define cursor values for the current EGA mode (25-line or 43-line). Within these three arguments, *a* defines the cursor for insert mode; *b* defines the cursor for overstrike mode; and *c*, which is always 8193, defines the cursor off mode.

d-f define the cursor values for the alternate EGA mode, in the same order described above.

g defines the cursor value for DOS.

h is always 1, which is the value used when EGA emulation is off.

The following examples are based on 25-line EGA mode. If you are in 43-line EGA mode, reverse the order of the current and alternate values.

df it=519,1543,8193,264,1792,8193,1543,1 Sets cursor values for a color monitor with an EGA card.

df it=1037,3085,8193,264,1543,8193,0,1 Sets cursor values for a Hewlett Packard Vectra RS/20 with an HP VGA card.

df it=1037,1543,8193,264,1543,8193,1037,1 Sets cursor values for an IBM XT/286 with an IBM EGA card.

DF IU

Information Menu — Tells XyWrite to store a file after the Document Info dialog box has been completed. (The initial default is 1.)

df iu=0 Return to the displayed document after clearing the Document Info dialog box.

df iu=1 Store the displayed file after completing the Document Info dialog box.

DF JB

Job Begin — Sends the specified printer codes to the printer at the beginning of every document. The printer codes are defined in the PC (Printer Control) table in the printer file. Refer to "Printer Files" for more information about PC tables.

df jb=#,#,#... Format of the JB setting, where # is the number of a printer code in the PC table.

df jb=170 Example of the JB setting.

DF KC

Key Click — Enables XyWrite to generate an audible click at every press of a key. The speed of the hardware affects the duration of the click, so if you have a faster machine you should increase the duration setting accordingly. We recommend multiplying the duration by 4 on 386 machines. (The initial default is 0,0, which turns the key click off.)

df kc=*t*,*d* Format of the KC setting, where *t* defines the tone of the beep and *d* defines the duration of the click. Both are numbers between 0 and 65534.

df kc=256,2048 Example of KC setting.

DF KS

Keyboard Speed — Adjusts the repeat rate of the keyboard. Repeat rate refers to the length of time a key must be pressed before it starts to repeat and the speed with which it repeats. (The initial default is 0,1.)

df ks=*s*,*l* Format of the KS setting, where *s* is a number between 0 and 31 that indicates the repeat rate (0 is the fastest; 31 is the slowest); and *l* is a number between 0 and 3 that indicates the lag time (0 is the shortest; 3 is the longest).

df ks=10,1 Example of the KS setting.

DF L0

Action Bar Color Control — Specifies the colors of the various items in the action bar.

df l0=*i*,*h*,*a*,*b* Format of the L0 setting, where *i* sets the color for items in the action bar; *h* sets the color for highlighted action bar items; *a* sets the color for the accelerator character in each action bar item; and *b* sets the background color of the action bar.

df l0=117,125,116,117 Example of the L0 setting.

DF L1

Command Line Color Control — Specifies the colors of various items in the command line.

df l1=*c,t,w,f* Format of the L1 setting, where *c* sets the color for the current path; *t* sets the color for the text you type in the command line; *w* sets the color for any wild cards you used in command strings; and *f* sets the color for the function calls you include in your XPL programs.

df l1=87,92,101,84 Example of the L1 setting.

DF L2

Status Line Color Control — Specifies the colors of messages and other information displayed on the status line.

df l2=*l,m,b,o,s* Format of the L2 setting, where *l* sets the color for field labels and indicators; *m* sets the color for the standard messages and prompts that XyWrite displays; *b* sets the color for the background and the contents of fields, such as current filename, page number, etc.; *o* is the color for messages that require a user response; and *s* sets the color for warning messages.

df l2=93,87,95,93,84 Example of the L2 setting.

DF L3

Ruler Line Color Control — Specifies the colors of various items on the ruler line.

df l3=*d,c,m,t* Format of the L3 setting, where *d* sets the color for the dots and numbers in the ruler line; *c* sets the color of the cursor location on the ruler line; *m* sets the color of the tab indicators in the ruler line; and *t* sets the color of the dot or marker that indicates the active column in a table.

df l3=93,125,86,205 Example of the L3 setting.

DF L4

Pull-Down Menu Colors — Defines the colors of the pull-down menus. (The initial default is 15,112,112,7,112.)

df l4=*i,h,a,d,l* Format of the L4 setting, where *i* is the color of available items in the pull-down menu; *h* is the color of the highlighted item; *a* is the color of the accelerator character; *d* is the color of the dimmed (or unavailable) items in the menu; and *l* is the color of highlighted items in the list box.

df l4=117,125,116,120,112 Example of L4 setting.

DF LC

Line-End Character — Specifies the character to be displayed as a carriage return (hard return). (The initial value is ←.)

df lc=*c* Format of the LC setting, where *c* is any character.

df lc=↑ Example of the LC setting.

DF LH

Low-High — Specifies the size and baseline for superscript and subscript characters in the Bitstream Speedo fonts. This setting has no effect on printer fonts (i.e., built-in fonts, soft fonts, or cartridge fonts).

df lh=*s,u,d* Format of the LH setting, where *s* is the percentage of text size to be used for subscript and superscript characters and *u* and *d* determine the baseline for superscript and subscript characters, respectively. The *u* and *d* values are percentages of *s* that define how much you want to move the baseline up or down.

df lh=75,45,25 Example of the LH setting. For this example, if the text is set in 12-point type, then the superscript and subscript characters will be in 9-point type (75% of 12-point). Superscript text will be set 4.09 points (45% of 9 point) up from the baseline and subscript text will be set 2.25 points (25% of 9 point) below the baseline.

DF LZ

Format Redlining Date — Defines the way the date is displayed in Redlining logon strings. (The initial default is MM/DD.)

df lz=*date* Format of the LZ setting, where *date* is the month and day and any punctuation you want XyWrite to use when Redlining changes are made. Refer to the DZ setting above for information on available formats.

df lz=DD-MM Example of the LZ setting.

DF MA

Find Match — Defines the number of characters that XyWrite uses to find a match when comparing two files. The number may not be less than 2. (The initial default is 40.)

df ma=*n* Format of the MA setting, where *n* is the number of characters to base a match on.

df ma=10 Example of MA setting.

DF MF

Mode for Forms — Defines the mode you want to use for creating the blank fill-in areas in forms. (The initial default is NM.)

df mf=*m* Format of the MF setting, where *m* is the mode you want to use for the blank fill-in areas in forms.

df mf=ul Example of the MF setting.

DF MG

Message — Displays a message on the status line that is overwritten only by system messages. The default message is restored after the system message clears.

df mg=*message* Format of the MG setting, where *message* is any message you want displayed on the status line as a default.

df mg=Have a Nice Day! Example of the MG setting.

Notes:

- When specifying a default message from the command line, you must use the /NV switch; otherwise XyWrite will not recognize any spaces you type in the message prompt.
- When a default message is displayed, you cannot see the descriptions of the command markers that normally appear on the status line when you put the cursor on a marker.

DF MR

Metric Ruler — Specifies the spacing of the ruler markers. (The initial default is 10.)

df mr=10 Establishes 10ths of an inch as the spacing of the ruler markers.

df mr=4 Establishes centimeters as the spacing of the ruler markers.

DF MT

Military Time — Modifies the way time is displayed by the SEC, TM, and NOW commands. (The initial default is 0.)

df mt=0 Displays time in AM/PM format (for example, 10:30).

df mt=1 Displays time in 24-hour format (for example, 22:30).

DF MW

Maximize Windows — Specifies how window border and scroll bars are displayed. (The initial default is 1.)

df mw=0 Uses the maximum screen size for all windows, which means there is no border (or scroll bar) displayed.

df mw=1 Displays a one-unit-wide border around all windows; when a file is displayed, the right and bottom borders are scroll bars. When MW=1, the usable text area is 78 by 20 (rather than 80 by 22).

df mw=2 Staggers the borders of windows, so each window you open is displayed one unit in and 1 unit down from the previous window. As with MW=1, the right and bottom borders are scroll bars when a file is displayed. The usable text area varies, depending on the order in which the window was opened.

df mw=4 Remembers the size of each window, and uses that size the next time you open the window.

You can add two or more values to apply more than one effect to windows. For example, DF MW=5 displays a one-unit-wide border around all windows, and also remembers the size of windows.

DF NC

Normal Carriage Return — Defines whether XyWrite sends the carriage return to the printer in normal mode or in the current mode. (The initial default is 1.)

df nc=1 XyWrite puts out carriage returns in normal mode (MD NM).

df nc=0 XyWrite puts out carriage returns in the mode that is in effect at the end of the line.

DF ND

Network Drives — Specifies the drives that are shared by network users. XyWrite applies file locking to these drives, so that the same file cannot be opened for editing by more than one person at a time.

df nd=*d1,d2,d3,d4,d5. . .d26* Format of the ND setting.

df nd=f,x Example of the ND setting.

DF NE

No Errors from Printer — Defines whether XyWrite checks for printer status. (The initial default value is 0.)

df ne=0 Checks printer status and displays an error message if the printer returns an error.

df ne=1 Does not check printer status except to verify that the printer is ready (on-line).

DF NL

Network Login — Defines the location of login files. The login location is established for both stand-alone and LAN environments.

df nl=*d:\path* Format of the NL setting, where *d:* is the drive and *\path* is the subdirectory that contains the login files.

df nl=x:\login Example of the NL setting.

DF NW

New Window — Enables XyWrite to open a new window when needed. (The initial default is 3.)

df nw=0 Disables automatic windows. You must open and close windows manually.

df nw=1 Enables the automatic opening and closing of windows. XyWrite opens a new window every time you execute a command that requires one (e.g., NEW or CALL with a file on-screen) and closes the window whenever you issue an ABORT command. You can display a directory, call a file, look it over, abort it, and then return to the directory to call up another file; XyWrite automatically opens a new window when necessary and keeps the directory in an old window.

df nw=2 Same as NW=1 except XyWrite does not automatically close the current window when you execute ABORT.

df nw=3 Same as NW=1 except when you call a file with the directory displayed, the file replaces the directory.

DF O1

Options — Specifies how XyWrite performs error correction for differences between screen and printer fonts. (The initial default is 0.)

df o1=0 Performs error correction between words.

df o1=1 Performs error correction between characters. (This setting improves the display of boxes created with the BOX command.)

DF OB

Overstrike Beep — Defines the tone and duration of the beep you hear whenever you overstrike a character. The speed of the hardware will affect the duration of the beep so if you have a faster machine you should increase the duration setting accordingly. We recommend multiplying the duration by 4 on 386 machines. The initial setting is 0,0 which turns the overstrike beep off.

df ob=*t,d* Format of the OB setting, where *t* defines the tone of the beep and *d* defines the duration of the beep. Both are numbers between 0 and 65534.

df ob=756,10000 Example of OB setting.

DF OD

Offset Display - Determines whether XyWrites hides or shows the offset (specified by the OF command) in formatted view. Also controls how the offset is displayed in graphic view. (The initial default is 0.)

df od=0 Hide the offset.

df od=1 Show the offset.

df od=4 Show offset as a gray screen in graphic view (if not set, the offset appears in the same background as the text).

DF OE

Open Editor — On a LAN, defines whether you want XyWrite to leave EDITOR.EXE open or closed after reading it. When EDITOR.EXE is open, performance is faster. However, on some networks, users may encounter sharing violations when it is left open. (The initial default is 0.)

df oe=0 Closes EDITOR.EXE after reading it.

df oe=1 Leaves EDITOR.EXE open after reading it.

DF OL

Outline Fonts — Specifies the location of the outline fonts. (The initial default is C:\BTFONTS\.)

df ol=*d:\path\* Format of the OL setting, where *d:* is the drive and \*path*\ is the subdirectory that contains the outline fonts. (Note that the closing backslash is required.)

df ol=d:\btfonts\ Example of the OL setting.

DF P.

Truncated Path Name — Adjusts the length of the path name XyWrite displays on the status line, from 1 to 14 characters. (The initial default is 14.)

df p.=*n* Format of the P. setting, where *n* is the number of characters of the current directory name that you want XyWrite to display on the status line.

df p.=5 Example of the P. setting.

DF PD

Pad Spaces — Works in conjunction with linear cursor keys, which let you move the cursor off to the right of a line of text and begin to enter new text. PD (Pad Spaces) determines whether spaces are added to fill the gap to the *left* of the cursor. To illustrate, imagine the following: You move the cursor to the right (enabled by Linear Right in the keyboard file), past the end of a line of text (past the line-end arrow). You then begin typing. If PD=1, the text is inserted exactly where the cursor is located, and the gap between the line-end arrow and the new text is filled in with spaces. If PD=0, the cursor jumps left to the line-end arrow and the text is inserted adjacent to the existing text. (The initial default is 0.)

df pd=0 Text will be inserted at the left of the line-end arrow.

df pd=1 Spaces will be inserted to fill the gap between line-end arrow and cursor position.

DF PK

Page Break Color — Specifies the color that is used for page breaks. (The initial default is 7.)

df pk=*n* Format of the PK setting, where *n* is the number of the display mode for the page break.

df pk=9 Example of the PK setting.

DF PX

Page Break Character — Specifies the character that is displayed for page breaks. (The initial default is 196, which produces —.)

df px=*n* Format of the PX setting, where *n* is the number of the XyWrite character you want displayed for page breaks.

df px=95 Example of the PX setting.

DF R2

Mouse Double Click — Sets the time interval for a double click. The time is measured in 1/20 of a second. (The initial default is 8.)

df r2=*n* Format of the R2 setting, where *n* is the time interval.

df r2=5 Example of the R2 setting.

DF RB

Reverse Buttons — Reverses the assignment of the mouse functions from the left to the right button and vice versa. (The initial default is 0.)

df rb=0 Assigns right-hand operation.

df rb=1 Assigns left-hand operation.

DF RD

Redline Data — Defines the display of the redline tags that identify who made an edit and when it was made. (The initial default is 1,14,9,0,249.)

df rd=*o,n,m,p,c* Format of the RD setting, where *o* turns the data capture on or off (0 for off or 1 for on); *n* is the number of captured characters displayed in the text area; *m* is the display mode number for the captured information; *p* turns a prompt line display of the captured data on and off (0 for off, 1 for on); and *c* is the character that separates the captured data from the text.

df rd=1,9,7,1,249 Example of the RD setting.

DF RI

Mouse Repetition Interval — Adjusts the rate the document scrolls when using the mouse in scroll bars and when selecting text with the drag method. (The initial default is 5.)

df ri=*s* Format of the RI setting, where *s* is the scroll speed, measured in 1/20 of a second.

df ri=7 Example of the RI setting.

DF RL

Ruler Markers — Specifies the characters shown on the third line of the display — the ruler. (The initial default ▶◀▼⌂└┐░├┤•·).

df rl=*abcdefghijk* Format of the RL setting, where *a* is the Tab marker for Flush Left Tabs (TS 1); *b* is the Tab marker for Flush Right Tabs (TS 1R); *c* is the Tab marker for Flush Center Tabs (TS 1C); *d* is the Tab marker for Decimal Tabs (TS 1D); *e* is the marker for the first Indent Paragraph value; *f* is the marker for the second Indent Paragraph value; and *g* is the marker for the position of the cursor column (*g* cannot be changed — it is always character 176); *h* is the left margin marker (established by the LM command); *i* is the right margin marker (established by the RM command); *j* is the column marker for every 5th column; and *k* is the standard column marker. You can substitute any characters for a ruler marker (except for *g*). If you choose to use a ruler symbol other than the original XyWrite setting, be sure to follow this exact order to replace the correct one.

df rl=lrcd └┐░├┤•· Example of RL setting.

DF RN

Round Off Numbers — Specifies the way XyWrite handles fractional line numbers in the header. (The initial default is 0.)

df rn=0 Causes XyWrite to display the actual line number values when you are in draft view with page breaks on.

df rn=1 Causes XyWrite to round off line number values in the header to the nearest half (for example, 1-5.86 would become 1-6, while 1-5.3 would become 1-5.5).

DF RS

Record Separator — Specifies the character or string you want to use to separate records in a data file. (The initial default is a carriage return/line feed combination.)

df rs=*r* Format of the RS setting, where *r* is the character or string you want to use as a record separator. If you want to use a carriage return/line feed as *part* of the record separator, you must enter it as Ctrl Alt 13 Ctrl Alt 10. XyWrite enters it like this: ♪◙.

df rs=♪◙♪◙ Example of the RS setting.

DF RX

Ratio for X Direction — Sets the mouse-to-pixel ratio for horizontal mouse movement. (The initial default is 15.)

df rx=*n* Format of the RX setting, where *n* is a number from 1 to 32767. The higher the number, the less the cursor moves.

df rx=25 Example of the RX setting.

DF RY

Ratio for Y Direction — Sets the mouse-to-pixel ratio for vertical mouse movement. (The initial default is 15.)

df ry=*n* Format of the RY setting, where *n* is a number from 1 to 32767. The higher the number, the less the cursor moves.

df ry=25 Example of the RY setting.

DF RZ

Record Size — Defines the maximum number of characters that XyWrite recognizes when sorting data file records. The initial default is 3500.

df rz=*n* Format of the RZ setting, where *n* is the number of characters you want in a record.

df rz=5000 Example of the RZ setting.

DF SG

Text Macro — Assigns a string of text or numbers to a text macro key.

df sg=*n,string* Format of the SG setting, where *n* is any text macro (A-Z or 000-999) and *string* is the text or numbers you want to save to the text macro.

df sg=999,yes Example of the SG setting.

Note: When assigning a text macro from the command line, you must use the /NV switch; otherwise, XyWrite will not recognize any spaces you type in the message prompt.

DF SI

Screen Resolution — Specifies the screen resolution in pixels per inch. (We recommend 96,72 for EGA and HGA; 96,96 for VGA; and 96,48 for CGA.)

df si=*h,v* Format of the SI setting, where *h* defines the horizontal screen resolution in pixels per inch and *v* defines the vertical screen resolution in pixels per inch.

df si=96,72 Example of the SI setting.

DF SK

Sort Key — Defines the sorting rules, used by the SORT command, including the number of characters used in the sort. (The initial default is 1,80.)

df sk=*n1,n2* Format of the SK setting, where *n1* is one, or a combination of, the following numbers:

0 Sorts numbers by first digit (01, 10, 2, 3)
1 Sorts numbers in decimal order (01, 2, 3, 10)
2 Sorts entries in reverse order (Z to A, 9 to 0)
4 Deletes duplicate entries.

n2 is the number of characters to be used for sorting.

df sk=1,40 Example of the SK setting.

DF SL

Screen Length — Defines how many lines to display on a terminal that has more than the standard 25 lines. Of the number you choose, the first three lines will belong to the header and the remainder will be the text area. *The SL setting should not be used with any of the standard 25-line displays.* If you are using an EGA adapter, refer to the EG command. (The initial default is 25.)

df sl=*n* Format of SL setting, where *n* is the number of screen lines.

df sl=44 Example of the SL setting.

DF SM

Show Menus — Defines the way the action bar behaves. (The initial default is 1.)

df sm=0 Causes the action bar to disappear after you make a choice from a pull-down menu. The action bar will not reappear until you press F10.

df sm=1 Causes the action bar to remain at all times unless you press F5 to access the command line. The action bar will reappear when you press F10.

DF SO

Sort Setting — Defines the field or section of the field that you want XyWrite to use for sorting a data file with the SORTD command.

df so=*field1,field2. . .fieldn* Format of the SO setting, where *field1, field2. . .fieldn* are made up of one or more of the following parameters:

F# Field number (F1, F2, ...).
W# Word within field (W1 is the first word; W2 is the second word, etc.) A negative number means count from the back of the field. (The default value is the entire field.)
L# Length of field, or number of characters to use in the sort. (The default value is 20.)
N Sort numbers in numerical order, e.g., 1, 2, 5, 10, 20, 100. (The default is to sort numbers by first digit, e.g., 1, 10, 100, 2, 20, 5.)
R Reverse the sorting order.

df so=f1w3l5nr Example of the SO setting.

DF SQ

Sequential Page Numbering — Defines the way the PRINT,*a-b* command counts pages as it prepares to send them to the printer. (The initial default is 0.)

df sq=0 Enables the PRINT command to honor the printed page number. Thus, print chapter,5 would print the page which has page number 5 printed on it. This is the number the Page-Page Depth indicator shows.

df sq=1 Sets the PRINT command to refer to the pages sequentially, starting at the beginning of the file. This setting disregards the printed page numbers (which can be changed with the Set Page Number command). Thus, print chapter,5 would print the fifth page in the file, regardless of its printed page number.

DF ST

Show Tab — Specifies how tabs are displayed on the screen. (The initial default is 1.)

df st=0 Causes the tabs to appear on the screen as spaces.

df st=1 Causes the tab character (◌) to appear on the screen in expanded view; tabs appear as spaces in all other views.

df st=2 Causes the tabs to appear as a right arrow (→) in draft views and as a space in expanded and graphic view.

df st=3 Causes the tabs to appear as ◌ in expanded view and → in draft view.

DF SW

Screen Width — Defines how many "columns" (or characters) you want to display on a monitor that has a wide screen mode (one that lets you display up to 132 columns on the screen). If your monitor supports a wide screen mode, you must *set it up in DOS before starting XyWrite.* (Refer to the documentation supplied with your computer for information on how to do this.) Once in XyWrite, set SW to match the DOS setting.

df sw=*n* Format of the SW setting, where *n* is the number of columns wide you want your screen to be. This number must match the DOS setting. (Standard columns are 40, 80, 120, and 132.)

df sw=132 Example of the SW setting.

Note: Wide screen mode has no effect on graphic view.

DF TB

Tab Character Control — Defines how tabs will print: either as a tab character or as spaces. (The initial default is 0.)

df tb=0 Causes XyWrite to convert every tab to equivalent spaces on output to the printer.

df tb=1 Causes XyWrite to send tabs in the file to the printer as the tab character (○), ASCII 9. This setting is valuable for printing to disk (PRINTF) when you want to retain tab characters. Use this setting when you want to produce a file (with PRINTF) that keeps its tabs (rather than converts them to spaces).

DF TF

Ignore Top Margin — Defines how XyWrite interprets the top margin (TP) embedded command. (The initial default is 0.)

df tf=0 Causes XyWrite to honor the top margin (TP) command.

df tf=1 Causes XyWrite to ignore the top margin (TP) command. You would use this if the top margin has been set by hand and the file contains a TP command that you want to override.

DF UH

Unit Horizontal — Defines the default horizontal unit of measure for input and display. (The initial default is IN,IN).

df uh=*entryunit,displayunit* Format of the UH setting, where *entryunit* is the size of one default horizontal entry unit (the unit of measure you use to enter arguments) and *displayunit* is the size of one default horizontal display unit (the unit of measure you want XyWrite to display). Valid units are in (inches), di (tenths of an inch), pi (picas), pt (points), cm (centimeters), mm (millimeters), dd (didots), and ci (ciceros). We recommend that you use the same value for *entryunit* and *displayunit*.

df uh=dl,dl↵ Example of the UH setting.

DF UL

Underline Setting — Specifies how tabs and spaces are underlined within text that is underlined with MD UL. (The initial default is 1.)

df ul=0 Underline everything.

df ul=1 Underline everything but tabs.

df ul=2 Underline everything but tabs and spaces.

df ul=3 Underline only letters and numbers.

DF UM

Unhide Modes — Turns the display of mode markers on and off. (The initial default is 0.)

df um=0 Does not display markers for mode commands.

df um=1 Displays markers for mode commands.

DF UN

Untitled File — Specifies whether XyWrite opens an untitled file when you close the last file. UN works in conjunction with the NW (New Window) setting. (The initial default is 0.)

df un=0 Does not automatically create an untitled file.

df un=1 Automatically creates an untitled file in the last open window when DF NW=3.

DF UR

Use Pointer — Turns the mouse pointer on and off.

df ur=0 XyWrite does not recognize mouse movement.

df ur=1 XyWrite recognizes mouse movement.

DF UV

Unit Vertical — Defines the default vertical unit of measure for input and display. (The initial default is IN,IN.)

df uv=*entryunit,displayunit* Format of the UV setting, where *entryunit* is the size of one default vertical entry unit (the unit of measure you use to enter arguments) and *displayunit* is the size of one default vertical display unit (the unit of measure you want XyWrite to display). Valid units are in (inches), di (tenths of an inch), pi (picas), pt (points), cm (centimeters), mm (millimeters), dd (didots), ci (ciceros), and li (lines). We recommend that you use the same value for *entryunit* and *displayunit*.

df uv=ll,ll Example of the UV setting.

DF VF

Variable Forms — Enables you to extend a form field into more than one line. (The initial default is 0.)

df vf=0 Does not extend the form field. Pressing ↵ moves the cursor to the next field.

df vf=1 Extends the form field so you can enter multiple lines.

DF WA

Error Message Wait Time — Defines how long an error message will display before it is replaced by another message. (The initial default is 18.)

df wa=*t* Format of the WA setting, where *t* is the minimum length of time the message will display. A value of 18 is 1 second; 36 is 2 seconds, etc.

Note: WA also affects the speed at which programs execute when DB is active.

DF WB

Window Border Characters — Defines the characters that are used for the borders of the windows.

df wb=*abcdefghijklmn* Format of the WB setting, where *a* is the character in the upper left corner; *b* is the character in the upper right corner; *c* is the character on the horizontal sides; *d* is the character in lower left corner, *e* is the character in the lower right corner; *f* is the character on the vertical sides; *g* and *h* are the up and down indicators on the vertical scroll bars; *i* and *j* are the right and left indicators on the horizontal scroll bars; *k* and *m* are the characters in the vertical and horizontal scroll bars, respectively; and *l* and *n* are the characters in the vertical and horizontal slider box.

df wb=┌ ┐─└ ┘│↑↓→ ← ▒ █ ░ █ Example of the WB setting.

DF WF

Wrap to Fit - Specifies the way line endings are displayed in draft view and expanded view. This setting has no effect on formatted or graphic view. The initial default is 1.

df wf=0 In draft view, displays line endings according to the format you have established.

df wf=1 In draft view and in expanded view, displays text within the current window borders. Hyphenation is not applied. You do not have to scroll horizontally to view text.

DF WO

Word Overstrike Editing — Defines the overstrike typing mode for text entry. (The initial default is 0.)

df wo=0 Causes text entered at the cursor to overwrite text characters and word separators *except* carriage returns.

df wo=1 Causes text entered at the cursor to overwrite all text and word separators *except* the space, tab, and carriage return characters.

df wo=2 Causes text entered at the cursor to overwrite text characters but not word separators.

DF WS **Whole-Space Justification** — Specifies whole-space or microspace justification. (The initial default is 0.)

df ws=0 When you specify justification (with JU), causes XyWrite to justify text using *partial* spaces — what we call "microjustification." The spaces are inserted between characters and words.

df ws=1 When you specify justification (with JU), causes XyWrite to justify text by adding *whole* spaces between words (rather than using partial spaces) and no space within words. Whole-space justification is much faster than microjustification on some printers, and may be preferred in those cases.

DF WW **Conversion Filters Path** — Specifies the location of the document conversion filters. The initial default is C:\XY4\FILTERS\.

df ww=*d:\path\* Format of the WW setting, where *d:* is the drive and *\path* is the subdirectory that contains the filters. (Note that the closing backslash is required.)

df ww=d:\xy4\filters\ Example of the WW setting.

DF XD **Read-Only Directories** — Specifies whether directories are read-only or read/write. (The initial default is 1.)

df xd=0 Makes directories read/write.

df xd=1 Makes directories read-only and highlights the line that the cursor is on.

DF XF **Extract Fields** — Defines the fields that you want the SORTD command to extract from the data file and output to the sorted file.

df xf=*f1,f2...* Format of the XF setting, where *f1,f2...* are the fields you want to extract.

df xf=f1,f3 Example of the XF setting.

DF XL **Selective Directory** — Enables the selective directory feature of XyWrite. A selective directory is one that lists only files that match certain criteria. These criteria, which are based on the summary information maintained by Document Info, are defined in the XL setting. You display the selective directory by adding the /SL switch to the DIR command (for example, type: [F5]dir/sl \xy4[↵]).

df xl=*expression* Format of the XL setting, where *expression* defines the criteria you want XyWrite to use when selecting files for a selective directory. Refer to "User Programming" for information on building expressions, such as a list of available operators.

df xl=«ya^au»==("ruth") Example of the XL setting.

DF XM

Transpose Messages — Specifies the order in which XyWrite displays Page-Depth Number, Filename, and Time indicators on the status line. (The initial default is *PL *FI *TI.)

df xm=*in1 in2 in3* Format of the XM setting, where *in1 in2 in3* are the following indicator mnemonics, separated by a space, in the order you want them to appear on the status line:

*PL Page-line number
*FI Filename
*TI Time

df xm=*PL *FI Example of the XM setting.

DF XR

Extract Records — Specifies the records you want SORTD to extract.

df xr=*fnop"variable"* Format of the XR setting, where *fn* is the field number; *op* is the symbol for the condition you are testing for; and *"variable"* is what you are comparing the field against. *"Variable"* must be enclosed in double quotation marks as shown.

df xr=f8=="01824" Example of the XR setting.

DF XT

Expand Triangles — Defines whether XyWrite displays an abbreviated or expanded explanation of the marker under the cursor. (The initial default is 1.)

df xt=0 Causes XyWrite to display only the command and its value on the status line (for example, LM1).

df xt=1 Causes XyWrite to display an explanation of the command as well as its value (for example, Left Margin 1). (Menus must be loaded for this setting to work.)

DF XY

XyWrite Screen Fonts — Defines the location of the file SCRFONTS.BIN. The initial default is C:\BTFONTS.

df xy=*d:\path\* Format of the XY setting, where *d:* is the drive and *\path* is the subdirectory that contains the screen fonts. (Note that the closing backslash is required.)

df xy=d:\btfonts\ Example of the XY setting.

Note: If you run other application programs that use BITSTREAM screen fonts, the fonts must reside in the BTFONTS directory off the root directory.

DF ZM

Zoom - Enlarges or reduces graphic view by the amount you specify. The default is 75, which means that the full width of a standard 8.5-inch page is displayed in graphic view.

df zm=*n* Format of the ZM setting, where *n* is the percentage (from 5-400) that you want to enlarge or reduce the display in graphic view. If you specify 100, the text and graphics will be displayed in approximately the same size that they will be printed.

df zm=85 Example of the ZM setting.

DF ZS

Point Sizes — Defines the point sizes that are displayed in the menus for scalable fonts. (The initial default is 8, 9, 10, 12, 14, 18, 24, 36, 48, 72, 96, 144.)

df zs=*size1,size2,. . .* Format of the ZS setting, where *size1,size2,. . .* is a list of the point sizes you want displayed in the point size menu for scalable fonts.

df zs=10,12,14 Example of the ZS setting.

Display Mode Colors

INTRO

Color settings are another customizable feature of XyWrite. They constitute the second major category of information in SETTINGS.DFL.

Display Mode Colors MD

FORMAT MD *nn*=#

nn is a two-letter mode command or a number from 0-255
is a number from the color table (see Notes #5 and #6).

MENU Advanced Preferences Defaults

PURPOSE MD (Mode) commands define the color associated with each display mode in draft view. XyWrite provides recommended display values for the following types of monitors:

- Color (COLOR.DSP)
- Monochrome (MONO.DSP)
- Gas plasma (GAS.DSP)
- Gray-scale (G-SCALE.DSP)
- LCD (LCD.DSP)

When you install XyWrite, you are asked what type of monitor your system has. The display values for your monitor are merged from the separate display file into SETTINGS.DFL.

You use the MD command only when you want to change a color from the default value. You can change the values in SETTINGS.DFL or create a custom color file that you load separately.

ACTION

Creating a Color File

To create a custom color file:

1. Determine what colors you want to use for each mode. To display the available colors, choose "Color choices" from the Help index. If you don't have the help file loaded, refer to Notes #5 and #6 for a list of available colors.

2. Make a copy of the display file that most closely resembles what you want. For example:

   Type: F5 copy color.dsp,custom.dsp↵

3. Open the new file you created in step 2.

   Type: F5 call custom.dsp↵

4. Search for the display mode you want to change. For example:

   Type: F5 se /md bo/↵

5. Immediately after the equals sign, type the number of the new color.

6. Repeat steps 4 and 5 for each display mode you want to change.

7. Store and load the color file. For example:

   Type: F5store↵
   Type: F5load custom.dsp↵

NOTE #1

Startup. You can have your custom color file automatically loaded at startup by adding a line to your STARTUP.INT file. For example, the following entry loads a display file named CUSTOM.DSP.

```
BC load custom.dsp←
```

Be sure to add this line *after* the line that loads SETTINGS.DFL; remember, SETTINGS.DFL contains color settings and the last set of values loaded is the one that takes precedence. Refer to "Startup File" for more information on modifying STARTUP.INT.

NOTE #2

Selected Text. If you change the display colors, you may get unexpected results when you select a block of text. The DD default setting interacts with the display modes to produce the color of selected text. You may need to adjust the value of the DD setting after you load your custom colors. Refer to "Default Settings" for more information about the DD setting.

NOTE #3

Mode Numbers. In addition to the display modes defined by two-letter mode commands (NM, BO, etc.), you can define colors for display modes 0-255. Do not confuse these mode numbers with color numbers; while the default values are often the same (e.g., MD 15=15), they are not equivalent.

Another important note about mode numbers relates to the way XyWrite interprets them for graphic view. If you use the embedded MD command with a mode number to format your document (e.g., MD 113) and then display the document in graphic view, you will get unexpected results. That is because XyWrite uses the following formula to determine how numbered modes appear in graphic view:

1. XyWrite subtracts the mode number from 255. For MD 113, the result would be 142.

2. XyWrite compares the result to the following chart to determine which bits are set.

Mode	Bit
Normal	0
Bold	1
Underline	2
Reverse	4
Superscript	8
Subscript	16
Italic	32
Insert	64
Delete	128

Result: For the current example, the following bits are turned on: 128, 8, 4, and 2 (128 + 8 + 4 + 2 = 142). That means that text that is in MD 113 would display in graphic view as struck-through (Delete), superscript, underline, and reverse.

When you print the document, the numbered modes are printed using the definition specified in the Print Type (PT) table in the printer file.

NOTE #4 Standard Colors. XyWrite comes with the following display values (for a color monitor).

MD NM=23	White (dim) on blue
MD BO=31	Bright white on blue
MD UL=19	Cyan on blue
MD BU=27	Light cyan on blue
MD IT=26	Green on blue
MD BI=30	Yellow on blue
MD RV=113	Blue on white
MD SU=87	White (dim) on magenta
MD SD=96	Black on brown
DF DD=16	Define Display setting.

NOTE #5 Monochrome Table. On a monochrome monitor, the following table is in effect. Add 128 to a number to make it blink. "White on Black" means white characters on a black background.

0	Black on Black (invisible)
1	White on Black, underlined
2-7	White on Black, not underlined
8	Black on Black (invisible)
9	Bright White on Black, underlined
10-15	Bright White on Black, not underlined

(cont'd)

Display Mode Colors

112	Black on White
120	Black on White
128	Black on Black (invisible)
129	Flashing White on Black, underlined
131-135	Flashing White on Black, not underlined
136	Black on Black (invisible)
137	Flashing Bright White on Black, underlined
138-143	Flashing Bright White on Black, not underlined
240	Flashing Black on White
248	Flashing Black on White

The only values that don't display are: 0, 8, 128, and 136.

NOTE #6 **Color Table.** To select a color (for a color monitor), pick one color for the characters and another color for the background. Add the two numbers together. Use this number with the MD command. For example, for bright white characters (15) on a blue background (16), use 31.

Characters (Foreground)

0	Black Characters
1	Blue Characters
2	Green Characters
3	Cyan Characters
4	Red Characters
5	Magenta Characters
6	Brown Characters
7	White (Dim) Characters
8	Gray Characters
9	Light Blue Characters
10	Light Green Characters
11	Light Cyan Characters
12	Light Red Characters
13	Light Magenta Characters
14	Yellow Characters
15	Bright White Characters

Background

0	Black Background
16	Blue Background
32	Green Background
48	Cyan Background
64	Red Background
80	Magenta Background
96	Brown Background
112	White (Dim) Background
128	Blinking Characters, Black Background
144	Blinking Characters, Blue Background

160	Blinking Characters, Green Background
176	Blinking Characters, Cyan Background
192	Blinking Characters, Red Background
208	Blinking Characters, Magenta Background
224	Blinking Characters, Brown Background
240	Blinking Characters, White (Dim) Background

NOTE #7 **Using the Menus.** If you prefer, you can use the menus to create a custom color file. To do this, select "Screen — Type Style Colors" from the Default menu. To load the custom file, select "Screen — Color Sets" from the Default menu.

Customization Tables

INTRO

Customization tables are another way to change the built-in values of XyWrite. They constitute the third major category of information in SETTINGS.DFL. This section describes the following customization tables:

- PP (Pick Printer) table
- SE (Separator) table
- CS (Counter String) table

Pick Printer Table PP:

FORMAT

PP:#
d1 file1.prn notes1
.
.
dn filen.prn notesn

# is the number of port/printer file combinations defined in the table.
d1...dn is a printer port (see Note #1).

file1.prn is the printer file associated with *d1*
filen.prn is the printer file associated with *dn*

notes1 is the description of *d1*
notesn is the description of *dn*

The three columns are separated by tabs.

MENU

File Printer Setup...

PURPOSE

The PP table defines the information that appears in the Select Printer File dialog box. The PP table, which is created during the installation procedure, lists all printer port/printer file combinations available on your system and the printer file associated with each.

ACTION

Modifying the PP Table

To modify the PP table because, for example, you added a printer to your computer setup:

1. Call SETTINGS.DFL.

2. Move the cursor to the PP table, which is near the bottom of the file.

3. Since we're adding a printer, add 1 to the number after the PP:.

4. At the end of the table, type the port designation, a tab, the printer file you want loaded when you select this entry, a tab, and (optional) a description of the new printer. For example:

   ```
   LPT1    POST35N.PRN    PostScript printer
   ```

5. Store and load SETTINGS.DFL.

Result: The next time you issue the SETP command, the new printer will be listed as one of the choices.

NOTE #1

Print Destination. Your workstation has one or more ports which are designated LPT1, LPT2, LPT3, COM1, COM2, etc. These ports are used to communicate with local and network printers (and other output devices). If the printer is local, the port is set up to send information directly to it. If the printer is on the network, network software commands *intercept* the data from the port and *redirect* it to the network printer. If local printers exist, they are usually connected to the first ports (LPT1, LPT2, etc.); network printers use the next available ports.

In order for XyWrite to differentiate between local printer ports and ports that are being redirected by the network software, XyWrite uses different naming conventions. Enter the number 1 for a local *parallel* printer connected to port LPT1, the number 2 for a local *parallel* printer connected to port LPT2, etc. Enter LPT1 if the network software is set up to redirect data from LPT1 to a network printer, LPT2 if the network software is set up to redirect data from LPT2 to a network printer, etc.

EXAMPLE

The following table is an example of a PP table. Refer to SETTINGS.DFL to review the PP table XyWrite created for your setup during installation.

```
PP:3
LPT1    D:\XY4\POST35N.PRN          Apple LaserWriter II/NT
1       D:\XY4\EPFX850.PRN          Epson FX-850
LPT2    D:\XY4\HPLJ-3D.PRN          Hewlett Packard LaserJet IIID
```

Separator Table

FORMAT

SE:3
string1
string2
string3

string1 defines the characters XyWrite recognizes as word separators. *string2* defines the characters XyWrite recognizes as sentence separators. *string3* defines the characters XyWrite recognizes as paragraph separators.

PURPOSE

The SE (Separator) Table allows you to define the characters you want XyWrite to recognize as word, sentence, and paragraph separators. You can select the separator characters from the first 256 characters in the character set (see Appendix E for a complete list of characters). Once defined, XyWrite uses these separators when you perform a function on a word, sentence, or paragraph. Such functions include defining, deleting, spelling, hyphenating, and moving the cursor.

For example, if you press F4 to define the current paragraph, XyWrite looks for the paragraph separator before and after the current cursor location and defines the text between those two boundaries. If you press Ctrl F7 to check the spelling of the current word, XyWrite looks for the word separators before and after the cursor location to determine the boundaries of the word.

The default separators are:

Word: + = ( ) [ ] { } < > : ; ' " , . ! ¡ ? ¿ / ○ ^ ¦ ▲ £ ƒ x ÷ ° ≡ ± — ″ ¶ §
plus the line draw and fill characters.

Sentence: . ! ? ¿ ¡

Paragraph: ◙ ♪ →

If you want to use different separators, add an SE table to SETTINGS.DFL. You must include all three strings, and you must use all of the characters you want XyWrite to recognize, not just the ones you are adding.

ACTION

Adding an SE Table
To add an SE table to SETTINGS.DFL:

1. Open SETTINGS.DFL.

   Type: [F5]call settings.dfl[↵]

2. Move the cursor to the end of the file.

   Press: [Ctrl] [End]

3. Type in all four lines of the table. Remember, you must enter all of the characters you want XyWrite to recognize as separators.

4. Store and load SETTINGS.DFL.

Result: The new separators are now in effect.

NOTE

Default Separator Table. You can request a soft copy of the default separator table from the Technical Services Department.

Counter String Table

CS:

FORMAT

```
CS:n
string1
string2
.
.
.
stringn
```

n is the number of strings
string1, string2, . . . stringn define the characters you want used for customized numbering functions.

PURPOSE

The CS table lets you define a custom set of characters that you want XyWrite to use when it automatically numbers pages, chapters, footnotes, or other counters. Each string in the table can contain any number of characters, but they must be from the first 256 characters in the character set (see Appendix E for a list of characters).

The default custom counters are *, **, ***, and so on. If you want to use different custom counters, add a CS table to SETTINGS.DFL.

ACTION

Adding a CS Table

To add a CS table to SETTINGS.DFL:

1. Open SETTINGS.DFL.

   Type: F5 call settings.dfl ↵

2. Move the cursor to the end of the file.

   Press: Ctrl End

3. Type in the table. You can put in as many strings are you like. Just be sure that the number on the first line correctly indicates the number of strings in the table. For example:

   ```
   CS:4
   !
   @
   •
   $
   ```

4. Store and load SETTINGS.DFL.

Result: The new counters are now in effect. When you use this table for counting, the list of strings will eventually be exhausted. After this point, the first string will be used twice, then the second, and so on.

Keyboard File

INTRO

The keyboard is your connection to XyWrite — you can modify the keyboard functions to your liking. You can rearrange keys, and you can reduce the most complicated function to a single keystroke.

This section also covers the function calls, which form the foundation of XyWrite. Once you become familiar with the keyboard file, you may find yourself inventing new combinations of functions to best serve your purpose.

CONTENTS

Modifying Keyboard Files

PURPOSE

Did you ever wish you could change the meaning of the keys on the keyboard? In this section you will learn how to change a key or even the whole keyboard to suit your needs.

Here are two simple ways you can change the keyboard file:

- **Create shortcuts.** For example, assign the key combination CtrlD to be a shortcut for F5dir↵, to display a directory.
- **Assign new functions to keys.** You can alter the keyboard file to do things you could otherwise not do. You can assign any of the function calls you prefer to a key. For example, you can change the cursor up and down keys to Linear Up (LU) and Down (LD). Linear Up and Down allow the cursor to move directly up or down without ever switching columns.

You generally assign basic functions to the keyboard file. From these you can construct more elaborate routines in program files.

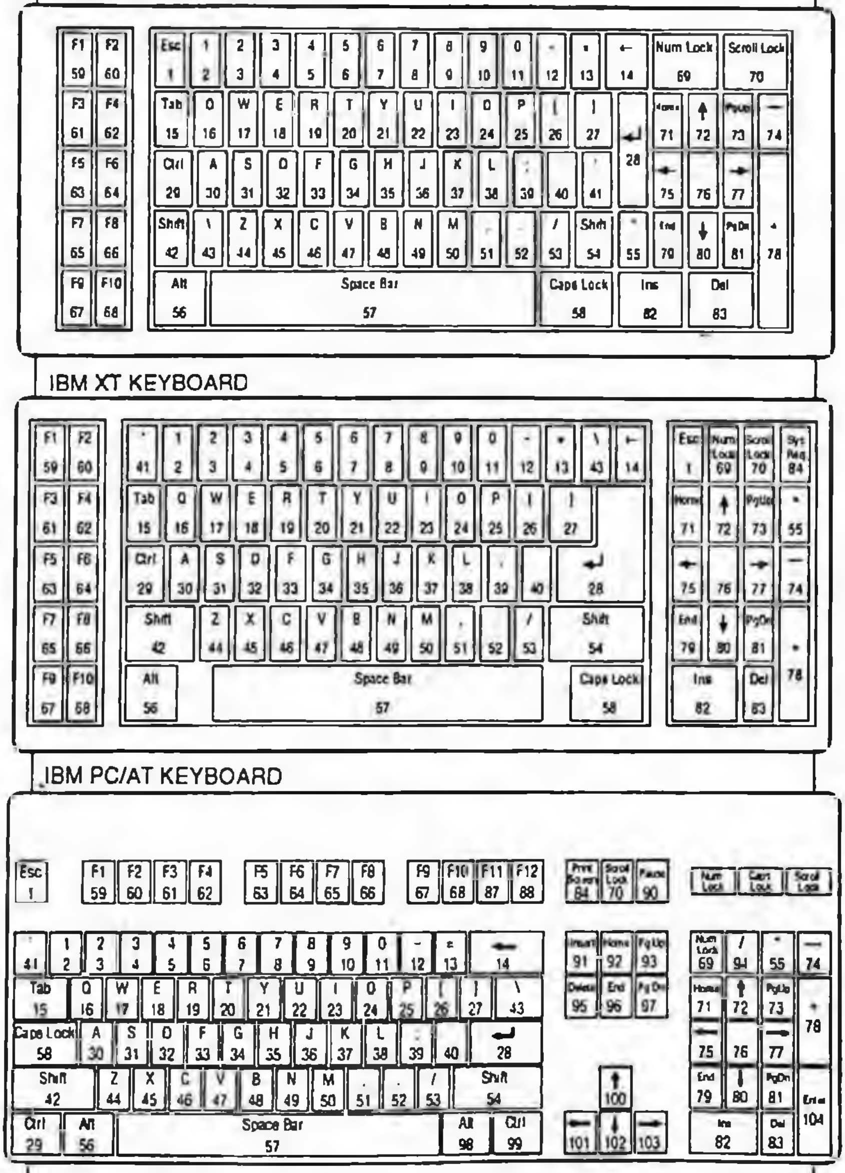
Before you learn key assignment, you should know how XyWrite takes keyboard input and puts the desired character on the screen. This is done with key codes in a keyboard file called XY4.KBD, which is contained on the original XyWrite disk.

The figure on the facing page shows how each key is numbered. For example, when you press key number 30, the keyboard file translates code 30 to mean the letter "A."

You use the LOAD command to load a new keyboard into XyWrite. It is a feature of XyWrite that you can load a keyboard file any time that XyWrite is running.

KEYBOARD DIAGRAM

STANDARD IBM KEYBOARD



IBM XT KEYBOARD

IBM PC/AT KEYBOARD

How to Modify Your Keyboard File. This section describes how a keyboard file is constructed. The XY4.KBD file is organized into five parts:

1. Comments	Example: ;END OF TABLE
2. Number of Keys	Example: KEYS=105
3. Shifting Keys	Example: CTRL=29,99
4. Tables	Example: TABLE=CTRL
5. Key definitions	Example: 30=a

Each of these parts requires some explanation:

1. **Comments.** Every line that begins with a semicolon is a comment — it has no effect on key assignments.

2. **Number of Keys.** The first definition in the keyboard file is the KEYS statement. To accommodate the 83- and the 104-key IBM keyboards plus a mouse, the KEYS statement is:

   ```
   KEYS=105
   ```

3. **Shifting Keys.** The XY4.KBD keyboard file comes with seven shifting *keys* (two Alt, two Ctrl, two Shift, and one Caps Lock), which define four shifting *states*. Each shifting key can shift the keyboard to a new set of key definitions. In XY4.KBD, the shifting states are defined as follows:

   ```
   CTRL=29,99
   ALT=56,98
   SHIFT=42,54
   CAPS=58,T:C
   ```

   XyWrite can have up to six shifting states (using any names you want — no numbers allowed). That is, if you keep these four states, you can define two more (or you could delete these four and define six new states). You can use these keys alone (e.g., TABLE=CTRL) or in combination (e.g., TABLE=CTRL+ALT) to create up to 20 tables. For example, you could define Print Screen to be a shifting key.

   The order of these definitions designates their priority. In the above example, if you press both Ctrl and Caps Lock, the CTRL table prevails (because there is no CTRL+CAPS table).

Options for shifting key definitions.

,N	Numeric Lock
,T:*n*	Toggle

n is any letter you specify. When you include *n* with a shifting key definition, XyWrite displays the letter you specify in the upper right corner of the screen to indicate that a toggle key is *on*.

N (Numeric Lock) indicates that when you press that shifting key, the sense of the Numeric Lock key changes. The statement SHIFT=42,54,N means that if Numeric Lock is on, pressing either Shift key changes the number pad to a cursor pad.

T (Toggle) defines a key as a toggle shifting key — when you press the key, the keyboard table associated with that shifting key is enabled and stays active until you press the shifting key again. Caps Lock is the only toggle shifting key on the default XY4.KBD keyboard, but you can define up to three others. (Num Lock, Scroll Lock, and Automatic Uppercase do not count — they are not toggle *shifting* keys.) You can display any letter in the header when a toggle key is on. For example, CAPS=58,T:C causes the letter "C" to appear when you press Caps Lock. The "C" disappears when you press Caps Lock again.

EXAMPLE: Let's say you need Greek letters to write technical text. You first create TABLE=PRTSCR, assigning Greek letters to keys. You next make PrintScreen a shifting key: PRTSCR=84,T would enable you to switch the Greek keyboard on and off. PRTSCR=84,T:G would also cause the letter G to appear on the command line while the PRTSCR table is in effect.

4. **Tables.** The keyboard file XY4.KBD is made up of ten tables. (You can have up to 20 tables.) Each table can give you an entire set of keys. The ten tables are:

```
TABLE=
TABLE=CAPS
TABLE=SHIFT
TABLE=SHIFT+CAPS
TABLE=CTRL
TABLE=ALT
TABLE=CTRL+SHIFT
TABLE=ALT+SHIFT
TABLE=CTRL+ALT
TABLE=CTRL+ALT+SHIFT
```

5. **Key Definitions.** Following the statement TABLE= you will find the key definitions.

Format: *#=f1,f2,f3,...*

is the key code — a number from 1 to 120
f1,f2,f3,... are function calls or characters

Examples:

30=a	The letter "a" is assigned to key number 30.
72=CU	The function CU (Cursor Up) is assigned to key 72.
68=GT,SH	The two functions GT, SH (Go to Text, Show Help) are assigned to key 68.
31=BC,s,a,v,e,XC	The command SAVE is executed by the series of keystrokes BC (Blank the Command Line), spelling SAVE, and XC (Execute). These keystrokes are all assigned to key 31.

Notice in this last example that when you enter text, each character is separated by a comma. Punctuation and spaces must also be separated by commas.

(cont'd)

Modifying Keyboard Files

ACTION

Reassigning a Key

To demonstrate how to reassign a key, we will show how to change the [Ctrl] [Shift] [↑] key combination to move the cursor up five lines at a time.

1. Call the keyboard file to the screen:

   Type: [F5]ca xy4.kbd[↵]

2. Use the SEARCH command to find the table for the [Ctrl] [Shift] keys.

   Type: [F5]se /table=ctrl+shift/[↵]

3. Find the code 72, which represents the [↑] (cursor up) key.

   Type: [F5]se /72/[↵]

   Result: The cursor stops on the line reading:

   72=GH←

4. Change this line to read:

   72=CU,CU,CU,CU,CU←

   CU is a *function call*—it means Cursor Up. (Each time it is executed, it moves the cursor up one line.)

5. Store the keyboard file.

   Type: [F5]store[↵]

6. Load the altered keyboard file into memory.

   Type: [F5]load xy4.kbd[↵]

To test the new key assignment, move the cursor several lines down from the top of any open file. Now, while holding down [Ctrl] and [Shift], press the [↑] (cursor up) key. The cursor moves up five lines. With a similar procedure, it is easy to create a [Ctrl] [↓] (cursor down) function that will move the cursor down five lines.

ACTION

Creating a Command Key

You might find it helpful to display a list of files in a specific directory by pressing a single keystroke. Let's see how you can create a command key to display the \XY4 directory by pressing [Ctrl]X.

1. Type: [F5]ca xy4.kbd[↵]

2. Notice that the key code for X is 44. Search for the CTRL table.

   Type: [F5]se /table=ctrl/[↵]

   Result: The cursor should now be positioned at the top of the CTRL table.

3. Find code 44:

   Type: [F5]se /44/[↵]

4. Change the line to read:

   ```
   44=BC,d,i,r,  ,\,x,y,4,XC←
   ```

5. Type: [F5]store[↵]

6. Type: [F5]load xy4.kbd[↵]

Analysis. Before trying out this new command, let's see what it does step-by-step:

BC clears the command line (the same as key [F5]).
The letters dir \xy4 are typed on the command line.
XC executes the DIR command (the same as key [F9]).

To try out this new command, press [Ctrl]X — this should display a directory of \XY4.

ACTION

Loading a Keyboard File

To load a keyboard file into memory, enter the LOAD command along with the name of the keyboard file you want to load. For example:

Type: [F5]load ibm1.kbd[↵]

Result: Keyboard file IBM1.KBD is now in effect.

NOTE #1 **Executing Function Calls as Commands.** The FUNC command enables you to execute any of the two-letter function calls from the command line. The function call is executed as though it were assigned to a key and you pressed that key. This is a way to execute seldom-used functions.

To execute a function call from the command line, enter FUNC followed by the function call you want to execute. For example, to execute the function NM (No Marker):

Type: F5func nm↵

In this case, the format triangles and carriage-return arrows normally displayed on-screen are hidden from your view (although they remain embedded in the file).

Function calls that perform text-related functions require the cursor to be in the text when you execute them.

NOTE #2 **Unassigned Keys.** When using the keyboard file to assign new functions to keys, avoid using a key that already has something assigned to it. For example, in XY4.KBD the Alt key is used primarily to activate the menus; most of the Ctrl keys are used for keyboard shortcuts.

NOTE #3 **Keep Old Lines as Comments.** When you modify a line in the keyboard file, it's often a good idea to keep the old line by making it a comment. This way you can remember what the key used to do in case you want to revert it to its previous function. Put one or more semicolons (;) in front of the old line. For example, if you were to change key 72 from CU to LU:

```
;;;72=CU
72=LU
```

NOTE #4 **Multiple Character Assignment.** Whenever you use more than one character in a key assignment, you must start the assignment with a function call. If no other function call is appropriate, use NO (No Operation). For example, the assignment 74=A,n,d,r,e,w would not work, whereas 74=NO,A,n,d,r,e,w would work. (If a key assignment does not begin with a function call, only the first character is read.)

NOTE #5 **Load Requirement.** It is not *mandatory* that you load a keyboard file, unless you want a customized keyboard. XyWrite comes with its own keyboard file built in. You have to load a keyboard file only if you want to *modify* the keyboard from its standard settings.

NOTE #6 **Startup.** In order to use a keyboard file you have altered, you must load it each time you start XyWrite. You can have your keyboard file automatically loaded at startup by adding the LOAD command to your STARTUP.INT file. Refer to "Startup File" elsewhere in this manual for information on modifying STARTUP.INT.

Function Calls

PURPOSE

Function calls are two-letter instructions that represent basic keyboard actions. Function calls are assigned to keys in the keyboard file. For example, 77=CR means that Cursor Right is assigned to the cursor right-arrow key (key #77). Similarly, DL means select a line of text; CP means copy the selected text; and M2 means set bold mode.

There are two ways you can execute these function calls: (1) in the keyboard file and (2) with the FUNC command on the command line. These are both described in the previous section "Modifying Keyboard Files." Function calls also appear in program files — for example, **BC** appears in the STARTUP.INT file.

To execute several function calls in sequence with one keystroke, assign them to a key, separating them with commas. For example, 77=CR,CD would move the cursor right and then down.

NOTE #1

User Programming. You can use all of these function calls in programs except TS (Toggle Scroll Lock). TS cannot be entered into a program file, since you use it to begin and end the recording of a program. If you want to enter a function call into a program that is not assigned to a key, use the PFUNC command.

NOTE #2

Function Calls with Arguments. Several function calls (TM, MT, MN, JM, JH) take arguments. If you are using the function call in a keyboard file, be sure to put commas between each character in the argument; if you are using the function call in a program, omit the commas. For example:

JH,(,b,a,s,i,c,s,) in a keyboard file
JH (basics) in a program file

NOTE #3

Redefining Function Calls. You can use XyWrite's command override feature to redefine that functions associated with a function call. Refer to "Custom Commands" elsewhere in this manual for more information.

FUNCTION CALLS BY CATEGORY

Aligning Text

FC	Force center
FL	Flush left
FR	Flush right

Character Modes (see Note #3)

M0	Mode 0, prevailing
M1	Mode 1, normal
M2	Mode 2, bold
M3	Mode 3, underline
M4	Mode 4, reverse
M5	Mode 5, bold underline
M6	Mode 6, bold reverse
M7	Mode 7, superscript
M8	Mode 8, subscript
M9	Mode 9, italic
MX	Mode 0, prevailing
MZ	Bold italic mode

Columns

CF	Column functions
EC	End of cell
ED	Entire row define
EE	Erase entry
HC	Home of cell
MC	Mark cell
TE	Table entry
TL	Table column left
TM	Table move
TR	Table column right

Command Line

BC	Blank the command line
CC	Change cursor
CH	Clear header
GH	Go to header
GT	Go to text area
S+	Command stack up
S-	Command stack down
XC	Execute

Copying and Moving Text

CP	Copy selected text
MV	Move selected text

Counters/Numbering

C0	Counters (C0 to C9)

Cursor Movement in Text

BF	Bottom of file
BS	Bottom of screen
CD	Cursor down
CL	Cursor left
CR	Cursor right
CU	Cursor up
DB	Defined beginning
DE	Defined End
EL	Express left
ER	Express right
HM	Home
JC	Jump over commands
LB	Line begin
LD	Linear cursor down
LE	Line end
LL	Cursor linear left
LR	Cursor linear right
LU	Cursor linear up
MD	Move down
MU	Move up
NC	Next character
NF	Next formatted page
NL	Next line
NP	Next paragraph
NS	Next sentence
NT	Next tab
NW	Next word
PC	Previous character
PF	Previous formatted page
PL	Previous line
PP	Previous paragraph
PS	Previous sentence
PT	Previous tab
PU	Page up
PW	Previous word
TF	Top of file
WH	Home

Function Calls (cont'd)

Deleting Text

BD	Backspace delete
DN	Delete selection without saving
NU	No undelete
RB	Rubout previous word
RC	Rubout character
RD	Rubout defined block
RE	Rubout to end of line
RL	Rubout line
RP	Rubout paragraph
RS	Rubout sentence
RW	Rubout word
UD	Undelete
UN	Paste from clipboard
UP	Unpad spaces

Draft/Expanded/Graphic Views

CM	Change view Draft/Expanded
HG	Hide graphics
MK	No marker mode
NM	No markers
NR	No ruler
SP	Formatted view
TG	Expanded
TP	Toggle page Formatted/Expanded
WG	Draft view
WZ	Graphic view
XP	Expanded view

Foreign/Overstrike Characters (Note #5)

S1	Acute accent
S2	Grave accent
S3	Umlaut
S4	Circumflex
S5	O accent
S6	Tilde
S7	Underline
XX	Define floating accent

Math

DT	Display total
MT	Math function
SM	Sum
SU	Subtract

Menu/Help

AK	Accelerator key
B4	Dialog box
HF	Help frame
HL	Help
H@	Help screens
JH	Jump to help
JM	Jump to menus
OP	Old prompt
QH	Quick help
QP	Quick print
SH	Show help
ST	Show triangle help
XH	Cancel help
$0	Call Help (S0 to S9)
$A	Call Help (SA to SZ)

Non-Keyboard Characters (see Note #4)

R0	Non-keyboard character (R0 to R9)

Programming

<<	Search start command
>>	Search end command
RX	Execute recorded keystrokes
BX	Blind execute
Q2	Execute Too

Redlining

LT	Login toggle
RO	Redlining on/off

Search (see Note #6)

FD	Find difference
FM	Find match
IT	Insert tab
NN	Generic wild card
WA	Wild alpha
WC	Wild carriage return
WL	Wild letter
WN	Wild number
WS	Wild separator
WW	Wild string
WX	Wild any character

Selecting Blocks of Text

DC	Define column
DD	Delete selected block
DF	Define block
DL	Define line
DM	Define modify
DP	Define paragraph
DS	Define sentence
DW	Define word
DZ	Define end
NB	Non-breakable block
XD	Cancel define
YD	Cancel define (variation)

Spelling/Thesaurus

AC	Auto-check
AR	Expand abbreviation
AZ	Auto-replace toggle
FX	Fix spelling
Q1	Dialog box functions (Q1 to Q8)
SC	Spelling auto-check
SO	Spell one word
SY	Synonyms

System

BK	Break
CO	Comma
DO	Display on
DX	Display off
EF	Edit marker
EN	Edit next file
ES	Escape
ET	Elapsed time
EX	Exit
FF	Force fill
IB	IBM mode
KF	Keyboard flip
MN	Menu
MS	Mouse
NI	Not IBM-sensitive
NO	No operation
PR	Print screen
RV	Review
SA	Save
SF	Store all files
SL	Save all files
XS	Scoping markers
ZT	Zero time

Text Macros

AD	Append define to text macro
SG	Get text macro
SK	Show text macro key
SV	Save define
@0	Text macro (@0 to @9)
@A	Text macro (@A to @Z)
&0	Text macro (&0 to &9)
&A	Text macro (&A to &Z)

Toggle Keys

BT	Borders toggle
CI	Clear insert mode
CN	Clear numeric lock
CS	Clear scroll lock
MI	Momentary insert
NK	Toggle numeric lock
RK	Record keystrokes
SI	Set insert mode
SN	Set numeric lock
SS	Set scroll lock
TI	Toggle insert mode
TN	Toggle numeric lock
TO	Toggle overstrike
TS	Toggle scroll lock
TW	Toggle word

Transposing Text (see Note 9)

FX	Transpose text units
XN	Transpose text units

Windows

AS	Alternate screen
CB	Cycle backwards
NX	Next window
SW	Show window menu
#1	Window (#1 to #9)

FUNCTION CALLS ALPHABETICALLY

The keys listed in the right-hand column are those provided in the original XY4.KBD file. You can change these assignments.

Function Call		Key
<<	**Start command** – Indicate a single left double-angle bracket in a program as part of a search executed from the command line.	(none)
>>	**End command** – Indicate a single right double-angle bracket in a program as part of a search executed from the command line.	(none)
#1	**Window No. 1** – Move the cursor to window no. 1.	Ctrl Shift 1
#2	**Window No. 2** – Move the cursor to window no. 2.	Ctrl Shift 2
#3	**Window No. 3** – Move the cursor to window no. 3.	Ctrl Shift 3
#4	**Window No. 4** – Move the cursor to window no. 4.	Ctrl Shift 4
#5	**Window No. 5** – Move the cursor to window no. 5.	Ctrl Shift 5
#6	**Window No. 6** – Move the cursor to window no. 6.	Ctrl Shift 6
#7	**Window No. 7** – Move the cursor to window no. 7.	Ctrl Shift 7
#8	**Window No. 8** – Move the cursor to window no. 8.	Ctrl Shift 8
#9	**Window No. 9** – Move the cursor to window no. 9.	Ctrl Shift 9
$0 to	**Call Help $0** – Call the help frame with keyword $0.	(none)
$9	**Call Help $9** – Call the help frame with keyword $9.	(none)
$A to	**Call Help $A** – Call the help frame with keyword $A.	(none)
$Z	**Call Help $Z** – Call the help frame with keyword $Z.	(none)
@0 to	**Text Macro Key 0** – Place contents of text macro 0 at the cursor location (or run program on text macro 0). (See Note #2.)	(none)
@9	**Text Macro Key 9** – Place contents of text macro 9 at the cursor location (or run program on text macro 9). (See Note #2.)	(none)
@A to	**Text Macro Key A** – Place contents of text macro A at the cursor location (or run program on text macro A). (See Note #2.)	(none)
@Z	**Text Macro Key Z** – Place contents of text macro Z at the cursor location (or run program on text macro Z). (See Note #2.)	(none)
&0 to	**Text Macro &0** – Run program assigned with LDPM filename,&0.	(none)
&9	**Text Macro &9** – Run program assigned with LDPM filename,&9.	(none)
&A to	**Text Macro &A** – Run program assigned with LDPM filename,&A.	(none)
&Z	**Text Macro &Z** – Run program assigned with LDPM filename,&Z.	(none)

Function Call		Key
AC	**Auto-Check** – Turn Auto-Check on and off.	Alt F7
AD	**Append Define to Text Macro** – Add the currently selected block of text to the end of the text in the specified text macro.	(none)
AK	**Accelerator Key** – Move to a specific item in a dialog box when you press the accelerator key associated with the item.	Alt A to Alt Z Alt 0 to Alt 9
AR	**Expand Abbreviation** – Execute Expand Abbreviation.	Ctrl
AZ	**Auto-Replace** – Turn Auto-Replace on and off.	Ctrl Shift A
AS	**Alternate Screen** – Move cursor between the two windows last displayed.	Alt F6
B4	**Dialog Box** – Display the dialog box previous to the last one displayed.	(none)
BC	**Blank the Command Line** – Clear the command line and move the cursor next to the start of the command line.	F5
BD	**Backspace Delete** – Delete the previous character.	Backspace
BF	**Bottom of File** – Move cursor to the bottom of the file.	Ctrl End
BK	**Break** – Stop any command currently in progress (such as delete or search); stop any user program that is running.	Ctrl Break
BS	**Bottom of Screen** – Move cursor to bottom of screen.	Alt End
BT	**Border Toggle** – Toggle window borders on and off.	(none)
BX	**Blind Execute** – Execute a command within a program without moving the cursor to the command line. Parentheses are required around the command and any arguments. For example: BX (ca test)	(none)
C0	**Counter 0** – Insert the formatting command C0 in the text.	(none)
to		
C9	**Counter 9** – Insert the formatting command C9 in the text.	(none)
CB	**Cycle Backwards** – Move through windows in the reverse order in which they were opened.	Shift F6
CC	**Change Cursor** – Move cursor between the command line and text.	Ctrl F10, Shift F5, Esc, 5 (on numpad)
CD	**Cursor Down** – Move cursor down one line.	↓
CF	**Column Functions** – Perform function indicated (see Note #10).	(none)
CH	**Clear Header** – Delete the text on the command line without moving the cursor.	(none)
CI	**Clear Insert Mode** – Switch to Overstrike mode (from Insert).	(none)
CL	**Cursor Left** – Move left one space; wrap to previous line.	←
CM	**Change View Draft/Expanded** – Switch between draft and expanded views.	(none)
CN	**Clear Numeric Lock** – Turn off the Numeric Lock.	(none)

Function Call		Key
CO	**Comma** – Insert a comma into the text. CO is used in a keyboard file in place of the punctuation mark (,) for commands that contain a comma (such as IP 0,5). CO is needed since the comma mark (,) is used as a separator. Example: I,P,0,CO,5,	(none)
CP	**Copy Defined Block of Text** – Copy the currently selected block of text to the cursor position.	Ctrl C
CR	**Cursor Right** – Move right one character; wrap to next line.	→
CS	**Clear Scroll Lock** – Turn off Program mode.	(none)
CU	**Cursor Up** – Move cursor up one line.	↑
DB	**Defined Beginning** – Move the cursor to the beginning of the selected block.	(none)
DC	**Define Column** – Begin selecting a column of text.	Alt F3
DD	**Delete Selected Block** – End selecting a block if one is in progress and delete the block. If nothing is selected, it deletes the character the cursor is on.	(none)
DE	**Defined End** – Move cursor to end of the selected block.	(none)
DF	**Define Block** – Begin or end selecting a block of any size.	F3
DL	**Define Line** – Select the line of text the cursor is on.	(none)
DM	**Define Modify** – Extend a block of selected text to the cursor position.	Shift F3
DN	**Delete Selection.** – Delete selected text without saving it on the delete stack.	(none)
DO	**Display On** – Turn on the display (complement of DX).	(none)
DP	**Define Paragraph** – Select paragraph the cursor is on.	F4
DS	**Define Sentence** – Select the sentence the cursor is on.	Shift F4
DT	**Display Total** – Dump accumulated sum into the text at the cursor position.	Alt =
DW	**Define Word** – Select the word the cursor is on.	Ctrl Shift →
DX	**Display Off** – Freeze the display (complement of DO).	(none)
DZ	**Define End** – End selecting a block if one is in progress.	Shift + cursor keys
EA	**Edit Second Argument** – Opens command window for editing text only.	(none)
EC	**End of Cell** – Move the cursor to the bottom of the current cell.	(none)
ED	**Entire Row Define** – Select the current row of cells in the table.	(none)
EE	**Erase Entry** – Delete a row of entries in the table.	Shift – (on numpad)
EF	**Edit Marker** – Open a command marker (e.g., header or footnote) so that you can edit it without switching to expanded view.	Shift F1
EL	**Express Left** – Move cursor to far left of line and then to left end of previous lines.	Home
EN	**Edit Next File** – Open the next file that matches a global filename specification.	Ctrl Alt Shift N
ER	**Express Right** – Move cursor to far right of line and then to start and end of subsequent lines.	End
ES	**Escape** – Remove the currently displayed menu or Help screen, release selected text, or close the currently displayed command window.	Esc

Function Call		Key
ET	**Elapsed Time** – Record the amount of time elapsed since you issued the ZT (Zero Time) function call and insert it in the text. You can use the ET and the ZT function calls as a stopwatch to keep track of how long it takes to complete a task.	(none)
EX	**Exit** – Quit XyWrite, return to DOS (after prompting for verification).	Ctrl Alt Del
FC	**Force Center** – Insert FC (Force Center) command to center text between the left and right margins.	(none)
FD	**Find Difference** – Compare the two files in the current and previous windows — stop where files do *not* match.	Ctrl -
FF	**Force Fill** – Force the display to refresh.	(none)
FL	**Flush Left** – Insert FL (Flush Left) command.	(none)
FM	**Find Match** – Compare two files in the current and previous windows — stop where files match.	Ctrl +
FR	**Flush Right** – Insert FR (Flush Right) command.	(none)
FS	**Fix Spelling** – Return cursor to last misspelled word and display the spelling menu.	Ctrl E
FX	**Transpose Text** – Transpose indicated unit of text (see Note #9)	-
GH	**Go to Header** – Move cursor from the text to the previous cursor position on the command line without clearing the command line.	Ctrl Shift ↑
GT	**Go to Text Area** – Move cursor from the command line to the previous cursor position in the text area.	Ctrl Shift ↓
H@	**Help Screens** – Display Help menu or context-sensitive help.	F1
HC	**Home of Cell** – Move the cursor to the top of the current entry in the table.	(none)
HF	**Help Frame** – Display Help frame with the same % keyword as assigned to the key in the keyboard file.	(none)
HG	**Hide Graphics** – Display a border around the graphic area without displaying the graphic itself.	(none)
HL	**Help** – Display help frame related to first word on command line.	(none)
HM	**Home** – Move cursor to the top of the screen.	Alt Home
IB	**IBM Mode** – Clear the XyWrite keyboard buffer.	(none)
IT	**Insert Tab** – Insert a tab on the command line.	Ctrl Shift Tab
JC	**Jump over Commands** – If the cursor is on a command marker, move the cursor to the right until it is no longer on a command.	(none)
JH	**Jump to Help** – Display the Help frame that resides in the Help file with specified keyword. Parentheses are required around the keyword. For example: JH,(,b,a,s,i,c,s,)	(several keys)
JM	**Jump to Menu** – Display menu frame that resides in the Menu file with the specified keyword. Parentheses are required around the keyword, and commas must separate the letters of the keyword. For example: JM,(,o,p,t,i,o,n,1,)	(several keys)

Function Call		Key
KD	**Kill Dialog Box** – Remove current dialog box from screen.	(none)
KF	**Keyboard Flip** – Redirect the keyboard to XyWrite to prevent keystrokes being transmitted while a TSR application program is being accessed.	Ctrl Esc
LB	**Line Begin** – Move cursor to far left of line and no further.	(none)
LD	**Cursor Linear Down** – Move cursor directly down one line.	(none)
LE	**Line End** – Move cursor to far right of line and no further.	(none)
LL	**Cursor Linear Left** – Move cursor left one character; do not wrap to previous line.	(none)
LR	**Cursor Linear Right** – Move cursor right one character; allowable to move past the carriage return.	(none)
LT	**Login Toggle** – Suppress text display of captured redlining login information.	(none)
LU	**Cursor Linear Up** – Move cursor directly up one line.	(none)
M0	**Mode 0, Prevailing** – Text that you type is entered in the mode that exists at the cursor location.	Ctrl 0
M1	**Select Normal Mode** – Text that you type is entered in the normal mode (not highlighted).	Ctrl 1
M2	**Select Bold Mode** – Text that you type is entered in bold mode.	Ctrl 2
M3	**Select Underline Mode** – Text that you type is entered in underline mode.	Ctrl 3
M4	**Select Reverse Mode** – Text that you type is entered in reverse mode.	Ctrl 9
M5	**Select Bold Underline Mode** – Text that you type is entered in bold underline mode.	Ctrl 5
M6	**Select Bold Reverse Mode** – Text that you type is entered in bold reverse mode.	(none)
M7	**Select Superscript Mode** – Text that you type is entered in superscript mode.	Ctrl 7
M8	**Select Subscript Mode** – Text that you type is entered in subscript mode.	Ctrl 8
M9	**Select Italic Mode** – Text that you type is entered in italic mode.	Ctrl 4
MC	**Mark Cell** – Select the cell where the cursor is located in the table.	(none)
MD	**Move Down** – Scroll text and cursor down one line.	Ctrl PgUp
MI	**Momentary Insert** – Switch from Overstrike to Insert for a single insert. Enter this function call at the exact location that you want to make the insert.	(none)
MK	**No Marker Mode** – Remove embedded command and line ending markers from the displayed text.	Shift F11, Alt Shift F1
MN	**Menu** – Display information about the currently loaded values (see Note #7).	(none)

Function Call		Key
MS	**Mouse** – Designate that a mouse is installed. To use the mouse with XyWrite, the MS function call must be assigned to key 105 in the unshift, shift, caps, and shift+caps tables of the keyboard file.	
MT	**Math Function** – Multiply (*) or divide (/) accumulated sum by the selected number. (See Note #8.)	Alt ., Alt / (on numpad)
MU	**Move Up** – Scroll text and cursor up one line.	Ctrl PgDn
MV	**Move Defined Block of Text** – Move the currently selected block of text to the cursor position.	Ctrl M
MX	**Prevailing Mode.** Same as M0, but does not get inserted in programs.	(none)
MZ	**Select Bold Italic Mode** – Text that you type is entered in bold italic.	Ctrl 6
NB	**Non-Breakable Block** – Designate the selected block of text as unbreakable by inserting an NB (Non-Breakable) command at the beginning and a BB (Breakable Block) command at the end.	(none)
NC	**Next Character** – Move cursor to the next character.	(none)
NF	**Next Formatted Page** – Move cursor to first line of next printed page.	Alt PgDn
NI	**Not IBM-Sensitive Key** – Prevent a key from being passed to DOS. Use it at the beginning of key assignments for standard DOS key functions, such as PrtScr.	(several)
NK	**Toggle Numeric Lock** – Change the state of the Numeric Lock key.	(none)
NL	**Next Line** – Move cursor to start of next line.	Alt ↵
NM	**No Markers** – Hide the format triangles, carriage return arrows, and Page-Depth numbers. NM also switches the view to draft if it was in expanded.	(none)
NN	**Generic Wild Card** – Establish the key as a wild card. The character after the comma is the character XyWrite displays in reverse to indicate the wild card. The only valid characters used with NN are:	
	0-9 Interpreted as number of times string appears.	Alt Shift 0 to Alt Shift 9
	– Interprets string as anything but the next single character.	Alt Shift –
	O Indicates more than one string.	Alt Shift O
	◄ Interpreted as a carriage return.	Alt Shift R
	↨ Interpreted as a line feed.	Alt Shift F
	. Interpreted as a sentence separator.	Alt Shift .
NO	**No Operation** – XyWrite requires that keys with multiple character assignments start with a function call. Use NO in the keyboard file when no other function call is appropriate.	(several)
NP	**Next Paragraph** – Move cursor to start of next paragraph.	Alt ↓
NR	**Next Ruler** – Toggle the third line in the header between the tab ruler, a straight line, and an extra line of text.	(none)
NS	**Next Sentence** – Move cursor to start of next sentence.	(none)
NT	**Next Tab** – Move cursor to the next tab without moving the text.	Ctrl Tab
NU	**No Undelete** – Delete selected text without saving it for possible later undelete.	Shift Del

Function Call		Key
NW	**Next Word** – Move cursor to start of next word.	Ctrl →
NX	**Next Window** – Move the cursor successively through all open windows (in numerical order).	F6
OP	**Old Prompt** – Switch from your document to the previously accessed Menu/Help frame.	Shift F10
PC	**Previous Character** – Move cursor to the previous character.	(none)
PD	**Page Down (Next Screen)** – Scroll down one screen.	PgDn
PF	**Previous Formatted Page** – Move cursor to first line of previous printed page.	Alt PgUp
PL	**Previous Line** – Move cursor to start of previous line.	(none)
PP	**Previous Paragraph** – Move to start of previous paragraph.	Alt ↑
PR	**Print Screen** – Print the contents of the full screen (all 25 lines, including header).	(none)
PS	**Previous Sentence** – Move to start of previous sentence.	(none)
PT	**Previous Tab** – Move cursor to previous tab position.	Shift Tab
PU	**Page Up (Previous Screen)** – Scroll up one screen.	PgUp
PW	**Previous Word** – Move cursor to previous word.	Ctrl ←
Q1-Q8	**Spelling dialog box functions.**	F1 - F6, Esc
Q2	**Execute** – Execute the command entered with the BX function call.	(none)
QH	**Quick Help** – Display the Menu/Help screen associated with the frame in which the cursor is located in a displayed Menu/Help file.	Ctrl Alt Shift Q
QP	**Quick Print** – Print the menu screen in which the cursor is located in a displayed menu file.	(none)
R0	**Character 0** – Enter the single digit 0.	Ctrl Alt 0 (on numpad)
R1	**Character 1** – Enter the single digit 1.	Ctrl Alt 1 (on numpad)
R2	**Character 2** – Enter the single digit 2.	Ctrl Alt 2 (on numpad)
R3	**Character 3** – Enter the single digit 3.	Ctrl Alt 3 (on numpad)
R4	**Character 4** – Enter the single digit 4.	Ctrl Alt 4 (on numpad)
R5	**Character 5** – Enter the single digit 5.	Ctrl Alt 5 (on numpad)
R6	**Character 6** – Enter the single digit 6.	Ctrl Alt 6 (on numpad)
R7	**Character 7** – Enter the single digit 7.	Ctrl Alt 7 (on numpad)
R8	**Character 8** – Enter the single digit 8.	Ctrl Alt 8 (on numpad)
R9	**Character 9** – Enter the single digit 9.	Ctrl Alt 9 (on numpad)
RB	**Rubout Previous Word** – Delete the word before the word the cursor is on.	Ctrl Backspace
RC	**Rubout Character** – Delete character at point of cursor.	Del
RD	**Rubout Defined Block** – Delete selected block of text.	F12, Alt F2
RE	**Rubout to End of Line** – Delete text from cursor to the end of the line.	Alt Del
RK	**Record Keystrokes Mode** – Toggle Record Keystrokes mode on and off.	Ctrl K
RL	**Rubout Line** – Delete the line the cursor is on.	(none)
RO	**Redlining On/Off** – Turn redlining on and off.	(none)

Function Call		Key
RP	**Rubout Paragraph** – Delete the paragraph the cursor is located in.	Ctrl F12, Ctrl Alt F2
RS	**Rubout Sentence** – Delete the sentence the cursor is located in.	(none)
RV	**Review** – Preview the document that is currently displayed (same as PRINTS).	(none)
RW	**Rubout Word** – Delete the word the cursor is on.	Ctrl Delete
RX	**Execute Recorded Keystrokes** – Execute the last set of keystrokes you recorded.	Shift F9
S1	**Acute Accent**	(none)
S2	**Grave Accent**	(none)
S3	**Umlaut**	(none)
S4	**Circumflex**	(none)
S5	**O Accent**	(none)
S6	**Tilde**	(none)
S7	**Underline**	(none)
S+	**Stack Up** – Display the previous command in the command stack.	Ctrl ↑
S-	**Stack Down** – Display the next command in the command stack.	Ctrl ↓
SA	**Save** – Save the currently displayed file.	(none)
SC	**Spelling Auto-Check** – Turn Auto-Check on or off, depending on the argument you give the SC function call: SC,0 turns Auto-Check OFF; SC,1 turns it ON.	(none)
SF	**Store Files** – Store all open files in all windows.	(none)
SG	**Get Text Macro** – Output the contents of the text macro indicated by the next letter or number typed.	F2
SH	**Show Help** – Display the XyWrite action bar.	F10
SI	**Set Insert Mode** – Switch to insert mode (from Overstrike).	(none)
SK	**Show Text Macro Key** – Show contents of the text macro indicated by next letter or number typed.	(none)
SL	**Save Files** – Save all open files in all windows.	(none)
SM	**Sum** – Add the number the cursor is on to the total; if block is selected, sum all numbers in block.	Alt + (on numpad)
SN	**Set Numeric Lock** – Select the Numeric Lock state.	(none)
SO	**Spell One Word** – Check the spelling of a single word.	Ctrl F7
SP	**Show Page-Depth No.** – Switch to formatted view.	F8
SS	**Set Scroll Lock** – Turn on Program mode.	(none)
ST	**Show Triangle Help** – Displays help for the command marker under the cursor.	(none)
SU	**Subtract Value** – Subtract the number the cursor is located on from the total.	Alt – (on numpad)
SV	**Save Define** – Save the selected block of text to the text macro indicated by the next letter or number typed.	Shift F2
SW	**Show Window Menu** – Display the window menu.	Ctrl F6

Function Calls (cont'd)

Function Call		Key
SY	**Synonyms** – Display a list of synonyms.	Shift F7
TE	**Table Entry** – Create a new row of entries in the table.	Shift + (on numpad)
TF	**Top of File** – Move cursor to top of the file.	Ctrl Home
TG	**Expanded** – Turn on expanded view.	Ctrl F8
TI	**Toggle Insert Mode** – Switch between Insert and the active Overstrike mode and back.	Ins
TL	**Table Column Left** – Move the cursor one column to the left in the table.	Alt ←
TM	**Table Move** – Move the cursor in a column to the position indicated. by the next keystroke (see Note #1).	(none)
TN	**Toggle Numeric Lock** – Change the state of the Numeric Lock.	(none)
TO	**Toggle Overstrike** – Switch between Character Overstrike and Insert mode.	(none)
TP	**Toggle Formatted/Expanded** – Toggle between formatted and expanded view.	(none)
TR	**Table Column Right** – Move the cursor one column to the right in the table.	Alt →
TS	**Toggle Scroll Lock** – Switch in and out of program mode.	Scroll Lock
TW	**Toggle Word** – Switch between Insert mode and Word Overstrike mode.	Alt Ins
UD	**Undelete** – Restore the last text deleted.	F12, Alt F2
UN	**Undelete** – Paste copy from clipboard.	Shift Ins
UP	**Unpad Spaces** – Delete spaces between the cursor and the first character to its left.	(none)
WA	**Wild Alphanumeric** – Interpreted as any alphabetic or numeric character in a search.	Alt Shift A
WC	**Wild Carriage Return** – Interpreted as the ← character at the end of lines.	(none)
WG	**Draft View** – Switch text to draft view without Page-Depth indicators.	Alt F8
WH	**WordPerfect Home** – Emulate WordPerfect's implementation of the Home key (see Note #11).	(none)
WL	**Wild Letter** – Interpreted as any letter A-Z in a search.	Alt Shift L
WN	**Wild Number** – Interpreted as any number 0 through 9 in a search.	Alt Shift N
WS	**Wild Separator** – Interpreted as any word separator in a search.	Alt Shift S
WW	**Wild String** – Interpreted as any string from 1 to 80 characters in a search.	Alt Shift W
WX	**Wild Any Character** – Interpreted as any character in a search.	Alt Shift X
WZ	**Graphic View** – Switch text to graphic view.	Shift F8
XC	**Execute** – Execute the command that is currently on the command line.	F9

Function Call		Key
XD	**Cancel Define** - Release the selected block of text.	(none)
XH	**Clear Help** - Remove any currently displayed Menu or Help screens from view.	(several)
XM	**Express Middle** - Move cursor to the middle of the line.	(none)
XP	**Expanded View** - Switch text to expanded view.	(none)
XN	**Transpose Text** - Transpose the indicated text units (see Note #9).	(none)
XS	**Scoping markers** - Toggle the display of markers affected by scoping (scoping is turned on with the ES default setting).	(none)
XX	**Extra Accent** - Define a floating accent (see Note #5).	(none)
YD	**Cancel Define** (Variation) - Similar to XD — release selected text but don't close a footnote screen or other such screen. (Used in programming).	(none)
ZT	**Zero Time** - Reset the stopwatch function to zero and start the timer.	(none)

NOTE #1 Table Move. The TM function call moves the cursor to the position indicated by the next keystroke. The following options are available:

0 Move to next element in same column (stop at last element in column)
1 Move to previous element in same column (stop at first element in column)
2 Move to top element in current column
3 Move to bottom element in current column

NOTE #2 @0 - @9, @A - @Z. The @ ("at" sign) followed by a letter or a number is used to get text from the text macro key letter or number. For example, @A requests the text saved as "A" to be inserted at the position of the cursor.

NOTE #3 M0 - M9, MZ. The letter "M" followed by a number between 0 and 9 or the letter Z selects the character mode of text typed in at the keyboard. If there is a block selected, the character mode of the selected block is changed to the selected mode.

NOTE #4 R0 – R9. The letter "R" followed by a single digit (0 to 9) forms one digit of a XyWrite character. XyWrite characters not assigned to a key must be entered this way.

The first time R0 to R9 is encountered, the number following the R is accumulated as a digit of the decimal number. Subsequent keys add further digits to the code number being accumulated. (If a key is struck that is not one of these function calls, the code is cleared.) If the state of the keyboard changes (such as a shift key being released), the accumulated code is given to the program as complete.

As an example (in the XY4.KBD file), when Alt Ctrl is held down, numbers are assigned the R values. If you hold down the Alt and Ctrl keys and press the numbers 2,9,4, the Greek letter alpha (α) is entered in the text when you release the Alt and Ctrl keys.

NOTE #5 Accented Characters. The letter S followed by a number between 1 and 7 is used for entering foreign accented characters. When you use one of these calls, the indicated accent mark appears on the screen (at this stage, the accent mark may not display correctly). If the letter that is next pressed is a letter that accepts accents (for example, a, A, e, E, i, I).

The function call XX allows you to define additional characters as floating accents. The format of the XX function call is:

XX,#

where # is the character you want to define as a floating accent. You must also define the new floating accent in the AC (Accent) Table in the default file (see Appendix C for information about the AC Table).

NOTE #6 NN, WA, WC, WL, WN, WS, WW, WX. These function calls are used to indicate the wild card entries that are part of the Search string.

NOTE #7 **MN (Menu) Function Arguments.** The MN function has four arguments you can use to specify the list of choices you want XyWrite to display:

CD (Cartridge Directory) lists all the cartridges for the printer file you loaded. You can use the cursor keys to select one or more cartridges and press ↵ to load it.

FO (Fonts) lists all the available loaded internal, soft, and cartridge fonts.

MT (Match Type) lists the printer and screen fonts XyWrite will use at the cursor location in the document. Information displayed includes: the font you specified, the font XyWrite will use, the name of the screen font file, and the name of the screen font.

SZ (Size) lists point sizes available for the printer font you are using at the current cursor location. If you are using scalable fonts, XyWrite lists many of the sizes available, but allows you to enter a specific size.

The MN function call is used in XY4.MNU to display the four menus described above.

NOTE #8 **MT (Math) Functions.** The MT function call also accepts the plus (+) and minus (-) operators, although they are not mapped to keys.

NOTE #9 **Transpose Text.** There are two function calls that transpose text units: FX and XN. The FX function call allows you to map the transpose functions to a single key, while the XN function call lets you create a separate key for each transpose function. You press the key assigned to the FX function call in conjunction with a letter key (C, W, S, P) or number key (1-6) to transpose text as follows:

C or 1 If the cursor is on a character, transposes the current and previous characters. If the cursor is on a separator, transpose the two characters before the cursor.
W or 2 Transposes the current and previous words.
S or 3 Transposes the current and previous sentences.
P or 4 Transposes the current and previous paragraphs.
5 Transposes the current and previous characters.
6 Transposes the two characters that precede the cursor.

To assign a specific transpose function to a key, assign the function call XN, a comma, and a number 1-6 (as described above).

NOTE #10 **CF (Column Function).** The CF function call allows you to perform several editing functions on a table, depending on the argument that follows the function call:

1	Insert new column before current colmun
2	Insert new column after current column
3	Delete current column
4	Select current column
5	Select current table

NOTE #11 **WH (WordPerfect Home).** The WH function call works in conjunction with other function calls to emulate WordPerfect's use of the Home key. Specifically:

WH,CU	Moves cursor to top of screen
WH,-	Moves cursor to top of screen
WH,CD	Moves cursor to bottom of screen
WH,+	Moves cursor to bottom of screen
WH,CL	Moves cursor to beginning of line
WH,WH,CU	Moves cursor to top of document
WH,WH,CL	Moves cursor to beginning of line
WH,WH,CR	Moves cursor to end of line

If WH is followed by any other function, it is ignored.

Loading Information Into Memory

INTRO

Information gets loaded into memory in different ways. Every time you start XyWrite from DOS, the program is loaded into memory. Then, when XyWrite runs the program STARTUP.INT, the files that you have selected to be part of your standard system configuration are loaded into memory. When you display a file on screen, it is loaded in memory too.

Three commands load custom information into memory: LOAD, SETP, and CART. They can be used as part of the startup procedure, or entered from the command line after startup is complete.

There is no need to unload information from memory. XyWrite automatically releases sections of code that are not in use, and reloads them when you issue commands that require them.

CONTENTS

Load Customization Files LOAD

FORMAT C:XY4 LOAD *file1,file2,file3...*

file1, *file2*, *file3*, and so on are names of XyWrite customization files.

MENU Advanced Preferences Files...

PURPOSE

You use the LOAD command to load the information in one or more customization files into memory. You must load the files into memory before they take effect. Each XyWrite customization file starts with a special 4-character label that tells the LOAD command what type of file it is. The 11 different file types and the label associated with each are listed below.

• Default file	;PR;
• Dialog box file	;DG;
• Help file	;HL;
• Hyphenation file	;HY;
• Keyboard file	;KB;
• Menu file	;MN;
• Personal spelling dictionary	;SP;
• Printer file	;PR;
• Soft font file	;SF;
• Sort file	;SO;
• User interface file	;U2;

LOAD overwrites any information already in memory for the file type you are loading. For example, if you load a printer file called POST35N.PRN and then later load HPLJ-2P.PRN, XyWrite overwrites the printer-related information in memory from POST35N.PRN. *The exception to this rule is the default file.* If you load a new default file, only those defaults that are *redefined* in the new file are changed. For example, if the first default file you loaded specifies a value for OF and IP, and the second default file specifies a value for OF but not IP, then the original IP value remains in effect.

ACTION

Loading Customization Files

To load a keyboard file and a printer file into memory with one LOAD command:

1. Be sure the files you want to load start with the proper identifiers.

2. Issue the LOAD command.

   Type: F5 load custom.kbd,post35n.prn ↵

Result: Keyboard file CUSTOM.KBD and printer file POST35N.PRN are loaded into memory. Information in memory for a different keyboard or printer file is overwritten.

NOTE #1 **Multiple Spelling Dictionaries.** Because it is not unusual to want to use more than one personal spelling dictionary, XyWrite allows you to *append* the information in a spelling dictionary to the spelling information already in memory. To do this, put a plus sign before the name of the spelling dictionary you are appending. For example, to load PERS.SPL *and* BUS.SPL:

Type: F5load pers.spl+bus.spl↵

To later append the words in LEGAL.SPL:

Type: F5load +legal.spl↵

NOTE #2 **Memory Requirements.** XyWrite requires 384K of memory. If additional memory is available, it will use up to 640K. Because there are times when you don't need all of the XyWrite functionality that is in memory, the program automatically releases sections of code that are not in use. The released sections are automatically reloaded when you issue commands that require them.

NOTE #3 **STARTUP.INT.** If you use a standard set of customization files, you can put the LOAD command into STARTUP.INT. Then your standard files will be automatically loaded every time you start XyWrite.

FORMAT

C:\XY4 SETP *n*

n (optional) is the number of the port/printer file combination you want. If you omit the number, the Select Printer File dialog box is displayed.

MENU

File Printer Setup...

PURPOSE

The SETP command lets you review a list of port/printer file combinations that are available to you, tells you which one is currently active, and lets you select the one you want to use. When you select one of the combinations, XyWrite automatically loads the associated printer file.

The SETP command also runs an optional program file that is associated with the destination or printer file you have selected (see Note #2).

There are two ways you can select a port/printer file combination with the SETP command:

- By displaying the dialog box, then highlighting the print destination you want and activating the Load pushbutton
- By issuing the SETP command with the number of the port/printer file combination you want to activate

ACTION

Displaying a List of Available Port/Printer File Combinations
To display a list of available port/printer file combinations:

Type: F5 setp ↵

Result: The Select Printer File dialog box appears. Notice that one line is highlighted. The highlighted line is the port/printer file combination that XyWrite uses when you issue the PRINT command.

ACTION

Selecting a Printer
To change to another port/printer file combination, type the SETP command along with the line number of the combination you want. For example:

Type: F5 setp 2 ↵

Result: XyWrite directs printing to the port specified for the second line in the Select Printer File list box, and loads the printer file specified for that destination. XyWrite also executes the commands in NETWORK2.PRG, if it exists (see Note #2).

NOTE #1 **Print Destinations.** The ports listed in the SETP menu can be local printer ports (generally designated as 1 or 2) or network printer ports (generally designated as LPT1, LPT2, LPT3, COM1, COM2, etc.)

NOTE #2 **Assigning a Program to a Printer.** You can create a program file for one or more port/printer file combinations listed in the Select Printer File list box. These program files must be named NETWORK*n*.PRG, where *n* is a number from 1 to 9 that corresponds to the combination's position in the list box. Typically, these program files contain the commands required to download fonts to the printer, or to execute some network-specific spooling commands.

XyWrite automatically runs the NETWORK*n*.PRG files, which must be stored in a directory that is included in the path statement of your AUTOEXEC.BAT file.

NOTE #3 **LOAD vs SETP.** Both the LOAD and SETP commands can be used to load a printer file. However, the LOAD command does not direct XyWrite to a different port, nor does it update the list box in the Select Printer File dialog box. We recommend that you use SETP to load printer files if your configuration includes printers attached to different ports.

NOTE #4 **PP Table.** The destinations, printer files, and comments listed in the Select Printer File dialog box are controlled by the PP table in the default file and the ADDTBL command. See the descriptions of the ADDTBL command on the next and "Default Settings" elsewhere in this manual for more information.

NOTE #5 **STARTUP.INT.** If you usually use the same print destination, you can put the SETP command into STARTUP.INT. Then your standard printer information will be automatically loaded every time you start XyWrite.

FORMAT

[CMD] ADDTBL *port◌file.prn◌notes*

port is the printer port (see Note #1).
file.prn is the printer file associated with *port*
notes is the description of *file.prn*

The three fields are separated by tabs.

MENU

Not a menu item.

PURPOSE

ADDTBL lets you add to the list of printers displayed when you issue the SETP command or choose Printer Setup from the File menu without modifying the PP table in the default file. ADDTBL loads the information into memory and appends it to the end of the list of printers in the Select Printer File dialog box.

ADDTBL is most useful in a LAN environment that has network printers and local printers. You define network-level printers (those that are available to all users) in a PP (Pick Printer) table in SERVER.DFL. You define local printers with ADDTBL commands in *user*.LOG files. Both SERVER.DFL and *user*.LOG files are set up for you during installation. If you change printers after installation, modify the appropriate customization file.

ACTION

Adding a ADDTBL Command

To add a local printer for a network user:

1. Call the user's .LOG file. For example:

   Type: [F5]ca x:\xy4\userlogs\cit.log[↵]

2. Go to the end of the file and add the ADDTBL command. For example, to add a Panasonic KXP1124 dot matrix printer:

   Type: BX addtbl lpt1◌pan1124.prn◌Panasonic dot matrix printerQ2

3. Store the file.

Result: The next time user CIT logs on, the local printer will be added to the list of network printers displayed in the Select Printer File dialog box.

NOTE #1

Also See. Refer to "Default Settings" for information about the PP table and to the *LAN Administrator's Guide* for information about network customization files.

FORMAT C:XY4 CART *c1,c2,c3...*

c1,c2,c3 are the 1-, 2-, or 3-character abbreviations for the cartridges.

MENU File | Printer Setup...

PURPOSE CART loads font information about the specified cartridge fonts into memory. XyWrite provides support for over 40 Hewlett-Packard (or compatible) cartridges by storing font information in two binary font files: LJ.BIN, LJ3.BIN, and LJ4.BIN. LJ.BIN is used with all Hewlett-Packard LaserJet printers except the LaserJet III and LaserJet IV. LJ3.BIN is used with the Hewlett-Packard LaserJet III family; and LJ4.BIN is used with the Hewlett-Packard LaserJet IV.

XyWrite uses the information in the .BIN files to create the cartridge and typeface menus; more importantly, XyWrite uses this information when it determines line endings.

ACTION **Loading Cartridges**

To load cartridges A and GT:

1. Be sure the correct printer file is loaded.

2. Type: F5 cart a,gt ↵

Result: Font information for the A and GT (Global Text) cartridges is loaded into memory, overwriting font information for cartridges that were previously loaded.

NOTE #1 **Number of Cartridges.** The number of cartridges that can be loaded at any given time is dictated by the number of cartridges that the printer supports. For example, the LaserJet IIP only accepts one cartridge and the LaserJet II accepts two cartridges.

NOTE #2 **List of Cartridges.** To display a list of cartridges, load the LaserJet printer file that applies to the model of LaserJet you are using, and then activate the Select cartridge radio button from the Printer Setup dialog box. (Press F10, F, R to display the Printer Setup dialog box.)

NOTE #3 **Additional Cartridges.** The LJ.BIN, LJ3.BIN, and LJ4.BIN files support a wide range of HP LaserJet and compatible cartridges. However, new cartridges are introduced regularly. If you want to use a cartridge that is not currently supported, use the TFMADD program to add the cartridge's font information to LJ.BIN, LJ3.BIN, or LJ4.BIN. TFMADD is a utility that extracts font information from the Tagged Font Metric files which are supplied with the cartridges and appends it to the .BIN file you specify. For information on how to use TFMADD, refer to the README.PRN file supplied with XyWrite.

NOTE #4 **STARTUP.INT.** If you standardly use a particular set of cartridges, you can put the CART command into STARTUP.INT. Then the cartridge information will be automatically loaded every time you start XyWrite. The CART command must be issued after the printer file is loaded; that means you should put the CART command *after* the SETP command.

Menu, Dialog Box, and Help Files

INTRO

You can modify XY4.MNU, XY4.DLG, and XY4.HLP, the menu, dialog box, and help files delivered with XyWrite or you can construct your own menu, dialog box, or help file. This section describes the various components of XY4.MNU, XY4.DLG, and XY4.HLP and discusses customizing them by adding, deleting, or rearranging information.

CONTENTS

Appendix D, "Reserved XyWrite Elements," lists ID numbers, keywords, and macros reserved for use by XyWrite.

PURPOSE

A menu, dialog box, or help file is a text file that can be created or edited as you edit any other text file. However, the first line of the file must contain a specific 4-character label to tell the LOAD command what kind of file it is:

;MN; indicates a menu file

;DG; indicates a dialog box file

;HL; indicates a help file

You can include formatting commands on the second line of the file to control the way text is arranged when displayed on the screen.

After the formatting commands, menu, dialog box, and help files may have several comment lines. You can add as many comment lines as you want to describe such things as the contents of the file or any changes you've made to the file. (Refer to "Including Comments in a Menu, Dialog Box, or Help File" for a way to enter your notes.)

Following any comment lines, menu, dialog box, and help files consist of sections or frames that XyWrite uses to construct and operate each menu, dialog box, or help screen.

Menu, Dialog Box, and Help File Frames

INTRO

Each frame in a menu, dialog box, or help file designates the information XyWrite displays on a screen at a time. Menu frames may contain commands or programs the user can initiate from the screen.

In general, menu screens and dialog boxes enable the user to perform some task. There are several different frame types you can use depending on the way you want the menu screen to function. XY4.MNU and XY4.DLG contain the following frame types:

0	displays a menu bar
L	displays a pull-down or side menu
K	displays a dialog box
5	executes a routine
M	indicates a comment

Types 0, L, and K display information on the screen; Types 5 and M are invisible to the user. XY4.DLG also contains Types C and 6 frame types, which are specially dedicated to perform specific tasks and are not applicable to general uses.

In general, help screens provide information to the user. XY4.HLP contains the following frame types:

C	displays a scrollable help screen
9	provides access to detailed error messages

(XY4.HLP also includes some "routine only" Type 5 frames, which initialize some variables before moving to a help frame.)

Although menu, dialog box, and help frames have different uses, they have the same basic construction. Every frame starts with a label enclosed in a pair of double curly braces {{ }}. A frame begins at the label and ends at the next label (which signals the beginning of the next frame).

Following the label, frames may have one or more displayable lines. Displayable lines contain text that appears when the user accesses the help or menu screen.

Following the label or any displayable lines, frames may have one or more executable lines. Executable lines are not displayed on the screen. They contain commands or programs and their content depends on the function you want the frame to perform.

The various types of frames are described throughout this section based on their functions (such as displaying a menu bar and displaying a scrollable help screen).

Although frames are not listed in any particular order within a menu, dialog box, or help file, XyWrite can quickly find the topic on which the user requested information. Refer to "Loading Menu and Help Files" for more information on what happens to a menu, dialog box, or help file when it is loaded into memory.

Keyword Labels

The keyword label identifies the frame to XyWrite and allows you to link the various types of menu and help frames. Frame labels have the following format:

```
{{#,keyword1,keyword2,. . .}}←
```

# indicates the frame type. You enter the letter (K, L, M, or C) or the number (0, 5, 6, or 9) that corresponds to the frame type you want to use. Frame types and their functions are described in sections throughout this document.

keyword is the name by which the frame can be accessed. Appendix D lists keywords reserved for use by XyWrite.

Keywords

One or more keywords in the label indicate the way(s) the frame can be accessed. In general, there is no limit to the length of a keyword; however, shorter keywords take up less memory.

For example, the following label with the keyword *FileOpen* designates the beginning of a Type K frame:

```
{{K,FileOpen}}←
```

Although shorter keywords are preferable, a keyword can contain two or more words separated by a single space.

If you include several keywords (separated by commas) in a label, the frame can be accessed by different names. For example, the following Type 5 label provides access to the frame in two ways:

```
{{5,Cut,Sh Del}}←
```

Keywords can be either upper- or lowercase letters (or any mix); they are not case sensitive.

Keywords also link frames together, which enables the user to move directly from one task or topic to another. Refer to "Linking Frames" for more information.

Indicating the Entry Point to the Menus

!N is a special keyword you use to indicate the frame you want displayed when the user presses F10 to activate the menus.

In XY4.MNU, the entry point is the Type 0 (zero) frame that produces the menu bar. The keyword label for the menu bar is:

```
{{0,!n}}←
```

In XY4.HLP, the entry point is a Type 5 frame that tests whether the user is requesting help for menus or for the command line. The keyword label for the Type 5 frame is:

```
{{5,!n}}←
```

Linking Frames

INTRO

Three things you will find useful to link frames in a file are the JD (Jump to Dialog Box) command, the JM (Jump to Menu) function call, and the JH (Jump to Help) function call.

JD (Jump to Dialog Box) Command

JD jumps to the frame with the specified keyword and has the following form:

```
JD keyword
```

keyword is the keyword of the frame you want execution to jump to.

All execution in the menu takes place by passing from one menu frame to another. The menu bar is the entry point to the menus. When the user selects an item from the menu bar, a pull-down menu (Type L frame) appears. The JD (Jump to Dialog Box) command links the Type L frame to the Type 0 frame.

For example, the following line in a Type 0 menu frame causes execution to move to the menu frame with the keyword *MBFile* when the user selects "File" from the menu bar:

```
&File « JDMBFile » ,←
```

In the Menu file, you can also use JD to link a Type L to another Type L or a Type 5. In the help file, you can use JD to link Type C frames.

For example, the following line in a Type L menu frame causes execution to move to the menu frame with the keyword *New* when the user selects "New" from the pull-down menu:

```
&New... « JDNew » ,Create a new file←
```

JM (Jump to Menu) Function Call

The JM function call jumps to the menu frame with the specified keyword and is used to control the program flow from one dialog box to the next.

JM has the form:

```
JM (keyword)←
```

keyword is the keyword of the frame where you want execution to continue. (Parentheses are required around the keyword.)

For example, the following Type 5 frame clears macros 1001 and 1002 and then jumps to another menu frame with the keyword *NewA*:

```
{{5,New}}
■«SV1001,» «SV1002,» JM (NewA)■←
```

JM is also useful in XPL programs to create subroutines. Refer to the discussion of Executable Lines in "Creating a Dialog Box" for more information.

JH (Jump to Help) Function Call

You use the JH function call in the Menu file to display the frame that resides in the help file with the specified keyword.

JH has the form:

JH (*keyword*)

keyword is the keyword of the help frame. (Parentheses are required around the keyword.)

For example, the following Type 5 frame jumps to a Type C help frame with the keyword *GenHlp* that resides in the currently loaded help file:

```
{{5,GHelp}}←
■JH (GenHlp)■←
```

Including Comments in a Menu, Dialog Box, or Help File

INTRO

You can use a Type M frame to describe such things as the contents of the file, the contents of a frame, or any changes you've made to the file. You can enter as many lines as you want, since Type M frames are not loaded into memory. Type M frames are also useful when you are testing a menu file, because you can disable one or more frames by changing them to Type M frames.

Keyword Label

The first line of a Type M frame contains the keyword label (M enclosed in a pair of double curly braces {{ }}). Unlike other frame types, a Type M frame label does not require a keyword. However, in XY4.MNU and XY4.DLG, all comment frames have the keyword *Comment* for legibility.

For example, the following is one of the comment lines in XY4.DLG:

```
{{M,comment}}
################# GLOBAL SUBROUTINES #################←
```

You can find other examples in XY4.DLG, where comment frames are used to annotate the functions of various frames.

Displaying a Menu Bar

INTRO

You use a Type 0 (zero) frame to display a menu bar that lists the categories of the various available options. When the user presses F10, XyWrite constructs the menu from the information in the Type 0 frame and displays the menu horizontally across the top of the screen.

Keyword Label

The first line of a Type 0 frame contains the label (0, followed by one or more keywords) enclosed in a pair of double curly braces {{ }}.

Since the Type 0 frame in XY4.MNU is the entry point to the Menu facility, it has the special keyword !N.

Displayable Lines

After the keyword label, each subsequent line of a Type 0 frame contains the number of extra spaces you want between the menu names on the menu bar, the menu name you want displayed in the menu bar, a JD (Jump to Dialog Box) command to link the execution of the menu item to another screen type, and a comma. You can add a string of text you want XyWrite to display on the status line when the user highlights the keyword in the menu.

NOTE

Length of Menu Names. The menu bar must fit on a single line, across the width of the screen. Therefore, when choosing your menu names and spacing, make sure the total number of characters and spaces does not exceed 80.

XyWrite enters menu names in the menu bar in the same order they appear in the Type 0 frame.

For example, the following frame in XY4.MNU creates the menu bar. The label indicates a Type 0 frame that is the entry point to the menus (!N).

```
{{0,!n}}←
&File «JDMBFile»,←
 &Edit «JDMBEdit»,←
 &View «JDMBView»,←
 For&mat «JDMBFormat»,←
 &Proof «JDMBProof»,←
 &Options «JDMBOptions»,←
 &Advanced «JDMBAdvanced»,←
 &Window «JDMBWindow»,←
 &Help «JDMBHelp»,←
```

Displaying a Menu Bar

When the user chooses an item in the menu bar, execution moves to the menu frame indicated by the JD command.

The ampersands in the keywords indicate the accelerator keys; they do not appear on the menu bar. Refer to "Using Accelerator Characters in Menu Screens" for more information.

Creating a Pull-Down Menu

INTRO

You use a Type L frame to display a pull-down menu that lists the various options available for a particular menu bar item. When the user places the cursor on an item in the menu bar, XyWrite displays the pull-down menu directly below it. XyWrite constructs the pull-down menu from the information in the Type L frame.

Keyword Label

The first line of a Type L frame contains the keyword label (L, followed by one or more keywords) enclosed in a pair of double curly braces ({{ }}). The keyword must correspond to a menu name in the Type 0 frame associated with the pull-down menu.

Displayable Lines

After the keyword label, each subsequent line of a Type L frame contains a display symbol, the text and spacing you want displayed in the menu, a JD (Jump to Dialog Box) command to link the execution of the menu item to another screen type, a comma, and the descriptive text you want displayed on the status line whenever the user highlights the keyword on the menu.

NOTE

Length of Displayable Lines. A displayable line is limited to 32 characters up to the comma in the syntax. For each keyword this includes any accelerator keys, the «JDkeyword» command, and spaces. Any descriptive text to be displayed on the status line when the user highlights the keyword is not included in the 32-character limitation.

A pull-down menu can be as long as you want to make it; however, XyWrite displays only the first 22 or 24 lines, depending on the placement of screen elements, such as ruler, menu bar, command line, etc. The user can scroll down to view any additional items.

XyWrite enters keywords in the pull-down menu in the same order they appear in the Type L frame.

Display Symbols

You can use one or more of the following symbols in a Type L menu frame to control the display of the menu screen:

&	indicates the accelerator key (refer to "Using Accelerator Characters in Menu Screens" for more information).
≡	dims the menu item if the current window is empty, contains a read-only directory, or a read-only file or a form. (The item is not dimmed if a read/write file is displayed.) To enter ≡, press Ctrl Alt 240.

∈ dims the menu item if the current window is empty (not used for a directory or a file). To enter character ∈, press Ctrl Alt 238.

░ dims the menu item under all conditions. To enter character ░ press Ctrl Alt 176.

√ adds a ♦ to the item in the displayed pull-down menu if the indicated condition is true. To enter character √, press Ctrl Alt 251.

In Type L menu frames, you can have XyWrite test existing conditions and adjust the information it displays in the pull-down menu depending on the outcome of the tests. You precede the displayable text with a series of statements, each separated by a vertical bar (¦). For example:

```
statement1¦statement2¦statement3..., description←
```

XyWrite reads lines in Type L frames from left to right and evaluates any conditions it encounters. If the condition is true, XyWrite executes the next statement; if the condition is false, XyWrite skips to the next statement.

For example, the following Type L frame from XY4.MNU is displayed when the user selects "Spell" from the Proof pull-down menu.

```
{{L,CCSpell}}←
&Personal Dict...«JDLdDict»,Load or edit personal dictionaries←
≡¦&Main Dict...«JDMainDictAA»,Use a main dictionary←
░¦────────────────────────────────────
«VA$AC»==1¦√¦&Auto-Check    AltF7«JDAutoCheck»,Spell-check
words as you type...←
&Batch Spell...«JDBatSpChk»,Spell-check a series of files←
```

If the user selects 'Personal Dict," execution moves to the menu frame with the keyword *LdDict*. If the current window is empty, contains a read-only directory, a read-only file or a form, XyWrite dims "Main Dict" in the pull-down menu.

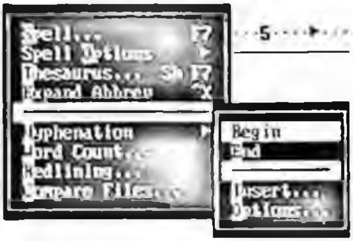
The long horizontal line separates "Personal Dict" and "Main Dict" from "Auto-Check" and "Batch Spell" and the ░¦ ensures that it cannot be selected. If Auto- Check is on, XyWrite displays a ♦ next to "Auto-Check." If the user selects "Batch Spell," execution moves to the menu frame with the keyword *BatSpChk*.

When the user places the cursor on an item in the pull-down menu, XyWrite displays on the status line the descriptive text you associated with that item's keyword in the menu file. "Personal Dict," "Main Dict," "Auto-Check," and "Batch Spell" can be accessed by pressing accelerator keys P, M, A, or B, respectively.

Creating a Pull-Down Menu (cont'd)

Cascading Menus

You can create a pull-down menu that "cascades" from another pull-down menu, that looks like this:



To do this, precede the JD command with an ASCII character 16 (▶). For example, the following displayable line in a Type L menu frame enables the cascading menu in the previous example to work:

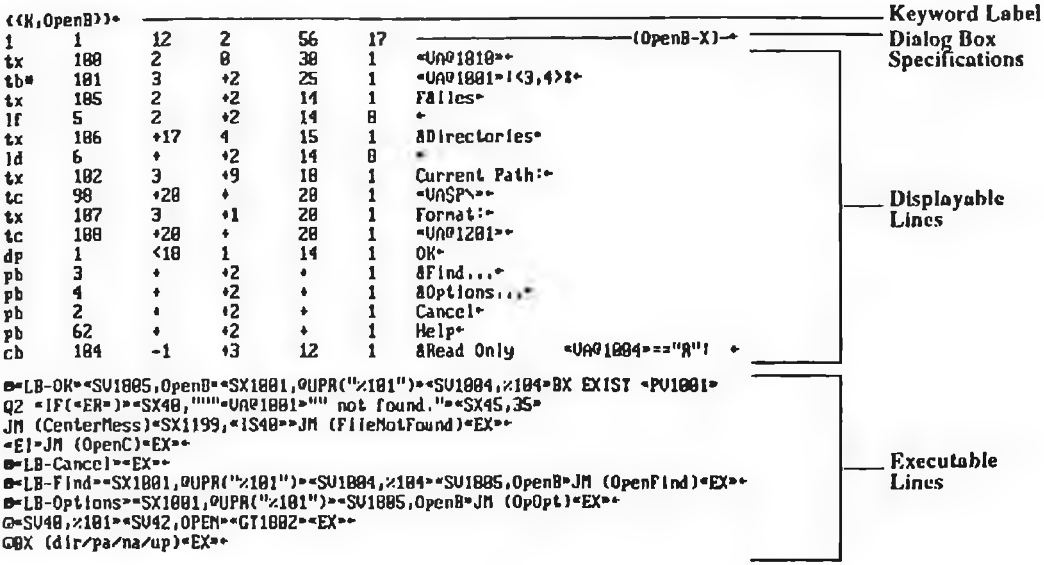
```
≡¦&Hyphenation    ▶«JDCCHyphn»,Control hyphenation
```

When the user selects "Hyphenation" from the Proof menu, XyWrite displays another Type L menu screen with the keyword *CCHyphn*.

INTRO

You use a Type K frame in a menu file to create a general-purpose dialog box. The resulting dialog box emulates a Microsoft Windows dialog box with features such as radio buttons, check boxes, scrollable lists and directories, horizontally scrollable fields, etc.

For example, the following Type K frame appears in XY4.DLG and has many of the features described throughout this document:



```
«(K,OpenB)»←
1       1       12      2       56      17      ──────────────(OpenB-X)←
tx      100     2       0       30      1       «VA@1010»←
tb*     101     3       +2      25      1       «VA@1001»!<3,4>!←
tx      105     2       +2      14      1       F&iles←
lf      5       2       +2      14      0       ←
tx      106     +17     4       15      1       &Directories←
ld      6       +       +2      14      0       ←
tx      102     3       +9      18      1       Current Path:←
tc      98      +20     +       20      1       «VA$P\»←
tx      107     3       +1      20      1       Format:←
tc      108     +20     +       20      1       «VA@1201»←
dp      1       <18     1       14      1       OK←
pb      3       +       +2      +       1       &Find...←
pb      4       +       +2      +       1       &Options...←
pb      2       +       +2      +       1       Cancel←
pb      62      +       +2      +       1       Help←
cb      104     -1      +3      12      1       &Read Only      «VA@1004»=="R"!  ←

●«LB-OK»«SV1005,OpenB»«SX1001,@UPR("%101")»«SV1004,%104»BX EXIST «PV1001»
Q2 «IF(«ER»)»«SX40,""""«VA@1001»""" not found."»«SX45,35»
JM (CenterMess)«SX1199,«IS40»»JM (FileNotFound)«EX»←
«EI»JM (OpenC)«EX»←
●«LB-Cancel»«EX»←
●«LB-Find»«SX1001,@UPR("%101")»«SV1004,%104»«SV1005,OpenB»JM (OpenFind)«EX»←
●«LB-Options»«SX1001,@UPR("%101")»«SV1005,OpenB»JM (OpOpt)«EX»←
@«SV40,%101»«SV42,OPEN»«GT100Z»«EX»←
@0X (dir/pa/na/up)«EX»←
```

Type K menu frames consist of four basic parts: the keyword label, dialog box specifications, displayable lines, and executable lines.

The *keyword label* is similar to those used for other type menu frames.

The line following the keyword label contains *dialog box specifications* for the size and placement of the dialog box as well as any text you want displayed as a title.

Displayable lines indicate the various items you want to appear in the dialog box and their size and placement. For example, you can include fixed text, a field for entering text, radio buttons, check boxes, pushbuttons, and scrollable lists. Displayable lines also can:

- indicate the tab order, that is, the order in which the user moves through the items in the dialog box

- indicate how the items appear when the dialog box is displayed
- have XyWrite dim an item under certain conditions
- have XyWrite verify that the information entered is valid for the function being performed
- indicate shortcuts, called accelerator characters, so the user can move directly to an item by pressing a key
- assign values and conditions that affect other items in the dialog box

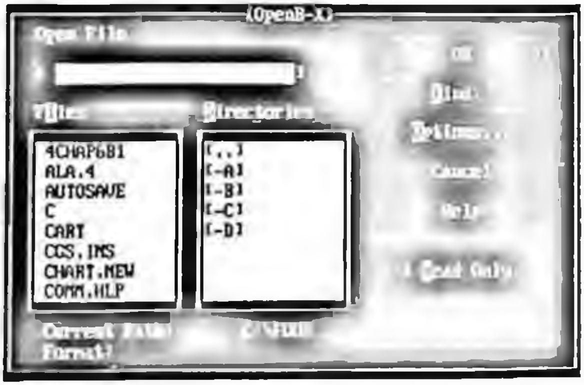
Executable lines all begin with either an ASCII character 2 (☻) or an ASCII character 1 (☺) to indicate the start of an XPL (XyWrite Programming Language) program. They are associated with items in the dialog box and perform the task indicated. You can pass information the user enters into various dialog box items to executable lines.

The memory requirements for the executable lines of a Type K menu frame are limited to approximately 3K – 10K, depending on the amount of memory used by the currently loaded set of macros.

You can test dialog boxes quickly while you are creating or editing them without saving and reloading the menu. To do this, position the cursor in the frame you want to test and press Ctrl Alt Shift Q. XyWrite immediately displays the associated screen.

For more information refer to "Testing Dialog Boxes and Help Frames."

The Type K menu frame shown above produces the following dialog box:



The user presses Tab to move through any dialog box from item to item (or Shift Tab to move backward from item to item). Cursor keys only move the cursor within fields. When the last (or first) item is reached, the cursor moves through the list again. Alternatively, the user can press an accelerator key to move directly to a specific field.

When the user puts the cursor on an item in a dialog box and presses F1, XyWrite displays the information for the specific item. Refer to "Providing Context-Sensitive Help Screens" for more information.

Keyword Label

The first line of a Type K menu frame contains the keyword label (K, followed by one or more keywords) enclosed in a pair of double curly braces {{ }}. In general, there is no limit to the length of a keyword; however, shorter keywords take up less memory.

NOTE

Illegal Characters. You cannot use ASCII characters 174 («) or 175 (») in keywords; they cannot be accessed with the JM (Jump to Menu) function call.

If you plan to provide on-line help for a dialog box, the keyword for the dialog box cannot end with a number (for example, OPEN2).

Dialog Box Specifications

Dialog box specifications determine the size of the dialog box, its placement on the screen, and the title that you want to appear in its top border.

Dialog box specifications appear on the second line of a Type K frame in the following form. The four fields must be separated by tabs.

scale	*placement*	*size*	*title*

scale values indicate the factors by which you want XyWrite to scale the numbers you enter to indicate horizontal and vertical units. By specifying values of 1 and 1 for the scale argument, you can enter values for x and y coordinates in one character units and one line units, respectively. Microsoft Windows requires 4 horizontal units and 10 vertical units per character. Therefore, if you import dialog boxes from Microsoft Windows, specify values of 4 and 10 for the scale argument.

placement values are x and y coordinates to indicate the location of the upper left corner of the dialog box from the left and top edges of the screen.

size values are x and y coordinates to indicate the width and height of the dialog box.

title indicates the title you want to appear in the upper left corner of the dialog box border.

For example, in the following dialog box the *scale* values are 1 and 1, *placement* values are 12 and 2, *size* values are 56 and 17, and the *title* is Open:

```
1	1	12	2	56	17	Open
```

The specifications in the above example produce a dialog box 56 characters wide and 17 lines high. Its upper left corner is located 12 characters from the left edge of the screen and 2 lines down from the top edge.

TIP

Use the Tab Settings Command. Since you must use tabs to separate the various values in the dialog box specifications and in the displayable lines, use the TS (Tab Settings) command to make the columns readable. You can enter the TS command immediately before the Type K keyword label.

Displayable Lines

The third and subsequent lines of the Type K frame determine the various items you want to appear in the dialog box and their placement and have the following form. The fields for the displayable lines must be separated by tabs.

control ID location size text option

control indicates one of the control codes that determines the type of dialog box item. The available codes are:

TX	Text (fixed)
TC	Text changing (updated by XyWrite; not editable by user)
TB	Text box (editable)
RB	Radio button
CB	Check box
PB	Pushbutton
DP	Default pushbutton
LF	List files
LD	List directories and drives
LB	List box
GB	Group box

(Each of these controls is described in a separate section below.)

A Type K menu frame can have up to 32 controls, including the dialog box itself. XyWrite counts the dialog box specification as one control and a *group* of radio buttons as one control. XyWrite does not count TX (Fixed text) or GB (Group box) controls.

To indicate where you want the cursor located when XyWrite first displays the dialog box, put an asterisk immediately after one control code; for example tb*.

Any line that begins with a semicolon is a comment – it has no effect on dialog box items.

You use a different line for each control you want to include in the dialog box. The order in which the controls appear in the frame determines the tab order or the order in which the user can move the cursor from item to item, by pressing Tab, when the dialog box is displayed. If you want to rearrange the tab order, you simply rearrange the lines in the menu frame.

ID indicates the number you assign to each control code. Each number must be unique within the menu frame; however, the numbers can be in any sequence. You can correspond these numbers directly to programs XyWrite runs when the user selects a pushbutton. (See "Executable Lines," below.) Since you normally would not correspond fixed text to a program, it is not necessary to assign ID numbers to TX controls, but it is helpful so you can tell which dialog box items overlap.

NOTE

Reserved ID Numbers. XyWrite reserves certain ID numbers for special features. Refer to Appendix D for more information.

location values are x and y coordinates to indicate the location of the item in relation to the upper left corner of the dialog box. You can use either a plus sign (+) or a minus sign (-) to specify relative locations. See "Positioning Items in a Dialog Box" for more information.

size contains x and y coordinates to indicate the width and height of the item (text, text box, pushbutton, check box, list box, etc.). To save time, you can reuse the value above it by including a plus sign (+) for the width and/or height.

text specifies the text or the initial value displayed next to (or within) the item in the dialog box. You can use the VA (Value of Variable) command when you want to include the current value of a XyWrite variable in the text. For example, when the dialog box is displayed, XyWrite includes the name of the current drive and path in the text after the word, "Current path."

```
tx  102 +  +20  100  12  Current path: «VA$pa»
```

You can also use the *text* argument to have XyWrite verify that the information entered in a text box (TB) is valid for the function being performed. Refer to "Validating Text Entry" for more information.

option has many uses: you can select a radio button or a check box automatically when the dialog box is displayed; dim an item in the dialog box; and include conditional statements to perform these tasks. Indicating default check boxes and radio buttons are described in their respective sections. Refer to "Dimming or Checking Dialog Box Items" for more information.

Positioning Items in a Dialog Box

You position items in a dialog box by specifying their coordinates. You specify horizontal coordinates in characters and vertical coordinates in lines. If you are using a scale of 1 x 1, the allowable coordinate values are 1-80 (horizontal) and 1 to 22 (vertical).

You specify *location* coordinates in relation to the upper left corner of the dialog box. (Prefacing a location coordinate with a < character indicates that you want to place the item in relation to the *right* border of the dialog box.)

NOTE

Radio Button Positions. XyWrite positions radio buttons one line apart, regardless of what you specify as a vertical coordinate. You cannot position radio buttons side by side.

To use absolute locations, type the exact coordinates where you want the items placed. For example, the following line positions a block of fixed text (ID number 102) 2 characters in from the left border of the dialog box and 1 line down from the top border. The text, "Example 1," is one line of up to 20 characters long.

```
tx  102  2  1  20  1  Example 1
```

To use relative locations, type the coordinate as you want the item to appear in relation to the preceding coordinate, using either a plus sign (+) or a minus sign (-).

For example, the first control line functions as described above; the second control line positions a text box (ID number 102) 3 characters in from the left border of the dialog box and 3 lines down from the top border. The text box is one line, 22 characters long and is initially displayed containing the name of the current file.

```
tx    102    2     1     20    1      Example 1
tb    103    +1    +2    22    1      «VA$fi»
```

TIP **Use Relative Locations.** It is much easier to redesign a dialog box with relative locations, because you need only change one number to move a column of items.

NOTE **Item's Position May Change.** The position of the item will change if you use relative positions and you rearrange the lines in the menu frame to change the tab order.

In order to have items line up vertically, specify either +0 or + as the horizontal coordinate. For example, the following control lines vertically align two radio buttons under a block of fixed text that is 12 characters in from the left border and 2 lines down from the top:

```
tx    102    12    2     20    1       Example 2
rb    103    +     +1    10    1       ODD
rb    104    +     +1    +     1       EVEN
```

Notice that the second radio button (ID number 104) is also 10 characters wide as specified by the + in the width *size* coordinate.

As another example, the following control lines vertically align two pushbuttons, starting 12 characters in from the right side of the dialog box:

```
dp    1      <12   1     10    +       &OK
pb    2      +     +2    +     +       &Cancel
```

Executable Lines

Following the list of control codes and placement information, ASCII character 2 (☻) or ASCII character 1 (☺) indicate an executable line. Either character indicates the start of an XPL (XyWrite Programming Language) program; however, an ASCII character 1 (☺) executes the program without removing the dialog box from the screen.

You must have an ASCII character 1 or an ASCII character 2 for each pushbutton you include in the dialog box.

NOTE **ASCII Character 1 Programs.** ASCII character 1 programs are useful in limited situations. One application is to refresh the contents of list files in a dialog box. For example, the following two lines in a Type K menu frame display a directory of filenames and a directory of paths in list boxes. The ASCII character 1 at the start of both lines enables the directories to be updated without removing the dialog box from the screen:

```
☺BX (dir/fi/na %101)«EX»
☺BX (dir/pa/na)«EX»
```

Four things you will find useful in executable lines are passing information to XPL programs, the BX (Blind Execute) function call, the JM (Jump to Menu) function call, and checking unchanged fields.

Passing Information to XPL Programs. You can use information from dialog box items in your XPL programs by prefacing the item's ID number with a %. If the ID number corresponds to a list box, the text the user highlights with the cursor is accessed by the program.

For example, the following produces a directory based on the path the user enters in the text box that has ID number 101:

```
☺BX (dir/fi/na %101)«EX»
```

BX (Blind Execute) Function Call. The BX function call executes a command from within a program without moving the cursor to or clearing the command line. If the command and any arguments are enclosed in parentheses ( ), square brackets [ ], or curly braces { }, XyWrite executes the command automatically. For example:

```
BX (ca test)
```

If they are not so enclosed, you must use the Q2 (Execute Too) function call with the BX function call to execute the command. Q2 is useful for executing a command string that may contain parentheses, square brackets, or curly braces. For example:

```
BX ca testQ2
```

NOTE

Illegal Characters. You cannot use angle brackets < > to enclose a command specified by the BX command.

JM (Jump to Menu) Function Call. The JM function call jumps to the menu frame with the specified keyword and is very useful in XPL programs in three ways:

- To control the program flow from one dialog box to the next
- To jump back to the same dialog box if a condition is not satisfied
- To call a subroutine; for example, to establish initial conditions for the program

JM has the form:

JM(*keyword*)

keyword is the keyword of the frame or subroutine

NOTE

The Exit Command. An EXIT command «EX» in the subroutine causes execution to return to the main program and resume.

For example, the following code jumps to a subroutine named SETNW3, calls a file the user specifies in a text box (ID 101), and jumps to another subroutine named RESTORENW:

```
JM(SETNW3) BX(call %101) JM(RESTORENW)
```

The subroutines are included in the same menu file in Type 5 frames. For example:

```
{{5,SETNW3}}
■«SX70,«VA$NW»» BX(d nw=3)«EX»

{{5,RESTORENW}}
■BX(d nw=«PV70»)«EX»
```

Checking Unchanged Fields. You can use the HF command in the program to direct XyWrite to execute portions of the program only if a particular field has changed. For example, if the user does not change the entry displayed in a text box, XyWrite skips that part of the program from the HF command to the EI (End If) command that follows.

HF has the form:

«HF*id*»*action*«EI»

id is the number of the control code for the item (usually a text box or a check box) that you want to determine is changed.

action is one or more commands you want executed if the item is changed.

For example, in the following routine execution moves to the label, "-Changed" if the user has changed the text box with ID 100; otherwise, execution jumps to the menu frame with the keyword *PrSettings*:

```
■«HF100»«GL-Changed»«EI»JM (PrSettings)«EX»
```

Displaying Fixed Text

There are two controls to display fixed text in a dialog box: TX and TC.

You use the TX (Text) control code to indicate a line of fixed text that is always displayed in the dialog box. In the *text* argument, you can specify up to 400 characters per TX control code. To have XyWrite automatically calculate the width of the item based on the width of the text, enter the @ character when specifying a width for the *size* argument.

When the text is displayed in the dialog box, the user cannot move the cursor to it.

For example, the following line in a Type K menu frame produces a line of fixed text that contains the words, "PRINT FILE:"

```
tx  100  2  1   @  1   PRINT FILE
```

You use the TC (Text Changing) control code to indicate a line of fixed text that XyWrite can update to reflect changing conditions. When the text is displayed in the dialog box, the user cannot select it; however, whenever a change occurs, the information is immediately updated to reflect the change.

For example, the following line in a Type K menu frame displays the current path name. When the current path changes, the dialog box immediately displays the new path.

```
tc  107  +90  +  70  12  «VA$pa»
```

Displaying a Text Entry Field

You use the TB (Text Box) control code to include a field into which the user can enter text. The user can select the displayed field either by pressing Tab or by using the mouse. XyWrite includes any *text* you specified in the control line when the dialog box is displayed. This text is automatically erased when the user starts typing in the first character position. However, if the user presses a cursor key first, XyWrite does not erase text when the user starts typing.

You can have XyWrite validate the text the user enters into the field. See "Validating Text Entry" for more information.

NOTE **Specifying Text Box Locations.** XyWrite uses square brackets to display the borders of a text box; for example, [This is a text box   ]. However, it does not include the square brackets when calculating the item's location coordinates. Therefore, when specifying location coordinates for text boxes, add one extra space to compensate for the left square bracket. (See the example on specifying relative locations in "Positioning Items in a Dialog Box.")

For example, the following line in a Type K menu frame produces a text box that contains the name of the current file:

```
tb  101  +1  +2  22  1  «VA$fi»
```

You can assign an accelerator key to a text box by preceding it with a TX (Text) control that has an accelerator key. Including an accelerator character in a TX control line forces the cursor to move to the next item when the dialog box is displayed and the accelerator key is pressed. Refer to "Using Accelerator Characters in Menu Screens" for more information.

Using Radio Buttons in a Dialog Box

A group of radio buttons covers all possibilities of a task. Radio buttons are like the buttons on a car radio – when one is pushed in, another pops out – they indicate items that are mutually exclusive.

Radio buttons may be organized into more than one group. The user must select one *for each group* in order to execute the dialog box. However, only one button in the set can be selected at a time.

You use the RB (Radio Button) control code to include radio buttons in a dialog box. Consecutive RB control codes form a set from which only one can be selected at one time.

For example, the following line in a Type K menu frame produces a set of radio buttons that enables the user to choose ALL, EVEN, or ODD:

```
rb    103    12    +1    10    1    ALL
rb    104    +     +1    +     1    EVEN
rb    105    +     +1    +     1    ODD
```

Enter ASCII character 251 as the *option* argument (after the text) to indicate the radio button that is automatically selected when the dialog box is displayed. To enter the character, type Ctrl Alt 251. You can also include a conditional statement to select the radio button depending on the current environment.

For example, the following line in a Type K menu frame selects radio button 3 if the display is draft view:

```
rb  3  +5  +12  40  12  Button &1  «VA$dt»==1|√
```

When XyWrite displays the dialog box, it indicates the selected radio button setting with an asterisk.

Creating a Dialog Box (cont'd)

Using Check Boxes in a Dialog Box

You use the CB (Check Box) control code to include a box the user can turn either on or off; it is equivalent to having the user type "Yes" or "No" to indicate a choice. You can have one or more check boxes in a dialog box; however, usually you would not group them.

For example, the following line in a Type K menu frame produces a check box that enables the user to choose to print only the selected block:

```
cb  107  12  +  25  1  &Print selected block
```

The user presses Spacebar to turn any check box on or off. XyWrite displays an "X" to indicate an activated check box.

Enter ASCII character 251 as the *option* argument (after the text) to indicate that a check box is automatically selected when XyWrite first displays the dialog box. To enter the character, press Ctrl Alt 251. You can also include a conditional statement to select the check box depending on the current environment.

For example, the following line in a Type K menu frame produces check box that is preselected if macro 1002 contains an "R:"

```
cb  107  -1  +3  11  1  &Read Only  «VA@1002»=="R"|√
```

Using Pushbuttons in a Dialog Box

Pushbuttons represent choices associated with procedures or execution; they begin or apply the action. Every XyWrite dialog box has at least one pushbutton.

You use the PB (Pushbutton) control code to include a pushbutton in a dialog box; you use the DP (Default Pushbutton) control code to include the default pushbutton. XyWrite displays the default pushbutton with a double angle bracket (<< >>). In XyWrite, the OK pushbutton is usually the default.

You include one XPL program for each pushbutton. The pushbutton with ID 1 runs the XPL program designated by the first ASCII character 2 (☻) or ASCII character 1 (☺); the pushbutton with ID 2 runs the second, etc.

There are several predefined pushbuttons you can use; for example, "OK" has 1 as an ID number and "CANCEL" has 2 as an ID number. ID numbers 50 and over are reserved for special functions — they do not correspond to an XPL program, but perform a function automatically. For example, "HELP" has 62 as an ID number and displays a help screen when the user selects it. Refer to Appendix D for a full list.

For example, the following lines in a Type K menu frame include a default pushbutton (OK) and a CANCEL pushbutton in the dialog box:

```
dp   1    40   1    10   1    OK
pb   2    +    +2   +    +    Cancel
```

Using Scrollable Lists in a Dialog Box

You can include three types of scrollable lists in a Type K menu frame:

- Files only
- Drives and Directories
- Any words you specify

You use the LF (List Files) control code to list all the files in a directory.

You use the LD (List Drives and Directories) control code to list all the drives available on your computer and directories available from the current directory. The user can select a drive by moving the cursor to a specific drive in the list and pressing ↵. If there is an asterisk (*) in the text box when the user executes the OK pushbutton, XyWrite displays a directory of the selected drive rather than execute the program associated with the first pushbutton.

For example, in the following Type K frame, the text box (ID 101) contains *.* to produce a list of files. If the user selects an item from the List Files Box (ID 105), XyWrite updates the contents of the text box with the selected information. If the text box contains a wild card and if the user selects the OK pushbutton (ID 1), XyWrite builds a new directory using that specification.

```
{{K,Open}}
1     1     6     2     66    16
tx    101   2     1     @     1     Open File:
tb*   102   2     +1    25    1     «VA@1001»
lf    6     2     +3    20    8     *.*
ld    106   +22   +0    20    8
tx    104   2     +9    40    1     Current Directory: «VA$pa»
dp    1     50    1     11    1     &OK
pb    3     +0    +2    11    1     &Find...
pb    4     +0    +2    11    1     O&ptions...
pb    2     +0    +2    11    1     &Cancel
pb    5     +0    +2    11    1     &Help
cb    107   +0    +3    11    1     &Read Only
```

You use the LB (List Box) control code to list any words you want. You specify the words in the *text* argument, separated by ASCII character 124 (¦). You can include a list of macros in a List box. For example:

```
lb 102 3 3 22 1 «VA@111»¦«VA@121»¦«VA@122»¦«VA@98»
¦«VA@101»←
```

The List Files Box, the List Drive and Directories Box, and the List Box can be linked to the closest text box above it in the list; so that whenever the user selects an item in one of the boxes and presses Spacebar, XyWrite inserts the selected item into the text box.

Special Lists. There are also some special values you can use to produce lists with the LB control:

@*x* lists the contents of a macro (x).
?*x* lists one of the following:

?AS	symbol sets
?BO	BORDER commands in the current document
?DZ	all sizes available for the default font and highlights the default size
?FO	fonts used in the current document
?IG	graphic files included in the current document
?MA	all macros
?PC	printer control codes
?PP	printers listed in the PP Table of SETTINGS.DFL
?PR	files currently waiting to be printed (automatically updated)
?SB	symbol set available for the current font
?SG	sizes of all macros (not contents), including extended macros and special macros for XyWrite
?SP	spelling alternates (used only by the spelling dialog box)

?SS	styles for the current file
?ST	commands in the command stack
?SY	synonyms (used only by the Thesaurus)
?SZ	sizes available for the current font
?WI	all open windows
?[*filename*	contents of *filename*
?fr*keyword*	contents of type 5 frame with label *keyword*

For example, the following produces a list box that when displayed lists all the BORDER commands in the current document:

```
lb   240   5   +2   15   2   ?BO
```

List Services. The List Services feature enables you to display a directory, select multiple files in the directory, and execute a command that affects all the selected files, one after another.

To display a directory from which multiple files can be selected, use the /LO switch with the DIR command. For example:

```
☺BX (dir/lo %101)«EX»
```

NOTE

DIR/LO in Executable Lines. When using DIR/LO in an executable line, you must always use ASCII character 1 (☺), never ASCII character 2 (☻).

To select a filename from the displayed directory, put the cursor on the filename and press the space bar. XyWrite indicates selected filenames with an asterisk.

To process a list of selected filenames with a specific command, use the /LS switch with one of the following commands:

ATTRIB (Change Attribute)
COPY (Copy a File)
COPY/MV (Move a File)
DEL (Delete a File from Disk)
REN (Rename a File)

For example, the following removes from disk all the files selected in the list box:

```
☺BX (del/ls %101)«EX»
```

Grouping Items in a Dialog Box

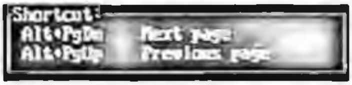
You use the GB (Group Box) control code to enclose related items, such as radio buttons, in a rectangular box. Optionally, you can add a title to the box, which XyWrite displays in the top border of the Group box. Group boxes organize items visually on the screen; they do not functionally group them.

When the dialog box is displayed, the user cannot move the cursor to the Group box unless it contains controls that the user needs to access (radio buttons, list boxes, etc.).

For example, the following groups two lines of fixed text into a box:

```
gb   107   1    6    35   4     Shortcut:
tx         +2   +1   28   1     Alt+PgDn   Next page
tx         +    +1   28   1     Alt+PgUp   Previous page
```

When the dialog box is displayed, the group box appears like this:



Shortcut:
Alt+PgDn   Next page
Alt+PgUp   Previous page

Dimming or Automatically Selecting Dialog Box Items

To display in a lighter screen display face any item that you do not want the user to access or to automatically select an item, include the following as the *option* argument:

condition1 ¦ *action1* ¦ *condition2* ¦ *action2*

condition is a test that can be evaluated to have a true or false value; it can be any boolean expression, such as normally found in an IF statement.

action is the action to take if the corresponding condition is true. To dim an item, enter ASCII character 176; to automatically select an item, enter ASCII character 251.

A condition usually starts with an embedded command. If there is no condition, it is assumed to be true. For example, the following option argument tests the value of the Default BK (Backup of Files) setting. If the setting is on (BK=1), XyWrite selects the item:

```
«vabk»==1¦√←
```

For another example, the following option argument uses $WS (Window Status) to test if there is a file open. If the window is empty ($WS=0), XyWrite dims the item:

```
«va$ws»==0¦▒←
```

You can test two or more conditions to determine whether or not to select or dim a radio button or a check box. The test can be true in one case or the other or in both cases (similar to OR and AND conditionals in programming).

To test two or more conditions (OR), one or more of which must be true in order for the test to be true, list each *condition* followed by a vertical bar (ASCII character 124) and the *action* to take if the condition is true. For example the option argument in the following control line checks to see if line spacing is set to .50 *or* if automatic leading is off:

```
rb  5  2  2  10  1  Half  «vals»==.50¦√|«vaal»==0¦√←
```

If line spacing is set to .50, XyWrite selects radio button 5 as the default and exits. Otherwise, it tests to see if automatic leading is off; and if so, it selects radio button 5 as the default and exits.

To test two or more conditions (AND), all of which must be true in order for the test to be true, list each *condition* followed by a vertical bar (ASCII character 124). At the end of the list include the *action* to take if the condition is true.

For example, the option argument in the following control line checks to see if line spacing is set to .50 *and* automatic leading is off:

```
rb  5  2  2  10  1  Half  «vals»==.50¦«vaal»==0¦√←
```

If both conditions are true, XyWrite selects radio button 5 as the default.

Validating Text Entry

You can have XyWrite determine whether the text the user types into a text box is a valid argument for a specific XyWrite command. You include the validation in the text argument of the TB (Text Box) control line.

Separate the validation from the text with the string ;**; followed by one of the following validation type indicators.

> indicates that the field is mandatory. If the field is left blank, the user cannot execute the dialog box. > must appear before any other validation.

< indicates a list of pushbuttons you *do not* want validated. (In XyWrite, pushbutton 2 is always Cancel and is never validated and pushbutton 62 is always help and never validated.) For example, the following avoids validation if the user chooses pushbutton 2, 3, 9, or 62: ;**;<3,9.

NOTE One Validation Per Control. You can only use one validation with each control with the exception of < and >. < and > are the only validation indicators you can use in conjunction with another type of validation.

@ immediately followed by a two-letter embedded command validates that the value the user enters into the text box is an acceptable argument for that command. If it is not, XyWrite displays the same error message it would if the command had been executed from the command line. For example, the following verifies that the user enters an acceptable value for the Left Margin command: ;**;@lm.

# indicates that the user can enter only an integer into the field.

- indicates that the user must enter two integers separated by a hyphen.

. indicates that the user can enter either a number with a decimal point or an integer into the field.

+ indicates that the expected user response is a valid math expression, such as 9*3.

% indicates that the expected user response is a single alphanumeric character (A to Z or 0 to 9).

\ validates that the path and drive the user entered exist.

↨ (ASCII character 23) validates that the filename the user enters does not contain any invalid characters (for example, * + = , . ? " / \ [ ] ; : « » |), that there are eight characters or fewer in the filename, and that there are three characters or fewer in the extension.

= validates that the filename the user entered exists, and performs the same validation as ↨.

^ validates that the filename the user entered *does not* exist.

$ accepts a number with a leading dollar sign if the user enters one; i.e., 11.50 and $11.50 are both valid user entries.

! indicates that the expected user response is a page-line number; for example, 1-4 (page 1, line 4).

/ indicates that the expected user response is a string of page number ranges; for example, 1-4/9-12 (pages 1 through 4 and 9 through 12).

] accepts the word "AUTO" used either for the SI (Size) argument in the FA (Frame Area) command or for line spacing in footnote and text formatting commands. Enter it with the two-letter command; for example, ;**;]RM.

¶ (ASCII character 20) validates that the user entered an acceptable vertical position for the FA (Frame Area) command (PC, LP, CP, RP, LC, CC, or RC). Enter it with a two-letter command that also requires a valid numeric vertical position, such as LS; for example, ;**;¶LS.

§ (ASCII character 21) validates that the user entered an acceptable horizontal position for the FA (Frame Area) command (PC, TP, CP, BP, TC, CC, or BC). Enter it with a two-letter command that also requires a valid numeric horizontal position, such as RM; for example, ;**; §RM.

[validates that the style the user entered exists in the current document.

For example, the following indicates that if the user selects any pushbutton other than 2, 3, 9, or 62, XyWrite validates that the value entered in the mandatory field is a valid argument for the Left Margin command:

```
;**;<3,9>@lm
```

As another example, the following text argument of a TB control line displays the current page width setting and validates that whatever new value the user enters into the field is a valid argument for the Page Width command:

```
«VApw»;**;@pw
```

Using One Control to Affect Another

You can set up text boxes, radio buttons, or check boxes to affect other controls in two ways:

- Entering text into a text box can automatically select a radio button.
- Selecting a radio button or a check box can change the contents of text boxes (TB or TC) and check or uncheck check boxes.

You do this by assigning values and conditions to controls. Selecting a pushbutton executes the XPL program whose ordinal position corresponds to the control's ID number — for example, the pushbutton with ID 1 runs the XPL program designated by the first ASCII character 2 (☻) or ASCII character 1 (☺); the pushbutton with ID 2 runs the second, etc.

By assigning an appropriate ID to a text box, you can automatically select a radio button as soon as the user types into that text box.

By assigning an appropriate ID to a radio button or a check box, you can provide an initial value for a text box or select a check box as soon as the user selects the control.

In either case, you use the following format for the executable line:

☻*c1=string1*¦*c2=string2*¦

c1 and *c2* are control numbers.

string1 and *string2* are the values you want included in controls C1 and C2, respectively. You could use the VA command to insert a current value into the control.

NOTE **Ending Executable Lines.** You must end each executable line with either an ASCII character 124 (¦) or an ASCII character 1 or an ASCII character 2.

For example, the following executable line automatically enters values into two fields if the user selects a radio button:

```
☻10=8.5¦11=11¦
```

Assume that the example executable line is the fifth in the group of executable lines for a "Page Setup" dialog box. When the user selects the "Letter" radio button that has an ID of 5, XyWrite automatically enters a value of 8.5 into the "Width" text box that has an ID of 10 and a value of 11 into the "Length" text box that has an ID of 11.

As another example, the following executable line outputs the contents of macro 1191 into the text box that has an ID of 6 and outputs the location of the startup file in the text box that has an ID of 7 if the user selects a radio button:

```
■6=«VA@1191» |7=«VA$st» |
```

As another example, the following executable line automatically selects the radio button that has an ID of 9 when the user enters text into a text box:

```
■9=1 |
```

Executing a Routine

Intro

You use a Type 5 frame in either the menu, dialog box, or help file to execute commands and programs. Although Type 5 frames may contain displayable lines, they are usually used as "routine only" frames, which initialize some variables before moving to a dialog box or a help frame.

Keyword Label

The first line of a Type 5 frame contains the label (5 followed by one or more keywords) enclosed in a pair of double curly braces {{ }}.

Displayable Lines

A Type 5 frame can contain up to twenty displayable lines. Refer to the next section, "Positioning Menu Screens," for information on determining the placement of Type 5 displayable lines on the screen.

If you do not want to display a screen, simply include the program as the second line of the menu frame.

Executable Lines

The content of the executable lines depends on the function you want the frame to perform. In order to execute a program, start the line with an ASCII character 2 (☻).

Executable lines can wordwrap. An executable line begins with the ASCII character 2 (☻) and ends at the next ASCII character 2 or EX command.

For an example, the following Type 5 frame contains a routine that saves the current status of the NW (New Window) default to macro 70 before setting the default to 0:

```
{{5,SetNW0}}
☻«SX70,«VA$NW» » BX (d NW=0)☻
```

After XyWrite performs the function that calls the SETNW0 subroutine, it can restore the original settings by calling another subroutine:

```
{{5,RestoreNW}}
☻BX (d NW=«PV70»)☻
```

Positioning Dialog Box and Help Screens

INTRO

You use the DGW (Menu Screen Window Position) command in a menu file to position a Type 5 frame or in a help file to position a Type C frame on the screen.

DGW has the following format:

```
dgw l,t,w,d
```

l is the number of columns you want the screen displayed from the left of the window

t is the number of lines you want the screen displayed from the top of the window; you must specify a number greater than 3

w is width of the screen specified in columns

d is the depth of the screen specified in lines.

For example, the following line in a frame positions a screen that is 78 columns wide and 20 columns deep, 1 column from the left and 4 lines from the top of the window:

```
«DGW1,4,78,20»
```

Using Accelerator Characters in Menu Screens

INTRO

When a menu screen is displayed, the user can move directly to a specific item by pressing the accelerator key associated with the item. When a menu screen is displayed, accelerator keys are highlighted.

In the pull-down menus, accelerator keys are the highlighted characters (for example, F); for the menu bar and dialog boxes, the accelerator keys are Alt and the highlighted character (for example, Alt A).

You set this up in the menu frame and in the keyboard file.

Menu Bar and Pull-Down Menus

To indicate the accelerator key in the menu frame that displays the menu bar or a pull-down menu, simply precede the keyword character you want as the accelerator with an ampersand (&).

For example, the following two displayable lines in the Type 0 frame in XY4.MNU designate F as the accelerator character for the File item and M as the accelerator character for the Format item:

```
{{0,In}}←
&File «JDMBFile»,←
:
:
For&mat «JDMBFormat»,←
:
```

When the menu bar is displayed, the user can make a selection by pressing the letter key that corresponds to the highlighted letter in the word (for example, press F to select File; press M to select Format).

Dialog Boxes

To indicate the accelerator key for a dialog box item, precede the character you want as the accelerator with an ampersand (&) in the *text* argument of the item's control line.

For example the following entry designates "2" as the accelerator key for the radio button (ID 105):

```
rb  105  +  +12  +  +  Button &2
```

Including an accelerator character in a TX control line forces the cursor to move to the next control the cursor can go to when the dialog box is displayed and the accelerator key is pressed.

For example the following entry designates "N" as the accelerator key for the text box (ID 101):

```
tx     100    8      8      100    12     &New Filename:
tb*    101    +      +14    100    12     123456789:
```

The cursor moves to the text box when the user presses Alt N.

NOTE

Using Accelerator Characters with Pushbuttons. If you associate an accelerator character with a pushbutton, XyWrite executes the program associated with the pushbutton when the user presses Alt and the accelerator key. It is as if the user pressed Tab to move to the item and pressed ↵.

Keyboard File

Assign the AK (Accelerator Key) function call to the corresponding character in the ALT table of the keyboard file. The AK function call has the form:

```
AK,accel
```

accel is the accelerator character you specified in the menu frame.

For example, the following notation in the ALT Table of a keyboard file assigns the accelerator character 2 to Alt 2:

```
3=ak,2
```

When the dialog box is displayed and the user presses Alt 2, the cursor moves directly to the text with "2" as the accelerator character.

Providing Context-Sensitive Help Screens

INTRO

Context-sensitive help screens provide information about a specific phase of a task the user is trying to accomplish. You set up the help screens differently, depending on the way they will be accessed: through help pushbuttons, individual dialog box items, or pull-down menus or other screen types. You can also provide help screens for command markers.

Help Pushbuttons

You can provide help for an entire dialog box by including a frame in the help file with a label in the following format:

{{#,*keyword*}}←

# indicates the frame type.

keyword is the name of the menu frame in the menu file.

For example, the following label designates the beginning of a Type C frame in the help file that provides information on the menu frame that has the keyword *PrintToFile:*

```
{{C,PrintToFile}}←
```

Result: When the user activates the help pushbutton in the dialog box, XyWrite displays the information contained in the help frame.

Refer to the next section, "Displaying a Scrollable Help Screen," for more information on Type C frames.

Dialog Box Items

You can provide help for the various items in a dialog box. Help is available by pressing F1 on any item you can tab to: text boxes, radio buttons, check boxes, list boxes, and pushbuttons. You set this up by including a frame in the help file with a label in the following format:

{{#,*keyword id*}}←

# indicates the frame type.

keyword is the name of the menu frame in the menu file that includes the field.

id is the number of the dialog box item for which you want to provide help. Do not include a space between *keyword* and *id*.

For example, the following label designates the beginning of a Type C frame in the help file that provides information on item 103 in the menu frame with the keyword *print*:

```
{{C,print103}}←
```

Result: When the user puts the cursor on the item and presses F1, XyWrite displays the information contained in the help frame.

Pull-Down Menus and Other Screens

To link context-sensitive help screens for Type 5, Type C, and Type L frames, in the help file create a Type C frame that has the same keyword as the frame in the menu file you want to chain it to. Add an @ character to the keyword in the new frame label.

For example, a help frame with the following keyword label in the help file provides information on the menu frame with the keyword *GHelp*:

```
{{C,GHelp@}}←
```

To provide help on an item in a pull-down menu (Type L), use the keyword inside the JD command from the menu file as the keyword for a frame in the help file followed by @.

For example, in the menu file the following item appears as part of a Type L menu frame:

```
&New... « JDNew» ,Create a new file
```

To provide information on that item, include a help frame in the help file with the keyword *New@*.

Command Markers

You can use the ST (Show Triangle) function call and a Type C help frame in the menu file to provide full explanations of command markers. When the user puts the cursor on a marker in a text file and executes the ST (Show Triangle) function call, XyWrite displays a frame (if one exists with the proper keyword in the menu file).

The help frame must reside in the menu file and have as one of its keywords the two-letter embedded command mnemonic preceded by a ^.

For example, the following Type C label has the keyword *OF* and can be used to provide help for OF (Offset) command markers:

```
{{C,^OF}}←
```

To make the function easier to use, assign ST to a key in the keyboard file. For example, the following entry in the keyboard file assigns the ST function call to key 35 (H):

```
35=XH,ST←
```

Result: When the user puts the cursor on an OF command marker in a text file and presses Alt Shift H, XyWrite displays the information contained in the Type C frame with the keyword *OF*.

Displaying a Scrollable Help Screen

INTRO

You use a Type C frame in the help file to display a help screen that you can scroll vertically. You can link a Type C frame to other help frames to replace the currently displayed screen; for example to display a related topic.

To replace the currently displayed screen, the user puts the cursor on a bold keyword and presses ↵.

When the screen is displayed, the cursor moves to any bold or underlined keyword visible. When the user presses cursor keys, the cursor moves from visible keyword to visible keyword.

Keyword Label

A Type C frame can consist of any number of lines. The first line contains the label (C, followed by one or more keywords) enclosed in a pair of double curly braces {{ }}.

The second line can contain a DGW (Dialog Window) command to set the size of the window, followed by any formatting commands you want to use. Refer to "Positioning Dialog Box or Help Screens" for more information.

Displayable Lines

Each subsequent line contains text you want displayed on the screen. To link the frame to another frame, include a word either in bold or in underline mode somewhere in the text and use the JD (Jump to Dialog Box) command with the other frame's keyword to link the two frames.

NOTE #1 **JD Command Position.** The JD command must appear immediately *after* the linking word and *before* the MDNM (Mode Normal) command.

NOTE #2 **Bold Mode.** You can use bold mode throughout the frame for emphasis.

For example, the following Type C frame in XY4.HLP has the label, "ghGetstart."

Displaying a Scrollable Help Screen (cont'd)

```
{{C,ghGetstart}}
«DGW05,04,70,19»  «MDBO»< Exit >«JDExHelp»«MDNM»
«MDBO»< Index >«JDIdxHlp»«MDNM»
«MDBO»< Keys >«JDKeyHlp»«MDNM»
«MDBO»< Previous >«JD»«MDNM»
------------------------------------------------------------
Getting started help topics

   «MDBO»System requirements«JDS49P» «MDNM»
   «MDBO»Overview«JDOverview» «MDNM»
   «MDBO»Using XyWrite help system«JDAbouthlp» «MDNM»
   «MDBO»Starting XyWrite«JDEditor» «MDNM»
   «MDBO»Running programs under XyWrite«JDD17P» «MDNM»
```

If the user puts the cursor on "Exit" and presses ↵, XyWrite executes the subroutine, ExHelp. If the user puts the cursor on "System Requirements," XyWrite displays the frame with the keyword *S49P*.

Including Text in Screen Borders

INTRO

When XyWrite constructs dialog box and help screens from information in a Type 5 frame with displayable lines or a Type C frame, it automatically includes a border for screen display. You can use the DGT (Menu Screen Top) command to automatically include text in the top of the border and the DGB (Menu Screen Bottom) command to automatically include text in the bottom of the border.

You specify border text for an individual dialog box or help screen by using DBT or DGB on the second line of the frame, immediately following the label.

When you type either command on the command line and press ↵, a command window opens on the screen. Type the text exactly as you want it to appear in the border, and then press F11 to close the command window and complete the procedure.

For example, the following includes the text "How to Find a File" in the top border of a Type C frame that discusses finding files:

```
{{C,find}}←
«DGTHow to Find a File»←
```

You can specify default text that you want included in the border of every help screen by using the DGT or the DGB command on the second line of the help file. For example, the following includes the text "Press the Return key to continue," in the bottom border of all the help screens generated by the file:

```
;HL;←
«DGBPress the Return key to continue»←
```

Displaying Detailed Explanations of Error Messages

INTRO

You use a Type 9 frame in the help file to activate the XyWrite feature for displaying a full screen of information about the last error message the user received. It is another way to access frames by using XyWrite's error coding system. A single Type 9 frame in the help file enables XyWrite to display the Type C frame that contains in its label the keyword number associated with the last error message displayed.

Keyword Label

A Type 9 frame consists of only one line which contains the label (9 followed by one or more keywords) enclosed in a pair of curly braces {{ }}. For example, in XY4.HLP the Type 9 frame looks something like this:

```
{{9,last error message,^F1}}←
```

In the example, the *9* indicates that the frame is a Type 9. *last error message* and *^F1* are the keywords associated with the frame.

Displayable Lines

You use a Type C frame to enter the message you want displayed for one or more errors, and use one or more internal XyWrite error numbers as keywords. Add leading zeros to error numbers less than three digits (for example, 006 or 032). You can scroll through the XY4.HLP file for many of the current error numbers. (All the frames that describe errors are grouped under the Type 9 frame label.)

For example, the XyWrite error message, "There is no command on the command line," is error message number 11. If the user presses Ctrl F1 when XyWrite generates that error message, XyWrite displays a full help screen describing the problem. XyWrite constructs the displayed screen from a Type C frame in XY4.HLP with the following label:

```
{{C,011}}←
```

In the label, the C indicates that the frame is a Type C and *011* is the internal number of the error message.

You can have XyWrite automatically display your error help screens whenever an error occurs. Use the Default EH=1 to activate the feature; EH=0 to disable it. (Although this feature is helpful for beginners, it can slow the performance of more experienced users.)

TIP

Building a List of Error Messages. The VA \nnn command returns the error message associated with error number *nnn*. You can use this command to create a macro that builds a complete list of error messages.

Testing Dialog Boxes and Help Frames

INTRO

XyWrite has several functions to assist you in debugging dialog box or help files.

- ?d and ?h test commands
- Quick Help (QH) function call
- Quick Print (QP) function call

?D and ?H Test Commands

You can quickly display the currently loaded menu, dialog box, or help frames by using the ?m command for dialog box frames or the ?h command for help frames. These commands have the following format:

keyword ?d or
keyword ?h

keyword is the keyword of the frame you want to display.

For example, to display the help frame with the keyword *credits*:

F5credits ?hF9

The QH (Quick Help) Function Call

QH lets you test frames without saving and reloading the menu or the help file.

The QH function call is assigned to Ctrl Alt ShiftQ in the XyWrite keyboard file.

To use QH, position the cursor in the frame you want to test and press Ctrl Alt ShiftQ. XyWrite immediately displays the associated screen.

NOTE

QH Not Useful with All Frames. QH does not work with Type 0, Type C, or Type L frames.

The QP (Quick Print) Function Call

QP prints the dialog box associated with the frame in which the cursor is located in a displayed dialog box or help file.

To use QP:

1. Position the cursor in the frame you want to test.

2. Type: F5func qp↵

XyWrite prints the dialog box as it will appear on the screen.

Accessing a Menu Screen with One Keystroke

INTRO

You use the JM (Jump to Menu) function call in the keyboard file to access Type K or Type 5 frames directly from the keyboard.

In the keyboard file, JM has the form:

```
JM,(,k,e,y,w,o,r,d,)
```

k,e,y,w,o,r,d is the keyword of the menu frame or subroutine. The keyword must appear in parentheses and each letter of the keyword must be separated by a comma.

In the XyWrite keyboard file, all the JM function calls are preceded by XH (Exit Help) commands to remove any currently displayed help, dialog box, or menu screens before jumping to the frame.

For example, the following notation in the CTRL Table of the XyWrite keyboard file assigns the Open dialog box to key 24 (O):

```
24=XH,JM,(,O,p,e,n,)
```

When the user presses Ctrl O, XyWrite first removes any help, dialog box, or menu screen displayed and then displays the dialog box that has the keyword *Open*.

Accessing a Help Screen with One Keystroke

INTRO

You use the JH (Jump to Help) function call in the keyboard file to access Type C frames directly from the keyboard.

In the keyboard file, JH has the form:

```
JH,(,k,e,y,w,o,r,d,)
```

k,e,y,w,o,r,d is the keyword of the help frame you want to access. The keyword must appear in parentheses and each letter of the keyword must be separated by a comma.

In the XyWrite keyboard file, all the JH function calls are preceded by XH (Exit Help) commands to remove any currently displayed help, dialog box, or menu screens before jumping to the frame.

For example, the following notation in the CTRL Table of the XyWrite keyboard file assigns the help frame with the keyword ^F1 to key 59 (F1):

```
59=XH,JH,(,^,F,1,)
```

When the user presses Ctrl F1, XyWrite first removes any help, dialog box, or menu screen displayed and then displays the help frame that has the keyword *^F1*.

Loading Menu, Dialog Box, and Help Files

INTRO

Use the LOAD command to load your menu, dialog box or help file as you would any other customization files. For example, to load a menu file named MY.MNU and a dialog box file named MY.DLG:

Type: F5 load my.mnu+my.dlg↵

If you haven't entered the special file label (;MN; or ;HL; or ;DG;) as the first line of the file you are loading, XyWrite displays the error message, "Customization file requires a file label."

Since menu, dialog box, and help files can become very large, XyWrite never keeps the whole file in memory. The first time you load a menu, dialog box, or Help file, XyWrite creates an internal index of the file and stores the index at the bottom of the file. This index, and certain other types of frames, are all that actually load into memory, which allows a very large menu, dialog box, or help file to consume very little memory. When you access the menus or help facility, XyWrite checks the index, much as you check the index of a book, to determine where to get the information you want displayed. It then refers back to the file stored on disk to get the information to display on the screen.

If you load the same unchanged menu, dialog box, or help file again, XyWrite does not rebuild the index, so the file loads very quickly. Whenever you edit a menu, dialog box, or help file and store it back to the disk, the index is removed. The next time you load the file, XyWrite rebuilds the index. Therefore, loading times vary depending on whether or not the menu, dialog box, or help file was edited.

Although a menu, dialog box, or help file can be any size that fits on a disk, the index cannot exceed 64K.

Printer Files

Intro

For the most part, printer manufacturers have adopted different control codes. As a result, software manufacturers have had to develop ways of handling printer information. We have developed a library of printer files that translate XyWrite commands into codes that the printer understands.

You may never need to modify your printer file, but this section gives you the information you need if you do want to customize it.

Contents

Basics of the Printer File

PURPOSE

A printer file is a file that contains the settings and control codes for a particular printer. XyWrite comes with printer files for over 150 specific printers. These files are ready to use—all you need to do is see which file applies to your printer, and then load that file with SETP or LOAD.

As part of the installation procedure, you are asked to identify the printers that are part of your current setup. The installation procedure then copies the appropriate printer files into the XY4 directory. To see the names of the printer files available on your computer, display a selective directory of XY4. (All XyWrite printer files have the extension PRN.)

Type: F5dirl \xy4*.prn↵

To find out which printer model a particular printer file supports, call up that file—the manufacturer's name and model numbers are listed at the top of each printer file.

ACTION

Making Changes in the Printer File
To make changes to the printer file, do the following:

1. Call to the display the printer file you use. For example:
   Type: F5call post35n.prn↵
2. Search through the printer file to see if the command you want to enter is already present.
3. Modify or add the commands you want.
4. To cause the setting you entered in the previous step to take effect, store the printer file and then load it as described in the next procedure.

NOTE

Testing Changes. We recommend that you save and load the printer file after each change so you can easily identify and correct errors.

ACTION

Loading the Printer File

When you install XyWrite, a SETP command is entered in STARTUP.INT to load the default printer file and establish the port. If you modify a printer file, you can reissue the SETP command to load the changes into memory. For example:

Type: F5setp 1↵

to load the printer file associated with the first printer listed in the Select Printer file dialog box.

If you do not have printers connected to different ports, you can use the LOAD command to load a printer file into memory. For example:

Type: F5load post35n.prn↵

Result: This loads your printer file into memory, setting up XyWrite to print to a PostScript printer. It does not redirect printing to a different port.

ACTION

Installing a Serial Printer

When installing a printer that runs off your serial port (COM1 or COM2), you must redirect the line printer port (LPT1) to the serial port. To do this, perform the following steps prior to starting XyWrite:

1. Be sure that MODE.COM is in your root directory.
2. Type the following two statements *prior* to starting XyWrite. Refer to your printer manual for the proper settings to include—here we've used 9600,n,8,1,p. The second statement redirects LPT1: to COM1. At the DOS prompt:

   Type: mode com1:9600,n,8,1,p↵

   Type: mode lpt1:=com1↵

   If the serial printer is part of your standard configuration, you can add these two statements to your AUTOEXEC.BAT file. (Refer to the DOS manual for more information on the mode command.)
3. Start XyWrite. At the DOS prompt:

   Type: editor↵

   At this point you are in XyWrite, where you can call, edit and store files.
4. After you quit XyWrite, issue the following command to return LPT1 and COM1 to their original states. At the DOS prompt:

   Type: mode lpt1:↵

How Do Printer Files Work?

When you write a document, you can indicate in text where you want different fonts (such as Times), attributes (such as underlining), orientation (such as landscape), or graphics. The printer file specifies exactly how these (and other) qualities are handled by the printer.

If you have loaded a PostScript printer file and you want text to be printed in Times Roman, you insert «UFTIMES» (Use Font Times) into the text. Then, when you print a document using PRINT, XyWrite inserts the printer codes assigned to the Times font into the text going to the printer. These codes are specified by a line in the printer file; for example:

```
FO<) s /Times-RomanS findfont [?X 0 0 ?Y 0 0] m (
```

Let's say you have loaded a PostScript printer file, and you send a file containing Times Roman text to your printer. The printer codes listed in the FO< statement above are inserted into the text in place of the embedded command UF TIMES. This causes Times Roman to begin. When XyWrite encounters the next UF command, it inserts the printer codes associated with the typeface specified in that command.

TERMINOLOGY

The printers supported by XyWrite fall into three categories: scalable printers, non-scalable printers, and Hewlett-Packard LaserJet-compatible printers.

scalable printer — A printer that uses character outlines rather than point-size-specific (bitmapped) characters. The outlines can be reduced or enlarged to produce a wide range of type sizes. PostScript printers (e.g., Apple LaserWriter NTX, IBM Personal PagePrinter, and QMS PS 810) are a common example of scalable printers. The Hewlett Packard LaserJet III and Canon LBP4 are also scalable printers.

non-scalable printer — A printer that uses point-size-specific (bitmapped) characters. Non-scalable printers support a more limited range of point sizes than scalable printers. Dot matrix printers (e.g., Epson LX 800) and LaserJet Series II printers are common examples of non-scalable printers.

Hewlett-Packard LaserJet printers (and compatibles) — A series of printers that use the PCL control language. LaserJet printers support a large character set, which is divided into a series of symbol sets.

The following terms apply to type specifications. The meaning of these terms varies slightly, depending on the printer. For example, bold may be a font on one printer but an attribute on another printer.

attribute — Any printer effect that modifies an existing font, such as underline, superscript or subscript.

font — A set of characters that have a unique type design (e.g., Pica, Elite, Courier or Helvetica). In XyWrite, *font* is a synonym for *typeface. There are two major categories of printer fonts: bitmapped fonts* and *outline fonts.*

A bitmapped font is one in which each character is defined as an array of dots. Bitmapped fonts are point-size specific, which means you can only print in the point sizes that are internal to the printer.

An outline font is one in which the characters are defined as shapes. Outline fonts can be scaled to any point size.

TERMINOLOGY (cont'd)

leading (pronounced lĕd-ing) — The vertical spacing of lines of type, measured from baseline to baseline.

_The rain in Spain
LEADING ↑falls mainly on the plain

point — A unit of measure frequently used to define type size and leading. Equal to approximately 1/72 inch.

point size — The height of a character. You can gauge the point size of a character approximately by measuring from the top of an uppercase letter to the bottom of a "y" or "g" and adding for white space above and below it.

8	ABCDEFGHIJKLMNOPQRSTUVWXYZ
10	ABCDEFGHIJKLMNOPQRSTU
12	ABCDEFGHIJKLMNOPQ
14	ABCDEFGHIJKLMNO
18	ABCDEFGHIJKL
24	ABCDEFGHI
30	ABCDEFG
42	ABCDE
48	ABCD
60	ABC

typeface family — A group of related typefaces (e.g, Times, Times Bold, Times Italic, Times Bold Italic).

Printer File Construction

The printer files provided on your XyWrite disks are already properly set up to work. However, if you want to *modify* your printer file, it would be good to understand how it has been constructed. Look at the printer file example on the next page. Although this is a condensed version of an actual printer file, it shows the 14 most common elements of a printer file. (Lines starting with a semicolon are comment lines. Vertical ellipses indicate areas where data has been omitted from the illustration.)

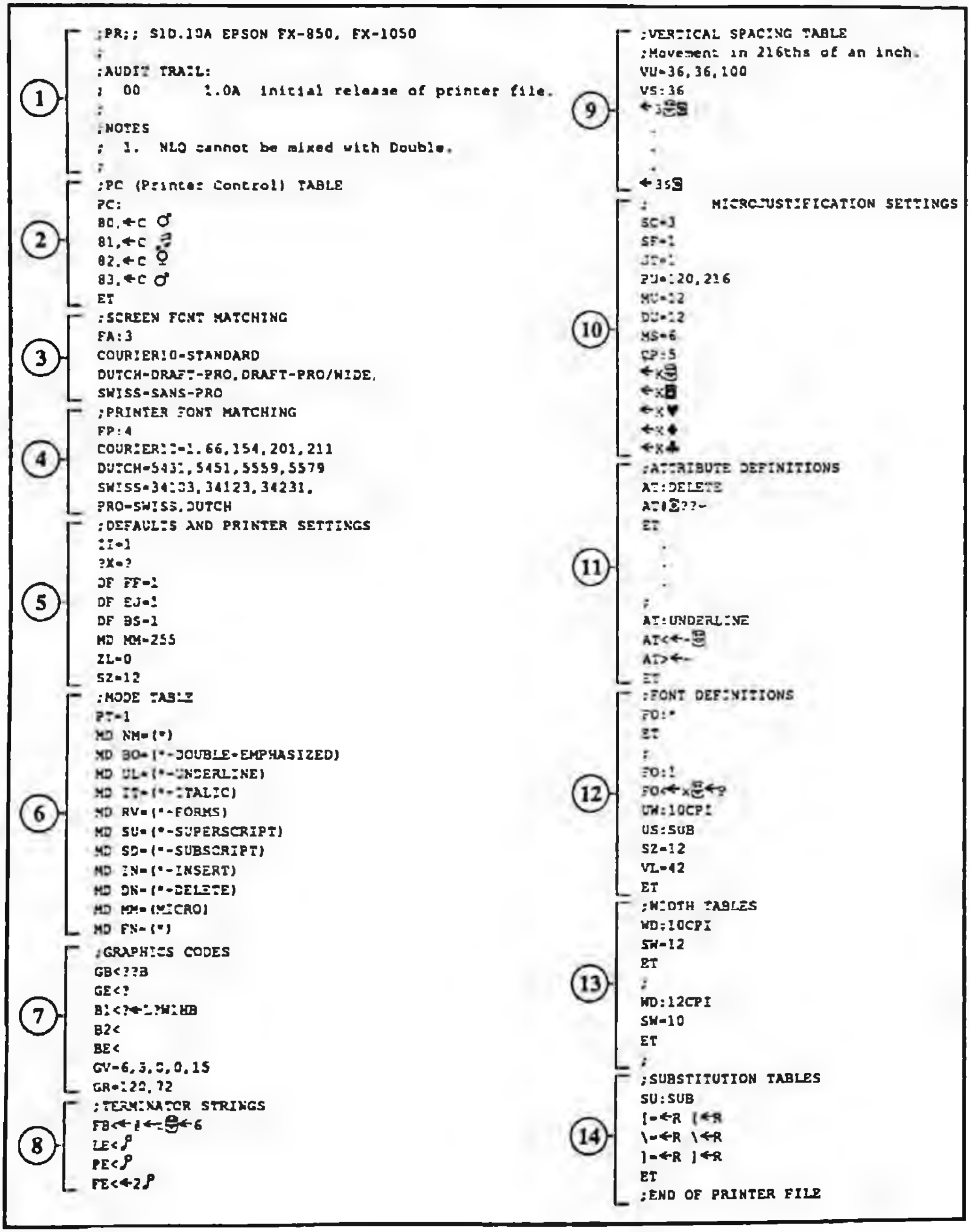
1. **Heading** — The file label, formatting commands, and notes.
2. **Printer Control Table** — Control codes for printer commands, such as the codes for changing paper trays.
3. **Screen Font Matching** — Associates font names in a UF command with screen fonts.
4. **Printer Font Matching** — Associates this printer's fonts with font names in a UF command.
5. **Defaults and Printer Settings** — Default settings that are printer-specific, such as the non-printable area, as well as other printer settings.
6. **Character Mode Table** — Associates character mode commands with attributes such as bold and underline.
7. **Graphics Codes** — Control codes for printing graphics and borders.
8. **Terminator Strings** — Characters to end a line, page, and file.
9. **Vertical Spacing Settings** — Control codes for vertical spacing.
10. **Microjustification Settings** — Control codes for horizontal spacing.
11. **Attribute Definitions** — Control codes for specific attributes.
12. **Font Definitions** — Definitions and control codes for specific fonts.
13. **Width Tables** — Width settings for fonts.
14. **Substitution Tables** — Control codes for specialized characters.

NOTE

Comment Lines. Any line starting with a semicolon (;) is a comment. The *entire line* is treated as a comment — you cannot place a comment to the right of a printer setting. The line is ignored when the printer file is loaded into memory. For example:

```
;This is a comment line.←
```

You can place comments throughout the printer file as you wish, except you cannot put comments in tables that have a specified number of lines (for example, VS:).



1
2
3
4
5
6
7
8
9
10
11
12
13
14

```
;PR:; S1D.10A EPSON FX-850, FX-1050
;
;AUDIT TRAIL:
;  00       1.0A  initial release of printer file.
;
;NOTES
;  1.  NLQ cannot be mixed with Double.
;
;PC (Printer Control) TABLE
PC:
80,←C ♂
81,←C ♫
82,←C ♀
83,←C ♂
ET
;SCREEN FONT MATCHING
FA:3
COURIER10=STANDARD
DUTCH=DRAFT-PRO,DRAFT-PRO/WIDE,
SWISS=SANS-PRO
;PRINTER FONT MATCHING
FP:4
COURIER12=1,66,154,201,211
DUTCH=5431,5451,5559,5579
SWISS=34133,34123,34231,
PRO-SWISS,DUTCH
;DEFAULTS AND PRINTER SETTINGS
II=1
?X=?
DF FF=1
DF EJ=1
DF BS=1
MD MM=255
ZL=0
SZ=12
;MODE TABLE
PT=1
MD NM=(*)
MD BO=(*-DOUBLE-EMPHASIZED)
MD UL=(*-UNDERLINE)
MD IT=(*-ITALIC)
MD RV=(*-FORMS)
MD SU=(*-SUPERSCRIPT)
MD SD=(*-SUBSCRIPT)
MD IN=(*-INSERT)
MD DN=(*-DELETE)
MD MM=(MICRO)
MD FN=(*)
;GRAPHICS CODES
GB<??B
GE<?
B1<?←L?W1HB
B2<
BE<
GV=6,3,0,0,15
GR=120,72
;TERMINATOR STRINGS
FB<←@←-▤←6
LE<♪
PE<♪
FE<←2♪
```

```
;VERTICAL SPACING TABLE
;Movement in 216ths of an inch.
VU=36,36,100
VS:36
←3▤▤
 .
 .
 .
←3▤
;         MICROJUSTIFICATION SETTINGS
SC=3
SF=1
JT=1
PU=120,216
MU=12
DU=12
MS=6
CP:5
←K▤
←K■
←K♥
←K♦
←K♣
;ATTRIBUTE DEFINITIONS
AT:DELETE
AT#▤??-
ET
   .
   .
   .
;
AT:UNDERLINE
AT<←-▤
AT>←-
ET
;FONT DEFINITIONS
FO:*
ET
;
FO:1
FO<←x▤←P
UW:10CPI
US:SUB
SZ=12
VL=42
ET
;WIDTH TABLES
WD:10CPI
SW=12
ET
;
WD:12CPI
SW=10
ET
;
;SUBSTITUTION TABLES
SU:SUB
[=←R [←R
\=←R \←R
]=←R ]←R
ET
;END OF PRINTER FILE
```

INTRO

The heading section of the printer file includes the file label, an audit trail, and notes. The size of the heading section varies from printer to printer, and most of the data it contains is for information only. The exception to this is the file label, which is required by XyWrite. Each line in the heading section is preceded by a semicolon, indicating that it is a comment.

LABEL

The first thing in every XyWrite printer file is the following label: =LA437

```
;PR«cp»;
```

LA437 / LA850*/none

where *cp* is the number of the code page (437 or 850) used to create the printer file. Most XyWrite 4.0 printer files created in code page 850. This label, which must be in uppercase, identifies the file as a printer file when you load it into memory. Immediately after the file label are formatting commands, the version number of the file, and the types of printers supported by the file.

AUDIT TRAIL

The audit trail describes the changes made to the printer file each time a new version is released.

NOTES

The notes area contains information accumulated by XYQUEST's Technical Services Department that will help you use the printer with XyWrite. It may include switch settings, sheet feeder information, memory requirements, etc.

*) with ext. char. support

Printer Control Table

INTRO

The PC (Printer Control) table allows you to send control strings to the printer. You use this table to control the printer at specific points within your document and to set printer defaults. For example, if your printer has two paper trays, you can change trays when you begin a new page. You could use letterhead paper for the cover page of a letter and plain paper for subsequent pages.

In XyWrite, a given PC number performs the same function on all printers. For example, PC 1 always turns on draft printing, and PC 141 always establishes landscape printing. The codes in the PC table are set up in groups. The first code in the group represents the default condition. Within each group, only one function can be turned on (for example, only one paper tray can be active). Refer to the table on the next page for a list of the PC numbers and associated printer functions.

Although the function associated with each PC number is the same from printer file to printer file, the content of the PC table varies widely from printer file to printer file. There are two reasons for the variation: (1) individual printers do not support all the functions defined in the master table; and (2) different printer manufacturers use different control codes to perform functions.

The format for the PC table is:

```
PC:
#,code
.
.
.
#,code
ET
```

where # is the PC number and *code* is the printer-specific control codes for that function.

LQ

1, Draft
2, Letter
3, Letter II
4, Letter III
5, NLQgothic
6, NLQcourier
7, Utility
9, Draft II

EF

10, Reverse Off
11, Reverse On
12, Outline Off
13, Outline On
14, Shadow Off
15, Shadow On
16, Inverse Off
17, Inverse On
18, User Set Off
19, User Set On
20, Reserved
21, Reserved
22, Reserved
23, Reserved
24, Reserved
25, Reserved
26, Script Up Off
27, Script Up On
28, Script Down Off
29, Script Down On
30, Double Underline Off
31, Double Underline On
32, Overscore Off
33, Overscore On
34, Floating Underline Off
35, Floating Underline On
36, Outline/Shadow Off
37, Outline/Shadow On
38, Shading Off
39, Shading On

DY

40, Black
41, Blue
42, Brown
43, Charcoal
44, Cyan
45, Green
46, Magenta
47, Maroon
48, Neutral
49, Olive
50, Orange
51, Red
52, Violet
53, White
54, Yellow
60, Black Background
61, Blue Background
62, Brown Background
63, Charcoal Background
64, Cyan Background
65, Green Background
66, Magenta Background
67, Maroon Background
68, Neutral Background
69, Olive Background
70, Orange Background
71, Red Background
72, Violet Background
73, White Background
74, Yellow Background

PB

80, Letter (8.5"x11")
81, Legal (8.5"x14")
82, A4 (210x297mm)
83, Executive (7.25"x10.5")
84, B5 (182x257mm)
85, Envelope #10 (4.13"x9.5")
86, Envelope #9 (3.88"x8.88")
87, Monarch Envelope
88, DL Envelope (110x220mm)
89, C5 Envelope (162x229mm)
90, B5 Envelope (176x250mm)

Printer Control Table (cont'd)

PB	120, Upper Tray (Bin 1)/Sheet feeder off
	121, Manual Feed
	122, Manual Envelope (Any Size)
	123, Lower Tray (Bin 2)/Sheetfeeder On
	124, Envelope Tray/Feeder
	125, Paper Bin 3
	126, Paper Bin 4
	127, Paper Bin 5
	128, Paper Bin 6
	129, Paper Bin 7
OR	140, Portrait
	141, Landscape
	142, Reverse Portrait (right-to-left)
	143, Reverse Landscape (right-to-left)
PC	148, Pluggable Font Card Off
	149, Pluggable Font Card 1
	150, Pluggable Font Card 2
	151, Pluggable Font Card 3
	152, Pluggable Font Card 4
	153, Pluggable Font Card 5
	154, Pluggable Font Card 6
	155, Pluggable Font Card 7
	156, Pluggable Font Card 8
	157, Pluggable Font Card 9
PC	158, Overlay Off
	159, Overlay 1
	160, Overlay 2
	161, Overlay 3
	162, Overlay 4
	163, Overlay 5
	164, Overlay 6
	165, Overlay 7
	166, Overlay 8
JB	170, Simplex (single-side)
	171, Duplex (long-edge)
	172, Duplex (short-edge)
PC	176, Transparency Off
	177, Transparency On
PC	178, Full Speed
	179, Half Speed
JB	240, Multiple Copies
PC	241, Shading Percentage
PC	242, Rotate Baseline Degrees
PC	243, Slant Character Percentage

As an example, the following table is from the printer file for the Epson FX850 printer:

```
PC:
80,←C ♂
81,←C ♫
82,←C ♀
83,←C ♂
120,←↓1
123,←↓2
ET
```

The first line (PC:) starts the table. The next four lines (labeled 80-83) are the codes for paper size (letter, legal, A4, Executive). The lines labeled 120 and 123 are the codes for paper trays. The last line (ET) ends the table.

Let's take a look at another example. The following PC table is from the Hewlett Packard LaserJet IIP printer file:

```
PC:
34,←&d@
35,←&d3D
80,←&l2A
81,←&l3A
82,←&l26A
83,←&l1A
85,←&l81A
87,←&l80A
88,←&l90A
89,←&l91A
120,←&l1H
121,←&l2H
122,←&l3H
123,←&l4H
124,←&l6H
140,←&l0O
141,←&l1O
240,?←&l?VX
ET
```

This PC table includes lines from five different code groups: underscore, paper size, paper tray, orientation, and multiple copies.

NOTE #1 **Reserved Numbers.** XyWrite uses a universal numbering scheme for printer codes. If you want to add a PC code to the table, use a number between 180 and 239.

NOTE #2 **Wild Cards.** Some printer functions defined in the PC table require a value in order to be executed. For example, PC 241 is the printer control code for shaded printing. In order to process this control code, XyWrite must pass the desired percentage of shading to the printer.

For this reason, PC codes 240-255 use a wild card in their definitions. The syntax of the wild card in the PC table is:

#,?code?Vcode

where # is the PC code; ? is the wild card indicator; *code* is the printer code for the function being defined; and V is the variable that will be replaced by the second argument of the PC command when you print your document.

NOTE #3 **How PC Codes Are Invoked.** PC codes are automatically invoked when you select a printer-related function from the menu. (Functions that are not supported by the currently loaded printer appear dimmed in the menus.) The menus execute one of the following commands, depending on the function:

JB (Job Begin)
PB (Page Begin)
PC (Printer Control)
LQ (Letter Quality)
EF (Special Effect)
DY (Dye)
OR (Orientation)

The JB, PB, and PC commands are described on the next page. Refer to the *Command Reference Guide* for more information about the LQ, EF, DY and OR commands.

Printer Control Commands

FORMAT

CMpb #
CMpc #,*v*
CMdefault jb=#,#,#...

is the number of the printer code.
v is a variable that is passed to the printer for PC codes 240-255.

PURPOSE

PB, PC, and JB all allow you to send printer codes defined in the PC table to the printer. However, each command processes the printer codes in a slightly different manner:

- JB (Job Begin) — This is a default command. It instructs XyWrite to process the specified PC code at the beginning of every document sent to the printer.

- PB (Page Begin) — This is an embedded command. It instructs XyWrite to process the specified PC code at the beginning of the page that contains it.

- PC (Printer Control) — This is an embedded command. It instructs XyWrite to process the specified PC code at the location of the command.

If you want to invoke one of the PC codes, it is important that you choose the appropriate command. For example, duplex printing needs to be established as a default (with JB), paper size needs to be established at the top of the page (with PB), and font card selection is generally established at the text location where you want the font card to take effect (with PC). The table of PC codes shown earlier in this section indicates the command you should use for each group of PC codes. (Remember, the PC codes can be invoked automatically via the menus.)

There are 25 groups of PC codes, plus four single codes at the end of the table. When you turn on one PC code in a group, the other codes in that group are automatically cancelled. PC codes invoked by the PC or PB commands are automatically reset at the end of a document; PC codes invoked by the JB default setting remain in effect until another JB command is issued (see Note #1).

Printer Control Commands (cont'd)

ACTION

Inserting a Printer Control String
Let's assume you are using a Hewlett Packard LaserJet IID printer and want to instruct the printer to use the lower paper tray.

1. Move the cursor to the page that you want to start printing from the lower paper tray.

2. Enter the PB command.

   Type: F5pb 123↵

Result: When you print the document, the code to switch to the lower paper tray is sent to the printer at the top of the page that contains the PB command. The printer continues to print from the lower paper tray until you enter another PB command from the same group of PC codes or until the end of the document.

ACTION

Sending a Variable to the Printer
Some PC codes (240 and above) require a value to complete the function. For example, if your printer supports a rotating baseline (code 242), you must specify the amount of the rotation. For example, to rotate 90 degrees:

Type: F5pc 242,90↵

NOTE #1

JZ Default Setting. The JZ default setting affects the way XyWrite resets the PB and JB commands. Refer to "Printer Settings" for information on the JZ setting.

Font Matching

INTRO

When you format a document, there are a number of type specifications in effect, even if you don't enter an embedded command for each of them. When you print your document (or display it in graphic view), XyWrite compares the type specifications in the document to the font tables in the printer file to determine what font most closely matches the current type specifications.

If you formatted your document using the menus, there will be an exact match between the type specifications in your document and the font tables in the printer file. However, if you received a XyWrite document that was formatted by someone else (perhaps for a different printer), XyWrite may not find an exact match.

When that happens, XyWrite looks for a font that most closely matches the type specifications. It makes the selection based on the following priorities:

1. **Orientation:** If the printer file supports portrait and landscape fonts, XyWrite matches on this specification first.
2. **Type Size:** From the list of fonts in the correct orientation, XyWrite looks for a font that is also in the requested type size.
3. **Typeface:** From the fonts that match the first two criteria, XyWrite looks for a font that is also in the requested typeface (e.g., Times or Courier). If there is no font table for the specified typeface, XyWrite looks at the FP (Printer Font) table for a typeface synonym, and then matches on that font table. (See "Printer Font Assignments" for more information on the FP table.)
4. **Type Style:** Type style (normal, bold, italic) is the next criteria XyWrite uses to match fonts.

The process of formatting a document for printing and formatting it for graphic view is the same. XyWrite uses the same font matching steps, with one modification. Instead of looking in the FP table for a font match, XyWrite looks at the FA (Font Association) table to determine what screen font to use to represent the printer font. (The printer font can be either the requested typeface or a synonym for the requested typeface, as described in Step 3 above.) See "Screen Font Assignments" for more information on the FA table.

NOTE

Hewlett Packard LaserJet Printers and Compatibles. If you are using a Hewlett Packard LaserJet printer or compatible, there are three additional type specifications that can be used for font matching: symbol set, pitch, and stroke weight. These specifications are evaluated after the typeface specification.

Screen Font Assignments

INTRO

When you display a document in graphic view, XyWrite uses screen fonts to create the display. XyWrite provides three styles of screen fonts: serif (DUTCH), sans serif (SWISS), and monospaced (COURIER10). While you may add more screen fonts (see Note #1), it is likely that your printer offers typefaces for which you do not have a corresponding screen font.

FA:

The Font Association table lets you associate each screen font with as many printer fonts as you want. You use this table to match the printer font to the most closely matching screen font (for example, to match the printer's serif typefaces with the DUTCH screen font).

Format:

```
FA:n
scrfnt=prfnt1,prfnt2,...
scrfnt=prfnt1,prfnt2,...
scrfnt=prfnt1,prfnt2,...
```

where *n* is the number of lines in the FA table; *scrfnt* is the name of the screen font; and *prfnt1*, *prfnt2* are the names of the printer fonts supported by this printer.

Example:

```
FA:3
courier10=courier,standard
dutch=bookman,newcenturyschlbk,zapfchancery,palatino,times
swiss=avantgarde,helvetica,helvetica-narrow
```

In this example, the screen font COURIER10 is used to represent printer fonts Courier and Standard; screen font Dutch represents printer fonts Bookman, New Century Schoolbook, Zapf Chancery, Palatino, and Times; screen font Swiss represents printer fonts Avant Garde, Helvetica, and Helvetica Narrow.

When you switch to graphic view, XyWrite checks the FA setting to determine what screen fonts you want to have represent the printer fonts selected in your file. It uses the widths of the printer fonts, so the line endings will be exactly as they will appear when printed. However, when a representative screen font is used, the screen display is not exactly a what-you-see-is-what-you-get display (see Note #2).

NOTE #1 **Screen Fonts.** XyWrite uses screen fonts in Bitstream Speedo format. There are many fonts available in this format from third-party vendors. To use them with XyWrite, install the new fonts in the directory set up for screen fonts (by default, this is the BTFONTS directory, which is off the root directory). Then delete the file SCRFONTS.BIN. XyWrite will recreate it the first time you switch to graphic view.

NOTE #2 **Screen/Printer Font Mismatch.** If the printer font widths differ significantly from the screen font widths, you may see overlapping characters in graphic view. Depending on the setting of the EP (Error Prompt) default, you may also see the message "Screen/printer font mismatch--display is incorrect." No action is required in response to this message—it is for your information only. When you print the document, the character alignment will be accurate. (For more information on EP, refer to "Default Settings.")

NOTE #3 **DCA Typefaces.** The Font Association tables in the XyWrite printer files include associations for DCA typefaces. These typefaces, which are represented by numbers, are used in IBM's standard document format called Document Content Architecture (or DCA).

NOTE #4 **Hewlett Packard PCL Fonts.** XyWrite automatically recognizes Hewlett Packard PCL bitmapped screen fonts. You do not need to include them in an FA table.

Printer Font Assignments

INTRO

Whether you are using a stand-alone machine or are on a network, chances are there will be times when you want to print a document on a different printer model than the one you normally use. Although different printer manufacturers use different typeface names, XyWrite printer files are designed to let you change printers without modifying the UF (Use Typeface) commands in your documents.

FP:

The FP table lets you associate one printer's font names with another printer's font names. This is handy if you use more than one printer, because you can output the same file on both printers without changing the file's UF commands.

Format:

FP:*n*
font=synonym1,synonym2,synonym3...

where *n* is the number of lines in the table, *font* is the name of the font used in this printer file and *synonym1, synonym2, synonym3...* are the names of the fonts used in alternate printer files.

Example:

```
FP:3
courier=elite
times=tmsrmn
helvetica=helv
```

When this printer file is loaded, XyWrite uses the printer font TIMES whenever it encounters the command UFTMSRMN in your documents; it uses the printer font COURIER when it encounters a UFELITE command; and it uses HELVETICA when it encounters a UFHELV command.

If you work in graphic view, you should also include the screen fonts in the FP table. The screen fonts need to be defined as *fonts* (for printing in image mode) and as *synonyms* (for printing in non-image mode from graphic view). For example:

```
FP:6
courier10=courier,standard
dutch=bookman,newcenturyschlbk,zapfchancery,palatino,times
swiss=avantgarde,helvetica,helvetica-narrow
courier=elite,courier10
times=tmsrmn,dutch
helvetica=helv,swiss
```

NOTE #1 **DCA Typefaces.** The FP tables in the XyWrite printer files include associations for DCA typefaces. These typefaces, which are represented by numbers, are used in IBM's standard document format called Document Content Architecture (or DCA).

NOTE #2 **Internal Fonts vs Outline Fonts.** When constructing an FP table, always list the outline fonts at the top and the internal fonts at the bottom.

NOTE #3 **Customizing FP Tables.** As delivered, the XyWrite printer files are not set up to share font names between printers—the number of potential combinations is too great. You can use the menus to customize the FP table to associate the font names used by the printers in your particular configuration.

NOTE #4 **Troubleshooting.** Let's say you installed a new Bitstream Speedo screen font in the \BTFONTS directory according to the directions provided with the font, and you deleted SCRFONTS.BIN so XyWrite could rebuild it with the new screen font information. Then you formatted a document using the new screen font and switched to graphic view. Instead of displaying the document in the new screen font, XyWrite displays the file in COURIER10. Why? When XyWrite goes into graphic view, it checks the FA and FP tables in the printer file to see what screen fonts are defined. If the screen font is not defined, XyWrite uses the default font (STANDARD), which is associated with screen font COURIER10.

To correct the problem, add the new screen font to the FA and FP tables according to the procedures described earlier. After making the changes, reload the printer file and then redisplay your document in graphic view.

NOTE #5 **Image Mode Printing.** Unless you are using a Hewlett Packard LaserJet or compatible, you need to set the IM (Image Mode) default to 1 in order to print screen fonts. XyWrite automatically downloads the fonts to Hewlett Packard LaserJet printers, making image mode unnecessary. Refer to "Default Settings" for more information about IM.

Default Settings

INTRO

Overview of Default Settings. All of the XyWrite default settings can be included in the printer file. Typically, though, only the defaults that are associated with the printer are included (the others are stored in SETTINGS.DFL). For example, some printers can perform a backspace while other printers cannot. The BS (Backspace) default setting is therefore set in the printer file.

When you set a default in the printer file, the format is:

DF *mn*=#

where *mn* is the two-letter name of the command and # is the value of the command. For example:

DF BS=1

DF BS

Backspace Control — Enables your printer to print a backspace, whether or not your printer can recognize a backspace character. (The initial default is 1.)

df bs=1 Use this setting if you are using a printer that can perform a backspace — in other words, if the printer responds to the backspace character—◘, ASCII value 8—by backspacing one character.

df bs=0 Use this setting if the printer *cannot* perform a backspace (that is, if it ignores the backspace character). Then, when XyWrite encounters a backspace character, it simulates the backspace function by printing that line in two passes.

DF EJ

Eject Last Page — Enables XyWrite to eject the very last page of a document. (The initial default is 1.)

df ej=1 Causes the very *last* page of a document to automatically eject from the printer. If Form Feed is off (DF FF=0), it sends blank lines to eject the last page. If Form Feed is on (DF FF=1), it sends a form feed character (♀) to eject the last page. Most laser printers require this setting.

df ej=0 Prevents the *last* page of a document from being ejected from the printer.

If there is a Running Footer command in the document, the last page will be ejected regardless of what the EJ setting is.

DF FF

Form Feed Character — Enables XyWrite to insert a form feed character at the end of each page. (The initial default is 1.)

df ff=0 Causes XyWrite to send enough carriage return/line feed characters at the end of each page to advance the paper to the top of the next page. The FD (Form Depth) setting determines the total number of lines XyWrite will send to the printer for each page.

df ff=1 Causes XyWrite to send a carriage return/line feed combination (♪◙), and a form feed character (♀), at the end of each page, to advance the paper to the top of the next page. Exception: when you turn off Eject Last Page (with DF EJ=0), there is no Form Feed character sent at the end of the *last* page. DF FF=1 is required on most laser printers.

Printer Settings

INTRO

Printer settings are similar to default settings, but they must be included in the printer file. You cannot set them from the command line as you can default settings, nor should you put them into SETTINGS.DFL. When you load a file that contains a printer setting, XyWrite doesn't just overwrite the value for the specific setting in memory; it unloads all previously loaded printer information from memory.

The following paragraphs describe the printer settings.

II

Italic Information. Establishes whether italic is a font or an attribute on this printer.

ii=0 Use this setting if the printer includes italic typefaces. (This is the default.)

ii=1 Use this setting if the printer uses an italic attribute. This setting is most commonly applied to dot matrix printers, and it enables the italic typestyle in the menus.

JZ

Job End — Specifies how XyWrite resets the JB (Job Begin) and PB (Page Begin) commands when you print a job. (Refer to the section "Printer Control Table" for a detailed explanation of the JB and PB commands.)

jz=0 Resets PB to 0 at the end of the job; does not reset JB.

jz=1 Resets JB and PB to 0 at the end of the job.

jz=2 Resets PB to 0 at the end of any page that contains a PB; does not reset JB.

jz=3 Resets PB to 0 at the end of every page. Resets JB to 0 at the end of the job.

NA

Non-Printable Area. Defines the page margin that is reserved by the printer and is therefore not available. For example, most laser printers do not start printing at the top of the page, nor do most dot matrix printers with sheet feeders.

na=*l,r,t,b* Format of the NA setting, where *l* is the non-printable area on the left of the page; *r* is the non-printable area on the right of the page; *t* is the non-printable area at the top of the page; *b* is the non-printable area at the bottom of the page.

na=.2in,.3in,.5in,.3in Example of the NA setting.

PM

Printer Memory. This setting gives XyWrite a variety of data about the printer's font-handling abilities.

pm=*m,f,c,o,v,s* Format of the PM setting, where *m* is the amount of memory available in the printer (in bytes); *f* is the maximum number of soft fonts that can be downloaded; *c* is the maximum number of cartridges the printer can hold; *o* is 1 or 0 (1 means add orientation to FO string or means that the orientation command is part of the FO string; 0 means don't add the orientation); *v* is the percentage by which you want XyWrite to increase the internal leading value of the soft fonts; *s* is the starting download position of printer fonts (the initial setting is 1000).

pm=1395,20,2,1,100,1000 Example of the PM setting.

PR

Printer Font Data. Specifies the file that contains the binary font information for internal and cartridge fonts and the model of printer that the fonts are for. At the current time, this setting applies to Hewlett Packard LaserJet printer files only, and there are two files that contain the font information: LJ.BIN and LJ3.BIN. LJ.BIN applies to all LaserJet printers except the Series III.

pr=_name,ty_ Format of the PR setting, where *name* is either LJ.BIN or LJ3.BIN and *ty* is one of the following labels:

LJ	LaserJet
L2	LaserJet II
2D	LaserJet IID
2P	LaserJet IIP
L3	LaserJet III
3P	LaserJet IIIP
3D	LaserJet IIID
SI	LaserJet IIISi

SB

Symbol Set. Defines the symbol sets that are displayed in the Typeface menu for Hewlett Packard LaserJet internal fonts.

sb=_s1,s2,...s10_ Format of the SB setting, where *s1, s2...s10* are the symbol set names.

sb=0u,10u,8u,7J,1u Example of the SB setting.

UO

Use Outline. Indicates whether or not a printer accepts downloaded fonts in Hewlett Packard format. (The initial default is 0.)

uo=0 Printer does not accept downloaded fonts in Hewlett Packard format.

uo=1 Printer does accept downloaded fonts in Hewlett Packard format.

XS

Substitution Character. Defines the character that will be printed when the requested character is not available. Only applies to characters over 255. (The substitution character defined by XS can be overridden for individual substitution tables. See "Substitution Tables" later in this section for more information.)

xs<_c_ Format of the XS setting, where *c* is the substitution character.

xs<0 Example of the XS setting.

ZL

Laser Printer. Establishes whether the printer is a laser printer and whether it supports landscape printing and envelope printing.

zl=0 Non-laser printer; does not support landscape or envelope printing.

zl=1 Laser printer; supports landscape printing and envelope printing in landscape mode.

zl=2 Laser printer; supports landscape printing but does not support envelope printing.

zl=3 Laser printer; supports landscape printing and envelope printing in portrait mode.

ZO

Orientation Output. Defines whether orientation codes are sent after the File Begin statement. This setting was designed specifically for PostScript printers. (The default is 0.)

zo=0 Sends orientation codes after the File Begin statement.

zo=1 Does not send orientation codes after the File Begin statement.

?X

Wild Card. Defines the wild card character for the printer file. The wild card can be applied to the following five printer file commands:

X Represents the horizontal value of the size command.
Y Represents the vertical value of the size command.
N Represents the font name.
R Represents re-encoding for the PostScript printer file.
? Represents bitmapping.

?x=*c* Format of the ?X setting, where *c* is the wild card character.

?x=! Example of the ?X setting.

Once defined, you can use the wild card in font definitions. For example, in a PostScript printer file, the string) !X !Y Times (will be converted to) 12 12 Times (when sent to the printer.

Character Mode Table

INTRO

Printers use attributes to change the way type appears on the page. Underlining is an attribute, for example. Different printers support different attributes, and have different rules for how they can be applied. Refer to the documentation provided with your printer for information on available attributes and possible restrictions.

PT=

The character mode table establishes the order in which attributes are applied to a font. The best way to describe a mode table is by example:

```
PT=1
MD NM=(*)
MD BO=(*+BOLD)
MD IT=(*+ITALIC)
MD RV=(*+FORMS)
MD IN=(*+INSERT)
MD DN=(*+DELETE)
MD SU=(*+SUPERSCRIPT)
MD SD=(*+SUBSCRIPT)
MD UL=(*+UNDERLINE)
MD FN=(*)
```

The first line (PT=1) identifies what follows as a character mode table. The asterisk is a wild card character that represents the font name. When XyWrite processes your document for printing, it substitutes the font name specified in the UF command for the asterisk in the mode table.

The asterisk is followed by one or more attribute names, where appropriate. (In some printer files, the attribute names precede the asterisk.) XyWrite uses these attributes when it cannot find a typeface with the requested mode built in. For example, suppose your text file contains a paragraph that is in italic. On most laser printers, italic is a typeface (e.g., Times Italic); however, on dot matrix printers, italic is a printer attribute that is applied to the current typeface.

You can use several attributes in each mode assignment, as long as the printer supports them. (Check the documentation supplied with your printer to determine what attributes can be combined.) Separate attributes with a plus sign (+); for example:

```
md bo=(*+emphasized+enhanced)
```

NOTE #1 **Mode Numbers.** You can assign attributes to mode numbers 1 to 255 as well. For example:

```
md 32=(*+compressed)
```

When you use numbered mode commands to format your document, they will not be displayed correctly in graphic view, although the printed output will be correct. (See "Display Mode Colors" in the Defaults section of this document for more information.)

NOTE #2 **Additive Modes.** Because XyWrite uses additive modes, it is not necessary to define separate entries for bold underline, bold italic, etc.

Graphics Settings

INTRO

The graphics settings define the control codes required by the printer to print graphics. There are nine different graphics settings. Different printers use the same commands differently, and not all graphic commands are necessary for every printer. For examples, refer to the printer files that were delivered with XyWrite.

GB<

Graphics Begin String. The control codes the printer needs to enter graphics mode. It selects data output, saves the current position, or turns on graphics mode. (See Note #2 for information about using wild cards with GB.)

Format: GB<*string*
Example: GB<?←&f0S←*t300r←*r0F←*r1A

GE<

Graphics End String. The control codes that tell the printer to exit graphics mode. It also restores the printer cursor position if the position was saved in the GB string.

Format: GE<*string*
Example: GE<?←*rB←&f1S

B1<
B2<
BE<

Raster Lines. These three settings send raster data. B1 is the Begin Raster string; B2 is the Additional Raster string; and BE is the End Raster string.

Format: B1<*string*
B2<*string*
BE<*string*
Example: B1<?←*b?W8DAW
B2<
BE<

GD<

Graphics Draw. The control codes for shading. If the printer has a draw command, XyWrite uses it when you request shading in an FA (Framed Area) or BO (Border) command. Using the printer code for shading is faster than rasterizing.

Format: GD<*string*
Example: GD<?) s box1 0 -?H1DA boxlin rlineto ?W1DA boxlin 0 rlineto 0 ?H1DA boxlin rlineto closepath ?-1DA box2 (

Graphics Settings (cont'd)

GV= **Graphics Variable.** Different types of printers handle graphics in different ways. This setting defines several variables for graphics processing. The format of the GV setting is:

Format: GV=*t,r,d,l,h,f*

where *t* is one of the following printer types:

0	Hewlett Packard LaserJet series	
1	PostScript	
2	9-pin dot matrix	Only fire one pin at a time; slower, but higher resolution
3	24-pin dot matrix	
4	Hewlett Packard DeskJet	
5	8-pin mode printer	
6	Interlace mode printer	
7	8-pin and interlace mode printer	
8	Printer does not support graphics	

r is the ratio of the vertical graphics resolution to printer vertical movement; it is only used for interlace mode, and is currently always set to 3 when interlace mode is active.

d is the format of the graphic data bits (0=high to low; 1=low to high).

l is the minimum length of a raster width string. (Typically, this value is set to 0.)

h is the height of the print head (distance between the top pin and the bottom pin) in vertical printer units.

f is one of the following settings for the printing device control flag.

0 Directs the printer to use a combination of whole spaces and microspaces to position the print head horizontally.

1 Directs the printer to use only microspaces to position the print head horizontally.

2 Directs the printer to output the font/attribute strings in reverse order.

4 Directs the printer to ouput a carriage return before printing in graphics mode.

8 Prevents the printer from outputting font/attribute exit strings before printing in graphics mode.

16 Directs XyWrite to send out the length of raster data defined by *l* rather than sending out a line at a time.

You can add two or more values to direct the printer to perform more than one of the functions. For example, if you set *f* to 3, the printer uses only microspaces and it outputs the font/attribute strings in reverse order.

Example: GV=0,1,0,0,0,8

GR= **Graphics Resolution.** The vertical and horizontal resolution of the printer's graphics modes.

Format: GR=*x,y*

where *x* is the number of horizontal movements in an inch and *y* is the distance between pins.

Example: GR=300,300

XP< **Horizontal Position.** This setting is for any printer that has an absolute position command. It specifies the absolute horizontal position of text and graphics.

Format: XP<*string*
Example: XP<?←*p?Z1DAX

YP< **Vertical Position.** This setting is for any printer that has an absolute position command. It specifies the absolute vertical position of text and graphics.

Format: YP<*string*
Example: YP<?←*p?Y1DAY

NOTE #1 **Wild Card Strings.** The graphics settings GB, GE, B1, B2, and BE use wild card strings to specify XyWrite output data. The first character after the less than symbol (<) defines the wild card for the setting. The next time that character is encountered, it is treated as the first character in a wild card string. The format of the string is:

cf1f2f3f4

where *c* is the wild card character. Any character that is not used in the printer code can be used as a wild card.

f1 determines what part of the graphic XyWrite is supposed to read; it can be W (graphic width), H (graphic height), T (total number of bytes or bits), - (horizontal graphic scale), or ¦ (vertical graphic scale).

f2 is the divider that is applied to *f1*; it can be 1 (indicating units are specified in bits) or 8 (indicating that units are specified in bytes).

f3 is the output format; it can be D (decimal) or H (hexadecimal). *f3* is only used when *f4* is set to ASCII.

f4 determines the form in which the values are returned; it can be A (ASCII format) or B (binary format).

For example:

```
B1<?←*b?W8DAw
```

In this example, ? is the wild card; ←*b is the transfer raster data code; and ?W8DA is the wild card string that provides the graphic width information in ASCII decimal bytes.

NOTE #2 **GB (Graphics Begin) Wild Cards.** In addition to the wild card strings explained in Note #1, the GB setting accepts two other wild cards: ?B and ?X. These wild cards represent binary data format and hex ASCII data format, respectively. For example, the following GB command is from an Epson FX850 printer file:

```
GB<??B
```

It indicates that the graphic data is in binary format.

Terminator Strings

INTRO

You can add the following assignments to your printer file to define printer codes at specific points in the document. (Place each statement on its own line.)

Soft return	LE<*string*
Beginning of File	FB<*string*
End of File	FE<*string*
End of each page	PG<*string*
Beginning of each page	PZ<*string*

LE

Line-End String. LE enables you to specify the characters inserted at the end of each *soft-return* line (that is, each word-wrapped line). These characters are inserted into the text as it is sent to the printer. (Hard-Return lines are handled under PE.)

Format:	LE<*string*
Default:	LE<♪

The default setting for the Line-End string is a carriage return (♪). You enter the carriage return character with [Ctrl] [Shift] 13.

Example #1:	LE< ←

This example inserts only a space at the end of each line — no carriage return or line feed. If you print to a file (PRINTF) with this setting, the lines within each paragraph remain wordwrapped (for further editing).

Example #2:	LE<←

This string contains no characters at all. You could use this to send to the printer thousands of uninterrupted characters, as if they were all one line. For example, you'd use it when sending a long string of graphics data to a dot matrix printer in graphics mode — with no line-end characters inserted.

FB

File Begin String. FB allows you to send characters to the printer at the *start* of a file.

Format:	FB<*string*
Example:	FB<←G

This example sends an Escape-G at the start of each file. On an IBM ProPrinter, this causes the characters to print second strike. If the JB (Job Begin) default is set, those printer codes are sent immediately after the FB code.

FE **File End String.** FE allows you to send characters to the printer at the *end* of a file.

Format:	FE<*string*
Example:	FE<♀

This example specifies that a form feed character be sent to the printer at the end of each file. This would eject the last page in most printers. The default FE statement contains the code to reset vertical spacing to 6 lines per inch.

PG **Page End String.** The PG (Page Break) setting lets you send a character string to the printer at the end of pages.

Format:	PG<*string*
Default:	PG<♪◙♀

The default sends carriage return, line feed, and form feed to the printer at the end of each page. If you use the PG setting, it is important to include the carriage return and line feed characters if you want the first line of the next page to print correctly. PG does *not* normally send the string after the last page of a file. To do that, you must specify DEFAULT FF=1 *and* DEFAULT EJ=1.

PZ **Page Begin String.** The PZ (Page Begin) setting lets you send a character string to the printer at the beginning of pages. It was implemented in Version 1.01 specifically for the Hewlett Packard LaserJet series of printers (and compatibles).

Format:	PZ<*string*
Example:	PZ<←*P0X←*P0Y

Vertical Spacing

INTRO

The vertical spacing section of the printer file contains the VU (Vertical Unit) setting and the VS (Vertical Spacing) table, each of which is described in detail below.

VU=

Vertical Unit Setting. The VU (Vertical Unit) setting in the printer file defines three separate vertical spacing specifications. Its format is:

VU=*x,y,z*

The first value (*x*) defines the number of minimum vertical movements the printer must make to equal 1/6 inch. Whenever you issue a Vertical Spacing command (LL, LS, EL), XyWrite multiplies the value in that command by *x*. For example, if *x* is 50 and you issue a Line Spacing command of LS 2, XyWrite sends instructions to the printer that it must move an amount equivalent to 100 minimum vertical movements (50 x 2).

The second value converts the output units into screen lines. It affects the way files are displayed on the screen when you are working with tables or when you type to screen with PRINTS.

The third value determines how many decimal places can be used when specifying line spacing, form depth, and other vertical format commands. It is always 100.

VS:

Vertical Spacing Table. The Vertical Spacing Table, which is optional, allows you to specify the printer codes for incremental vertical spacing. VS: is the vertical equivalent of CP: (see "Microjustification" later in this chapter). This table starts with VS:*n*, where *n* is the number of lines of code that follow. The next lines are the code directing the printer to move one vertical unit, two vertical units, and so on. For example:

```
VS:4←
←:←☻◙←
←:←♥◙←
←:←♦◙←
←:←♣◙←
ET←
```

The more entries in the VS table, the more efficient the printer is, especially for large vertical spaces. For example, if your printer moves in points and your table has only 2 entries (1 point, 2 points), moving 1 inch requires 36 movements of 2 points. However, if the table has 36 entries, moving 1 inch requires only 2 movements of 36 points.

NOTE

Default Value. If there is no VS table in the printer file, the built-in default value is a carriage return, and the default VU is 1,1,100.

Microjustification

INTRO

When you print a document, the XyWrite default is to print the text flush left, with a ragged right edge. This is unjustified text. By embedding a JU format command in your document, the text which follows will be printed justified — that is, with text flush against both margins. XyWrite justifies text by adding spaces between characters to stretch lines to the same length.

XyWrite offers two ways to justify text:

Whole-Space Justification works by adding whole spaces between the words.

Microjustification, however, adds space in fractional pieces between words and characters, rather than whole spaces only between words. We call these **partial spaces**. The advantage is that the text has a smoother, more refined appearance than with Whole Space Justification.

We have set up each printer for the highest quality printing — microjustification — if the printer supports it. However, microjustification can slow down some dot matrix printers significantly. If this is a problem you can set the printer file to whole-space justification (see Note #2).

The following settings control microjustification and relate format commands and screen display to type. You enter them into the printer file. These settings take effect only after you load the printer file with the LOAD or SETP command.

PU=

Printer Units — The PU setting converts printer units to internal units. Internal units are used by XyWrite to calculate line breaks, page length, and graphic distances. The format of the PU setting is:

PU=*h*,*v*

where *h* is the number of horizontal movements per inch and *v* is the number of vertical movements per inch. To calculate *h*, multiply the value of MU by 10; to calculate *v*, multiply the first value in the VU setting by 6.

MU=

Margin Units — The MU setting specifies the number of basic units in 1/10 inch. (A basic unit is an internal number used by XyWrite to calculate spacing.) Margin units are the units used in the LM, RM, TS and IP commands.

DU= **Display Units** — The DU setting specifies the number of basic units in 1/10 inch. Display units are used in determining the tabs and margins on-screen. DU is usually set the same as MU (Margin Units).

MS= **Microspace Units** — The MS setting specifies the number of printer escapement units in a margin unit. This setting may be different than MU for printers with character spacing different from print-head escapement, such as the Hewlett Packard LaserJet.

CP: **Character Pitch** — The CP setting specifies the printer codes for incremental horizontal spacing. CP is the horizontal equivalent of VS. This table starts with CP:*n*, where *n* is the number of lines of printer codes that are to follow. The next lines are the code directing the printer to move one horizontal unit, two horizontal units, and so on. For example:

```
CP:5
←K☺
←K☻
←K♥
←K♦
←K♣
```

SC= **Space Constant** — The SC setting specifies the number of partial spaces inserted between words before inserting space between characters within a word. The SC setting is typically set to 3 or 4.

SF= **Space Factor** — The SF setting specifies the ratio of partial spaces added between words to those added between characters within a word *after the SC spaces have been allocated. A larger number means more spaces will be added between words before they are added between characters.*

JL= **Justify Underline** — The JL setting enables (JL=1) or inhibits (JL=0) justification of text printed in the underline mode (MD UL) (see Note #3). JL is the same as JU.

JT= **Justification Type** — The JT setting is reserved for future use. All printers currently require JT=0.

NOTE #1 **How Partial Spaces are Allocated.** Before XyWrite prints a line, it calculates the number of spaces by which the line would fall short of the right margin, and then distributes those spaces between characters according to the values given for the Space Constant and the Space Factor. With SC=5 and SF=2, the first 5 partial spaces are added between words. After those spaces have been allocated, spaces are added in the ratio of 2 between words for every 1 between adjacent characters.

NOTE #2 **Selecting Whole Space Justification.** To disable microjustification and select whole-space justification, call up your printer file and insert the statement DF WS=1 on a line by itself. To switch back to microjustification, use DF WS=0 instead.

NOTE #3 **Justify Underline.** If you use JL=1, depending on the length of your printer's underline, there may be gaps between adjacent underlines. This is due to the partial spaces added between characters. If there are gaps, you should set JL=0 in your printer file.

NOTE #4 **Try Hyphenation.** Microjustification is an aesthetic feature — it makes your printed copy look better. But occasionally when a long word wraps to the next line, the line can look very stretched when printed. You can turn on automatic hyphenation or add soft hyphens to the long words to break them up.

Attribute Tables

INTRO

An attribute is a printer effect that modifies a font. Typical attributes are:

Underline
Superscript
Subscript
Italic
Bold
Emphasized
Insert
Delete
Forms

AT:

An Attribute Table contains the control codes for each attribute (or printing effect). The format of an attribute table is:

AT:*label*
AT<*on code*
AT>*off code*
ET

The first line (AT:*label*) defines the name of the attribute; the second line (AT<*on code*) defines the printer code that turns on this attribute; the third line (AT>*off code*) defines the printer code that turns off this attribute; and ET ends the table.

If you look at a printer file, you will see that there are already a number of these attributes defined, including the most common printer effects (underline, superscript, subscript, etc.) and the attributes for printing text that has been edited with Redlining on.

AT=

Attribute Value. This optional setting, which you enter in the Attribute Table as AT=*v*, uses the same values as the Font Value (FO=) setting described later.

Font Tables

INTRO

A font is a printer typeface, such as Courier or Helvetica. In some printers, a font may include a family of typefaces (bold, italic, etc.), and it may also be related to a type size and orientation. For most printers, XyWrite stores the font and width information in the printer files. For Hewlett Packard LaserJet printers (and compatibles), the font and width information is stored in a binary file (LJ.BIN or LJ3.BIN).

A Font Table contains the printer codes to access a particular character set. It can also contain the value of vertical leading (VL) associated with that character set and references to both a Substitution Table and a Width Table. (See "Terminology" earlier for examples of fonts.)

The following example of a font table is from the Panasonic 4420 printer file:

```
FO:CENTURY
FO<←(10U←(s1p10v0s3b23T
sz=10
md=bo
fy=1
vl=50
UW:10CENT-BOLD-10U
US:PCSYMBOL
ET
```

FO:

The FO: line defines the name of the Font Table, CENTURY. You can name the Font Table anything you want (as long as it is one word), and use the same name for different type variations within a type family (see MD=).

FO<

The FO< line contains the control codes that the printer needs to enter the CENTURY font. For more information on control strings, see "String Operators" which follows.

SZ=

The SZ line establishes the point size of the font being defined. (If the printer uses scalable fonts, the value in the SZ line is always -1.) The SZ line applies only to printers that support a range of point sizes.

The default size is established at the beginning of the printer file (DF SZ=12) as 12 point. The SZ formatting commands in your documents override the default setting.

MD=

The fourth line (MD=) defines the font variation defined by this font table. Legal font variations are normal, bold, italic, bold italic, superscript, and subscript (abbreviated nm, bo, it, bi, su, and sd).

OR=

The OR (Orientation) setting is used with printers that have separate fonts for landscape and portrait orientation. The format is:

OR=*n*

where *n* is 0 (for portrait orientation) or 1 (for landscape orientation). The default is 0. (If OR=1, the third value of the PM printer setting must also be set to 1.)

FY=

The FY (Family) defines the font family for this typeface. The format is:

FY=*n*

where *n* is 1 for serif; 2 for sans serif; 3 for monospaced; 4 for script; and 5 for decorative.

VL=

The next line (VL=) defines the vertical spacing, or leading value, associated with this font. VL is most useful for printers that support a range of point sizes, because it allows you to change from one type size to another within your document and have the leading change automatically. (VL is used only when Automatic Leading is on.) If the laser printer has scalable fonts, then XyWrite multiplies the value in the VL line by the point size to determine the leading (e.g., for 10 point type, the leading is 12 (10 X 1.2). If the printer does not have scalable fonts, the value in the VL line defines the leading for the specific point size.

The standard formula for calculating the value of VL is:

1. Add a modifier to the point size to allow white space between printed lines. The modifier can be any number you want, but the suggested values are: 1 for point sizes between 1 and 8; 2 for point sizes between 9 and 14; and 3 for point sizes over 14. For example, for 8-point type:

   8 + 1 = 9

2. Divide the result of step 1 by 72. For example:

   9/72 = .125

3. Multiply the result of step 2 by the second number in the PU (Printer Unit) command. For example:

   .125 x 216 = 27

UW:

UW: (Use Width Table) indicates which width table defines the width of the characters in this font.

US:

US: (Use Substitution Table) indicates which substitution table is used with this font.

FO= **Font Value.** The FO setting is an optional setting that controls several aspects of how your text looks — from underlining to microjustification. Enter the FO setting in the Font Table of the printer file. The format is:

FO=*v*

where *v* (value) is a decimal number. The values that can be used are made up of combinations from the table below (For example, if you want to use combinations 1 and 4, then the entry would be FO=5.)

Value	Usage
1	Do not output the exit string for the font when microjustifying. This allows underline mode on certain printers to be microjustified. (Assumes that there is an MM mode in the printer table.)
2	Do not output exit string for the font when microjustifying, and do not output the entry string when returning from microjustifying. This is used for superscript or subscript on printers that roll the carriage up or down.
4	Print soft hyphens in the current mode.
8	Allow microjustification of this font (overrides the JU setting).
16	Don't allow microjustification of this font (overrides the JU setting).
32	This is a foreign language font (which is basically used with special software that recognizes character sets in extra memory such as on an EGA card).

PS= **PostScript Flag.** Indicates whether the PostScript re-encode information in the FB (File Begin) statement is to be loaded into memory. The format is:

PS=*n*

where *n* is 0 if the re-encode information is not to be loaded and 1 if it is to be loaded.

ET ET (End Table) marks the end of a Font Table.

CN:

CN: (Cartridge Name) defines the set of fonts on a particular cartridge for all printers except Hewlett Packard LaserJets and compatibles (see Note #3). Cartridge sets must be defined after the last of the internal font definitions. (In standard XyWrite printer files, STANDARD is the last internal Font Table.)

The format of CN is:

```
CN:label
font-table1
font-table2
  .
  .
  .
font-tablen
```

where *label* is the name of the cartridge and *font-table1...font-tablen* are the Font Tables for the fonts on the named cartridge. A cartridge set includes all the Font Tables between CN commands, between a CN command and an SN command, or between the last CN command and the end of the printer file. For example:

```
CN:CENTURY
FO:CENTURY
FO<←[80C←[23y←)'1←[7l
FO>←(B←[0K←[23m←[22m
MD=NM
SZ=8
VL=40
UW:CENTURY8M
ET
:
FO:CENTURY
FO<←)'1←[7l←[3m
FO>←(B←[0K←[23m←[22m←[0m
MD=IT
SZ=8
VL=40
UW:CENTURY8I
ET
```

This example defines a cartridge named Century that contains two fonts: 8-point Century normal and 8-point Century italic.

XyWrite's menus use the CN setting to determine which font names to display in the Typeface dialog box when a cartridge is selected.

SN: SN (Soft Font Name) identifies the soft fonts supported by this printer file. This information is used to create the list of available soft fonts displayed by the menus. (The SN command is not used for Hewlett Packard LaserJet printer files or compatibles. XyWrite gets soft font information for this class of printers from the soft font files that are downloaded to the printer.)

The format of SN is:

SN:*diskname menuname comments*

where *diskname* is the name of the soft font as it is stored on disk; *menuname* is the soft font name you want displayed in the menus; and *comments* (optional) is any font information you want displayed in the menu. For example:

```
SN:3tms27.rm8 Times 8U Medium
```

NOTE #1 **Generic Font Table.** The first Font Table in every printer file is a generic one that defines the wild card used in the Character Mode Table described earlier. The format of this table is:

```
FO:*
ET
```

NOTE #2 **Standard Font Table.** The last internal Font Table in every printer file defines the STANDARD font, which is a 10 character-per-inch monospaced font. This is the default font which is used by XyWrite to format your documents when no other font is specified. If you define cartridge sets (with the CN command), enter those definitions after the STANDARD Font Table.

NOTE #3 **Hewlett Packard LaserJet Printers and Compatibles.** XyWrite gets the font information for cartridge and internal fonts for LaserJet printers and compatibles from the binary font files LJ.BIN and LJ3.BIN which are shipped with XyWrite.

<
>
#
%
*

String Operators. In both the Font Tables and the Attribute Tables, you can specify the printer codes that are sent to the printer. For example:

AT<←U or FO<←p1

Each statement includes a name (AT or FO), an operator (<) and a string (←U or ←p1).

The *string* is the series of printer control codes, and can be any number of characters. The operator tells XyWrite to insert the *string* into the text in a specific manner:

< insert string upon *entering* the character mode
> insert string upon *leaving* the character mode
# insert string *after* each character in the mode
% insert string *before* each character in the mode
* insert string *in place of* each character in the mode

The pound (#) and percent (%) symbols are used along with the backspace character (◘), mainly for overstrike.

Entering a Mode

Format: AT<*string*
Example: AT<←U

The two characters Escape (←) and U are inserted into the text string at the *start* of the attribute.

Leaving a Mode

Format: AT>*string*
Example: AT>←D

The two characters Escape (←) and D are inserted into the text at the *end* of the attribute.

Overstrike Character

Format: AT#◘*string*
Example: AT#◘-

The string "backspace, hyphen" is sent *after* each character. Thus, every character is overstruck with a hyphen. A special case of this follows.

Double-Strike Character

Format: AT#◘

The string "backspace, space" is a special case — it causes the printer to move back and print the same character again. (Notice that it does not move back and print a space.)

Pre-Strike Character

Format: AT%_string_
Example: AT%-◘

The string "hyphen, backspace" is sent _ahead_ of each character. Thus, the hyphen is printed, then overstruck by the next character.

Non-Printing Character

Format: AT*_string_
Example #1: AT*

Every character is removed and replaced with a single space. Thus, this is the non-printing mode, useful when spaces and all characters are the same size. (When using _proportionally-spaced characters,_ each space is narrower than the characters they replace. This can present unexpected results.)

Example #2: AT*←

Every character is removed altogether (and replaced with nothing). This is also a non-printing mode — the displayed characters do not take up any room on the printout.

INTRO

Different fonts have different character widths, and within proportional fonts, different characters are assigned to different widths. XyWrite uses this width information to determine how many characters will fit on a line of text.

WD:

A Width Table defines the widths of characters. The table begins with WD (Width Define) and SW (Standard Width) and ends with ET (End Table). SW and the other lines specify the width of characters—the characters are listed on the left of the equal sign (=), and the width is given on the right. The width is given in Basic Units (also called Width Units) — the smallest incremental width of which your printer is capable. If the printer uses scalable fonts, the widths are defined for 1-point fonts only.

To keep the list orderly, after SW we list the characters in order of size, narrowest first. For example:

```
WD:HELVETICA
SW=20
l.:=10
I=15
ij=16
bpqv=26
WXM=30
ET←
```

SW (Standard Width) specifies the width of all characters not otherwise specified in the table. You normally assign SW to the width that applies to the most number of characters. This prevents you from having to specifically list those characters in the Width Table.

NOTE #1

Determining Width Information. Ideally, the width information for a font is supplied by the font manufacturer. If you do not have that information, use the following procedure:

1. Create a test file and enter a line for each XyWrite character, starting with character 32. Each line should have the format:

   !nnnnnnnnnnnnnnnnnnnnnnnnn!#

   where ! marks the beginning and end of the character printout; *n* is 25 repetitions of the character being tested; and # (optional) is the character number.
2. Print the file.
3. Measure the width of each 25-character line. Don't include the exclamation points or the character number in your measurements.

4. Divide each measurement by 25 and multiply the resulting number by the first number in the PU (Printer Unit) setting.

   If the result of your calculation is a fraction, you need to convert it to a whole number. You can either round it off to the nearest whole number or you can "scale up." To scale up, you multiply all the widths (in every width table in the printer file), as well as the MU and DU values, by a common denominator.

5. Create a width table in the printer file and enter the width information.

Once the width table has been created and loaded into memory, you can test it using the following procedure.

1. Create a test file.
2. Enter the UF command that will test the width table you just created.
3. Turn on justification (JU) and set the right margin to 3 inches.

   Type: [F5]ju↵
   Type: [F5]rm 3↵

4. Enter a line for each XyWrite character. Start each line with an exclamation point; type an LD command with the character as an argument; and then end the line with an exclamation point. For example:

   Type: !
   Type: [F5]ld a↵
   Type: !

   Result: This creates a 3-inch wide line of lowercase a's which are preceded and followed by an exclamation point.

5. Print the file.
6. Examine the printout for places where the right margin is out of alignment. If you find any, make one of the following adjustments:

   If the line is too short, decrease the width of that character in the test printer file by one or two units and print the file (or section) again. *If the line is too long,* increase the width of that character in the test printer file by one or two units and print the file (or section) again.

7. After modifying the widths, make sure you store and reload the printer file before printing again.
8. If the widths are still not right, repeat steps 3 and 4 until they are.

NOTE

Character Widths. The width tables reflect the actual width of the characters. Changing the width size does not affect the size or shape of the characters.

Substitution Tables

INTRO

Your printer may be capable of printing characters that are not part of the XyWrite character set. To enable you to access those characters, XyWrite uses a Substitution Table.

SU:

A Substitution Table specifies how displayed characters are to be printed. It allows you to construct graphic symbols and math signs, for instance, or to print characters that are not part of the XyWrite character set but are part of the printer's character set. The format of the Substitution Table is:

```
SU:name,x
d=p
d=p
d=p
ET
```

where *name* is the name of the substitution table; *x* (optional) is a generic substitution character that is printed when the requested character is not available (it overrides the XS printer setting); *d* is the displayed character; *p* is the printer character or printer codes used to create a character. SU: and ET start and end the table.

The following example is a portion of the Substitution Table from the PostScript printer file:

```
SU:SUB
å=\276
À=\311
ä=\331
Å=\342
ET
```

NOTE #1

Common Names. Width Tables, Substitution Tables, and Font Tables can all use the same names. The only constraint is that Attribute Tables and Font Tables should not have the same names.

NOTE #2

Hewlett Packard LaserJet Printer and Compatibles. The Hewlett Packard LaserJet printers (and compatibles) provide several symbol sets. By default, the XyWrite printer files associate a particular symbol set with a font. If you want to use a different symbol set, you must create a Font Table that references the Substitution Table you prefer. (The Substitution Tables already exist in the printer files.) The Font Table must have a unique name, and must be supported by a Width Table.

NOTES

Sort File

INTRO

Every time you create an index, display a directory, or execute a SORT or SORTD command, XyWrite arranges the file in a predefined way. If you wish, you can customize the sorting order to match your needs.

CONTENTS

Sort File (cont'd)

PURPOSE

XyWrite has an internal set of sorting rules that automatically arranges directory listings, index entries and sorted files in alphabetical order and makes other prioritizing decisions. For example, in multi-word entries, XyWrite alphabetizes the first word only, unless more than one entry has the same first word. The result is a list of entries that looks like this:

wild alphanumeric, 6-56
wild letter, 6-57
wild number, 6-50
wildcard, 6-55

You can change the default sorting order by creating a new sort file that lists the XyWrite characters in the order you want them sorted. (XyWrite characters include letters, numerals and other symbols. The complete list is in Appendix E.) You can also tell XyWrite that you want a character evaluated as a character string (for example, treat ü as "ue" for sorting purposes).

ACTION

Creating the Sort File

The sort file contains a list of characters in the order in which you want them sorted. The first line of the file contains the identifying label (;SO;) that tells XyWrite that it is a sort file. The next line contains the character or characters that you want to have sorted first in your list (e.g., aA). The third line contains those characters that you want to have sorted second, and so on.

Any characters that you omit from the sort file are ignored when you create an index.

Let's create a sort file that builds alphabetical indexes, but ignores spaces between words in multi-word entries. It would sort the example above like this:

wild alphanumeric, 6-56
wildcard, 6-55
wild letter, 6-57
wild number, 6-50

1. Create a new file in which to store the sorting information.

   Type: F5new nospace.srt↵

2. On the first line, enter the label that identifies the file as a sort file.

   Type: ;SO;

3. On the next line, enter the characters to be sorted first. In this example, we leave the space out of the file altogether (normally, it would be the *first* character in the file).

   Type: aA

4. Continue entering all characters that you want to have included in your sort. Be sure to enter them in the correct order; those characters at the top of the sort file will be sorted first. For example:

   ```
   Type:  bB
          cC
          dD
          .
          .
          .
          zZ
   ```

5. Include all characters that you want the sort character to use. In this example, omit the blank space from the list.

6. Store the sort file and then load it using the procedure "Loading a Sort File."

   Type: F5st↵

ACTION

Sorting to a Sequence of Letters

To sort a single character as a sequence of characters, you enter the following definition in the sort file:

c=string

where *c* is a single character and *string* is the group of characters that defines the sort order for *c*. For example, if you want "ü" to be treated as the string "ue" when you sort, make the following entry in the sort file.

ü=ue

You can put this entry anywhere in the sort file, but we recommend putting it at the beginning of the file, right after the line that defines the file label (;SO;).

ACTION

Loading a Sort File

To load a sort file into memory, enter the LOAD command along with the name of the sort file you want to load. For example, if you have created a file called NOSPACE.SRT:

Type: F5load nospace.srt↵

Result: The sort file NOSPACE.SRT is now in effect.

NOTE #1 **Startup.** In order to use a sort file you have created, you must first load it into memory. You can have your sort file automatically loaded at startup by adding a line to your STARTUP.INT file. For example, the following entry loads a sort file named TABLE.SRT.

```
BC load table.srt←
```

Refer to "Startup File" for more information on modifying STARTUP.INT.

NOTE #2 **Sort Key.** When XyWrite arranges (or sorts) an index, it uses the first 80 characters in each entry to determine where the entry belongs in the list. This number is called the *sort key*. You can reduce the amount of memory required to arrange the index by reducing the size of the sort key. To change the size of the sort key, use the DEFAULT command to enter the SK (Sort Key) setting. (See "Default Settings" for more information.)

NOTE #3 **SO (Sort) Setting.** In addition to the order established by the sort file, the SORTD command uses the values established by the SO default setting. Refer to "Default Settings" for more information about SO. (The SO setting has no impact on the order of indexes, directory listings, or lists sorted with the SORT command.)

NOTE #4 **Directory Listings.** The DSORT command also gives you some options for changing the order in which directory listings are displayed. For example, DSORT allows you to sort files in reverse alphabetical order. For information on the DSORT command, refer to Chapter 2 of the *Command Reference Guide*. (The DSORT command has no impact on the order of indexes or lists sorted with the SORTD or SORT command.)

Startup File

INTRO

The STARTUP.INT file enables you to customize XyWrite to your needs. With it, you can tailor the initial conditions, such as loading a printer file for your printer and loading your own keyboard file.

If you are familiar with DOS, you'll notice the similarity between STARTUP.INT, which initializes XyWrite, and AUTOEXEC.BAT, which initializes DOS. It doesn't take much time to learn about the STARTUP.INT file, since there are no new commands to learn.

CONTENTS

PURPOSE

STARTUP.INT is the initial startup file for XyWrite. It contains commands that are automatically executed every time XyWrite is started up, which enables you to custom tailor the program. For example, STARTUP.INT can log you on; load your keyboard, help, and menu files; select the printer; load cartridges; set the order that filenames are displayed in directories.

STARTUP.INT is actually a program file (see "User Programming" for more information on program files). XyWrite comes with a simple STARTUP.INT file, which you can modify to include any commands you want.

ACTION

Running the STARTUP.INT File

You can run STARTUP.INT either automatically or manually.

- **Automatically.** Each time you *start* XyWrite, STARTUP.INT runs automatically:

  1. Start at the DOS prompt (for example: C>).
  2. Type: editor↵

  Result: XyWrite is loaded, the familiar XyWrite screen appears and STARTUP.INT runs automatically. Each command in the file is executed in sequence, appearing briefly on the command line. Upon completion, the XyWrite opening screen is displayed.

- **Manually.** While in XyWrite, you can run STARTUP.INT any time you want to reinitialize the program:

  1. Store any documents you are working on before you run STARTUP.INT.
  2. Type: F5run startup.int↵

  Result: XyWrite executes each command in the file one after another.

ACTION

Modifying STARTUP.INT

The following procedure shows you the simplest way to modify the STARTUP.INT file. As an example, the procedure adds the CART command to automatically load cartridge fonts during startup.

1. Call the existing STARTUP.INT file to the display:

   Type: F5call startup.int↵

2. Move the cursor to the line *after* the line containing the SETP command. Make sure you are in Insert Mode.

3. Each line in STARTUP.INT must begin with a function call. Most lines begin with BX, a function call that indicates that what follows is a command. One way to enter this function call is to copy an existing line in the file. Alternatively, you can issue the following command:

   Type: F5pfunc bx↵

4. Enter the CART command, followed by a one-, two-, or three-character abbreviation for the cartridge you want to load. If you include more than one name, separate them with commas. For example:

   Type: cart a,gt↵

5. Enter the function call Q2, which tells XyWrite to execute the command on this line. If you copied an existing line in step 4 above, the Q2 will already be there. Otherwise:

   Type: F5pfunc q2↵

6. (Optional) Type a comment after the Q2 function call. Comments start with the string ;*;. For example:

   Type: ;*; load cartridge fonts

7. Press: ↵

8. Store the file:

   Type: F5store↵

9. Test the file:

   Type: F5run startup.int↵

Result: The commands in the STARTUP.INT file are executed one at a time, although they do not appear on the command line. The command, CART, will now automatically load your font cartridges whenever you start XyWrite.

NOTE #1 **Order of Commands.** The order of commands in STARTUP.INT can be very important when setting defaults. The rule is: The last setting takes precedence. For example, in Case 1 below, if SETTINGS.DFL sets the offset to OF.8IN,.8IN (with the statement DF OF=.8IN,.8IN), then the offset would be overridden by the next statement, DEFAULT OF=1.2IN,1.2IN.

Case 1:

```
BX load settings.dflQ2 ←
BX default of=1.2in,1.2inQ2 ←
```

Similarly in Case 2, DEFAULT OF=1.2IN,1.2IN is overridden by the default settings in SETTINGS.DFL.

Case 2:

```
BX default of=1.2in,1.2inQ2 ←
BX load settings.dflQ2 ←
```

NOTE #2 **Alternate Startup File.** If you want, you can specify the name of an alternate startup file for XyWrite to run. For information on how to do this, refer to Chapter 2 of the *Command Reference Guide.*

User Programming

INTRO User programming involves using XyWrite Programming Language (XPL) to write a variety of customized application programs. You can make your programs as simple or as sophisticated as you want — you can record any sequence of keystrokes and create a program for later execution or you can use XPL commands to add more power and functionality to your programs.

CONTENTS

NOTE #1 **Programming Background.** This section contains advanced programming material and is not recommended for beginners. You need some programming background to get the full benefits of the programming features.

User Programming

PURPOSE

User Programming extends the power and functionality of XyWrite. You can create new functions based on combinations of existing functions to bring convenience and speed to your daily work.

For example, you can create procedures that accept keyboard input, such as an order entry system. You can also write user programs to accomplish complex editing tasks that conditionally require a change to be made. For example, you can create a program that removes list entries dated prior to a certain date.

User Programming commands enable you to:

- Save values to 1000 macros (00–999) during program execution
- Save strings, subroutines, or expressions in a macro
- Use the contents of a macro in an expression
- Divide the contents of a macro into parts, extract the parts, and then save them to other macros
- Insert the contents of a macro at the cursor position
- Clear macros from memory
- Branch on a condition (IF Statement)
- Test for errors
- Stop to wait for keyboard input any time during program execution
- Pass values to a program as it starts
- Determine the current cursor position
- Make use of the current filename, path, page number, line number, and other XyWrite values and settings
- Label a point in the program
- Jump to a label
- Switch to the window that contains a specific file
- Use operators and functions to perform various tasks in your program
- Produce an audible signal at a point in your program
- Use the XPL debugger when testing your program to pause execution at certain points
- Execute a command from within a program without clearing the command line

Procedures for User Programming

CREATING AND RUNNING AN XPL PROGRAM

The following provides an overview of the basic procedure for creating a user program. An explanation of most of the commands and functions described here are provided elsewhere in this section.

Step 1 — Plan the Program. Analyze the steps you'd take if you manually performed the task. You can then prepare a list of instructions to tell your computer how to do the same task automatically. If the task is complex, you may find it helpful to draw a flowchart. Consider all possibilities and try to anticipate problems.

Step 2 — Create or Call a Program File. Use the NE (New) command to create a program file or the CA (Call) command to open one.

Step 3 — Write the Program. Turn on program mode and type the exact keystrokes for the procedure you want. [Scroll Lock] turns program mode on and off. When program mode is on, XyWrite displays an "S" at the top right of the screen, displays a message on the status line, and records all your keystrokes as function calls in your program file.

You use the PFUNC (Put Function Call) command to insert function calls not assigned to keys into your program. (Be sure to turn off program mode before using the PFUNC command.)

When you are writing XPL programs, you can use macros to save strings or values. XyWrite provides one thousand programming macros, numbered 00 to 999, for use in XPL programs.

Record the keystrokes that represent the actions you want to occur when the program begins execution. When you reach a point in the program where you want to insert an *embedded* command, turn off program mode by pressing [Scroll Lock]. Type in the desired embedded commands. Each time, press [F5] to move to the command line, enter the command, and press [F9].

You can make your programs easier to read by including notes or comments throughout the program file; for example, to explain the program's purpose or to identify different sections of the program. Use the string ;*; in a program file to indicate that the text that follows is a comment. A comment can be as long as you want; and you can include as many different comments in a file as you need. A carriage return marks the end of a comment.

When you are finished entering the program, be sure to turn off program mode.

Step 4 — Store the Program. When the program is as you want it, store it on disk. You cannot run your program until it is stored on disk.

Step 5 — Test and Debug the Program. Try out your program to see if you get the results you expected. Use the RUN command to execute the program.

Default DB (Debugger) enables you to execute your program one element at a time so you can verify that the program is doing what you want. Turn debugging on before you run your program by setting the DB default. Refer to "Default Settings" for more information.

Step 6 — Load the Program to a Key. This step is optional, but it makes using your program easier. You use the LDPM (Load Program) command to load a program created in program mode to a key.

If you want to keep the program file loaded on the macro key for use after you quit XyWrite, use the STSGT (Store Macro) command.

ACTION

Entering XPL Commands

Many XPL commands are embedded commands, and you enter them in your program as you would any other embedded XyWrite commands. Before inserting an embedded command, be sure program mode is off. After you press F9 to enter certain of these embedded commands in your program, XyWrite opens a command window so you can include an argument for the command. Press Shift F1 to close the window when you have entered the appropriate argument. (To abort the command window, press Esc.) For example, to enter the command: «SX01,«RC»»:

1. Enter the SX command:

   Type: F5sx↵

2. Type the rest of the command in the command window:

   Type: 01,F5rc↵

   Press: Shift F1

You can also use expanded view for typing or modifying commands. For example, to enter the command: « SX01, « RC » »:

1. Press: Ctrl F8 to switch to expanded view.

2. Move the cursor where you want to insert the command, and:

   Press: Ctrl <
   Type: SX01,
   Press: Ctrl <
   Type: RC
   Press: Ctrl >
   Press: Ctrl >

3. Press: F8 to return to draft view.

ACTION

Revising XPL Programs

To revise a user program, you need to switch to expanded view to reveal all embedded commands:

1. Call up the program file you want to revise using the CA (Call) command. For example, to revise the file EXERCISE.PGM:

   Type: F5 ca exercise.pgm ↵

2. Press: Ctrl F8 to switch to expanded view.

   Result: The markers no longer appear — the commands are displayed expanded within double angle brackets.

3. Insert and delete any commands you want removed from the program including their surrounding double angle brackets.

4. Press: F8 to switch to draft view.

   Result: The commands that were expanded on the screen are now shown as markers.

Function Calls

Function calls are two-letter instructions that represent basic XyWrite activities. Some function calls are assigned to keys and some are not.

In a program, XyWrite function calls look like two bold letters followed by a space; however, they are unique, single-unit representations that must be created either by using program mode or by using the PFUNC (Put Function Call) command.

After you press [Scroll Lock] to turn program mode on, every key you press that is not an alphanumeric character causes XyWrite to record the function call assigned to that key into your program file. For example, if you press [F5], XyWrite enters the function call BC (Blank the Command Line) into your program.

You can use the PFUNC command to insert function calls into your program even if they are not assigned to keys. Be sure to turn off program mode before using the PFUNC command. For example, to enter the NM (No Markers) function call, which hides all embedded command markers and paragraph-end arrows

Type: [F5]pfunc nm[↵]

Refer to the section "Function Calls" elsewhere in this document for a complete list of all XyWrite function calls. You can use all of those function calls in programs except TS (Toggle Scroll Lock). TS cannot be entered into a program file, since you use it to begin and end the recording of a program.

Two function calls especially useful in XPL programs are BC (Blank the Command Line) and BX (Blind Execute).

The BC function call clears the command line and moves the cursor to the start of the command line. Example: **BC** ca test**XC**

The BX function call executes a command from within a program without moving the cursor to or clearing the command line. If the command and any arguments are enclosed in parentheses ( ), square brackets [ ], or curly braces { }, XyWrite executes the command automatically.
Example: **BX** (ca test)

Otherwise, if they are not so enclosed, you must use the Q2 (Execute) function call with the BX function call to execute the command. Q2 is useful for executing a command string that may contain parentheses, square brackets, or curly braces. Example: **BX** ca test**Q2**

Macros

Macros are storage areas, or buffers, you can use to save strings or values. XyWrite provides several types of macros: ordinary, programming, and additional programming.

- **Ordinary Macros: A-Z, 0-9**
  These are the macros that you usually access with the F2 keys. They are "permanent" in that their contents remain intact when the program stops running.
- **Programming Macros: 00-999**
  These one-thousand macros can be used only within an XPL program. The contents of programming macros 00-99 are saved only while the program is running and vanish at the end of each program. The contents of programming macros 100-999 remain intact until either they are overwritten or you quit XyWrite. Ordinary macros 0 through 9 are distinct and separate from programming macros 00 through 09.
- **Additional Programming Macros: &A-&Z, &0-&9**
  These can be used only with the LDPM (Load Program) and the RUN commands. Refer to the sections that describe these commands for more information. Like ordinary macros, these are "permanent" and remain intact until either you overwrite them or quit XyWrite.

If you use programming macros exclusively for programs, you can reserve your ordinary macros for on-the-fly use; since programming macros are not accessible from the keyboard.

There are several embedded commands you can use with programming macros. There are three kinds of information you can store in a programming macro: a numeric expression (value), a literal string of characters, or a program. The kind of information you are saving determines the command you use to save it and the command you use to access it. Refer to "XPL Commands" for more information on macro commands.

You can use the PR (Prompt) command in a program to display the contents of a macro on the status line. To do this you specify the number of the macro, preceded by an @ character in the *message* argument of the PR command. The macro must be the last thing in the PR command. Example: «prThe file is @98».

NOTE

Reserved Macros. There are also 1000 programming macros, numbered 1000 through 1999 that are reserved for use by XyWrite menus.

XPL Commands

There are four categories of XPL commands: Commands that Return Values, Flow Control Commands, Macro Commands, and Miscellaneous XPL Commands. Most of the commands are *embedded* commands and appear as markers in draft view. Arguments (values) to embedded commands can include macros, logical expressions, numeric expressions, or additional embedded commands.

Commands that Return Values

The following commands insert values into your program from XyWrite, from the user, or from another program. You can use any of these commands as values in expressions.

AS

Argument Insert. Takes the string passed in from the RUN *filename,string* command and saves it to macro 00. You can then use the PV (Put Variable) command to access the contents of macro 00 on the screen, in the text, or in an expression. For example, if you start the program with RUN EXERCISE.PGM,1234, then macro 00 contains the string 1234 (not the number) within the program. If you load the program to a key with LDPM and then run that program from a macro key, macro 00 will contain any string you enter on the command line. Since AS returns a *string* rather than a number, use IS00 when you want to use AS within SX; for example, «SX01,«IS00»».
Example #1: The new variable is «AS»
Example #2: «IF«AS»==«PV01»»

CL

Column Location of Cursor. Takes on the value of the current column position of the cursor. The columns on the display are numbered starting from the left at column 0. (In graphic view, CL displays the pixel position of the cursor rather than the column location.) Example: «SX01,«CL»»

CP

Character Position. Takes on the value of the current character position in number of characters from the beginning of the file. Example: «SX01,«CP»»

Once you have saved the cursor position and moved the cursor, you can use the JMP (Jump) command to return to that previous position: Example: F5 jmp «pv01» ↵

ER

Error. Takes on a logical value of either TRUE or FALSE. ER is TRUE if there was an error in the previously executed command; otherwise, it has the value FALSE. Frequently used with the SEARCH command — ER is TRUE if the search returns NOT FOUND. Example: «IF«ER»»

EXIST *filename*

Exist. Checks to see if a filename already exists in the directory you specify and takes on a logical value of either TRUE or FALSE. EXIST is TRUE if the file is not found; otherwise, it has the value FALSE. Example: BX exist « pv01 » Q2

RC

Read Character. Stops the program and waits for the user to press a key. When a key is pressed, RC takes on the value of that key and the program continues with the next step. Use RC in an expression; typically you would assign RC to a macro. Example: « SX01,« RC» »

VA *nm*

Value of Variable. Obtains the current value (or string) of the variable you request (*nm*). You can request any XyWrite default setting (for example, VA RM obtains the current value of the right margin) or any environment setting (for example, VA $PA obtains the current drive and path). Unlike the other commands in this section (except IS), you can use this command *either* in a program *or* directly in text. Refer to "VA Settings" for a list of all the variables you can use with the VA command. Example: « VALM »

You can use the /NV switch with the VA command to display the value on the status line. Example: F5va/nv lm↵

Flow Control Commands

The following commands control the flow of the program.

EX

Exit and Continue. EX and EX1 are two different ways to exit a program. EX is a subroutine return. When used at the main level of a program, the EX command stops the program. When used in a subroutine, EX exits from that subroutine and continues execution at the point the subroutine returns to. (A subroutine is any program within a program.) Example: « EX »

EX1

Exit and Stop. EX1 stops the program altogether. Unlike EX, it stops regardless of whether execution is in the main program or in a subroutine. Example: « EX1 »

GL *label*

Go to Label. Causes execution to jump to the label specified by the LB command. The *label* can be any length string. Example: « GLSTART » To make your programs easier to read, type two carriage returns after each GL statement. This will not affect operation of the program, since the GL command skips all statements and goes directly to the specified label. Label names are case sensitive; therefore, « glend » does *not* jump to « lbEND » .

IF *expression trueaction* **EI**

IF Condition. This command evaluates a boolean expression and determines whether the expression is TRUE or FALSE. If the expression is TRUE, the commands specified in *trueaction* are executed and then execution continues with the next statement after the EI command (End If) command. If the expression is FALSE, execution jumps to EI and continues from there. Examples:

Comparing Values:
«IF«PV01»==«PV02»»«GLA»«EI»«GLB»

Comparing Strings:
«IF«IS01»==«IS99»»«GLA»«EI»«GLB»

LB *label*

Label. Marks a spot in the program that you can jump to with the GL command. The *label* can be inserted at any point in a program, and can be any length string. LB can also be used simply as a comment in a program. (When writing a comment that contains spaces, though, the comment must be typed in expanded view.) Example: «LBSTART»

Macro Commands

In some of the following commands, you specify #, which is a macro: either an ordinary macro (A-Z, 0–9) or a programming macro (00–999). Macros &A–&Z and &0–&9 do *not* work with those commands.

CLRASG

Clear All Macros. Clears all macros — A-Z, 0–9, &A–&Z, &0–&9 and 100–1999. CLRASG is an immediate command. Example: F5clrasg↵

CLRSGT

Clear Ordinary Macros. Clears all ordinary macros A-Z and 0–9, but leaves programs assigned to programming macros (100–1999) and additional programming macros (&A–&Z and &0–&9) in memory. CLRSGT is an immediate command. Example: F5clrsgt↵

CLRXSGT

Clear Additional Macros. Clears only the additional programming macros (10–999). CLRXSGT is an immediate command. Example: F5clrxsgt↵

GT #

Get Macro. Inserts the text or invokes the program or subroutine assigned to the specified macro (#).

GT # is similar to the PV (Put Variable) command, except GT inserts text all at once, while PV inserts the text one character at a time. If overstrike mode is on, the text output by the GT command does not overwrite existing text. Example: «GT98»

IS # Insert macro. In User Programming, you can use IS within an IF statement to compare the string contents of one macro either to another macro or to a literal string enclosed in double quotes. (When used outside of XPL, IS operates as just another regular formatting command.)
Example #1: « IF « IS01 » == « IS99 » »
Example #2: « IF « IS02» =="XyWrite"

PV # Put Variable. Inserts the characters one at a time from the specified macro to the current location. May be used in two ways:

- When used outside of an expression, PV inserts the text or runs the program assigned to the macro — either on the command line or in text. If overstrike mode is on, the text output by the PV command overwrites existing text. Example: « SX01, « VASfi» » « PV01 »
- When used inside an expression, XyWrite interprets the content of the macro as a number and combines it with the rest of the expression to give a numeric result. Example: « IF « PV800 » < « PV801 » »

REMOVE # Remove. Clears any single macro # (or user program) from memory. REMOVE is an immediate command. Example: F5 remove x ↵

SAVE %#,*filename*
Save. Saves the contents of a macro # to a new file on disk *filename,* without opening the file. SAVE is an immediate command. Example: F5 save %b.myfile ↵

SU #,*string* Save Subroutine. Saves *string* as a program to the specified macro (#). (This is similar to SV, except the string is marked as a program.) You can run this subroutine within another program using either the GT or the PV command. The following example saves to macro 98 a subroutine that executes the SAVE command. Example: « SU98,BC saveXC» »

SV #,*stringexp* Save String. Saves a literal string to the specified macro (#). You can specify the string expression (*stringexp*); or if you omit the comma and *stringexp,* XyWrite saves any block of text that is selected when you run the program. If you omit *stringexp,* XyWrite clears the macro or initializes it with a size of 0. SV can be used only for saving literals. You can compare this string to contents of any other macro or to a literal string. Example #1 puts the string YES into macro 99; Example #2 initializes or clears macro 32; Example #3 saves the currently selected block to macro 33.
Example #1: « SV99,YES»
Example #2: « SV32,»
Example #3: « SV33»

SX *#,numericexp*

Save Expression. Evaluates a character or series of characters and saves the result to the specified macro (#). The character(s) can be a number (such as 88), a numeric expression (such as the result of 2 + 2), or a XyWrite variable (such as the current left margin setting).
Example #1: «SX01,25»
Example #2: «SX02,«IS99»»

XS *#1,#2,#3,#4,#5*

Extract String. Divides the contents of a macro into parts (often called parsing), then extracts and saves the parts to other macros. The XS command works by searching the specified macro for the character(s) you designate. If it finds a match, XS creates three new macros: one for storing the information that precedes the matched string; one for the matched string itself; and one for the information that follows the matched string.

#1 is the macro that contains the string you want to search.

#2 is the macro that contains the string you want to match. (You can use wild cards in this string.)

#3 is the macro where XS will store the information that precedes the matched string.

#4 is the macro where XS will store the matched string. (Unless you use wild cards in #2, the contents of this macro will be identical to the contents of #2.)

#5 is the macro where XS will store the information that follows the matched string.

For example, you may want to extract a filename extension to determine what printer file to use for a file or to what subdirectory to send the file, you could use the XS command in a program like this:

```
«XSa,b,x,y,z»
```

If macro A contains the string CHAPTER.DOC, macros B and Y contain a period, macro X contains the string CHAPTER, and macro Z contains the string DOC.

NOTE Wild Cards. XyWrite accepts wild card characters in the string you want to match. Use the same wild cards as you use with the Search command. To enter them into the program file, press Shift Esc followed by the wild card letter you want.

MISCELLANEOUS XPL COMMANDS

You can use the following commands for various tasks in your XPL programs:

- APFIL (to add a line to a stored file)
- BEEP (to produce an audible signal)
- GOFILE (to switch windows to an open file)
- LDPM (to load a program)
- PFUNC (to enter a program function call)
- RUN (to run a program)
- STSGT (to store macro keys to disk)

These commands are described in detail on the following pages.

Append to File APFIL

FORMAT C-XYW APFIL/*x filename,string*

x is the character that signals the beginning of the replacement string.
filename is the name of the text file you want to modify.
string is the replacement string.

MENU Not a menu item.

PURPOSE APFIL (Append to File) lets you change an existing line or add a new line to a file without opening the file. The XyWrite menus use this command to change a default setting in SETTINGS.DFL. If the file contains the command to establish the specified default, the command is changed to reflect the new setting; if the file does not contain a command for that setting, one is added to the end of the file.

When looking for a line to change, XyWrite uses the first section of the replacement string (up to the signal character *x*) as the search string for locating the line to modify.

ACTION **Changing a Default Setting in SETTINGS.DFL**

To change a default setting in SETTINGS.DFL:

1. Determine the location of SETTINGS.DFL (using VA$DL) and the setting you want to modify. Let's assume that the location of SETTINGS.DFL has been stored in macro #01.
2. Construct an APFIL command in your program. For example, to change the NW default to 0:

```
    BX apfil/= «pv01»,DF NW=0Q2
```

Result: when the program is run, SETTINGS.DFL will be changed so that the NW setting is zero. If SETTINGS.DFL does not already contain a line establishing an NW setting, one is added at the end of the file. If it already contains a line establishing an NW setting, that line is altered. All characters following the equal sign are removed. In their place, the number zero appears.

NOTE #1 Signal Character. The signal character does not have to be an equal sign. It can be any character contained in both the search and replacement string.

NOTE #2 If the text file contains more than one line that matches the search criteria, only the first instance is changed.

BEEP — Producing an Audible Signal

FORMAT C:\XY4 BEEP

This is an immediate command.

MENU Not a menu item.

PURPOSE BEEP produces an audible signal at a point in your program.

ACTION **Producing an Audible Signal**
To produce an audible signal:

1. Switch to expanded view:

   Press: [Ctrl] [F8]

2. Use PFUNC to enter either the BC (Blank the Command Line) or the BX (Blind Execute) function call:

   Type: [F5]pfunc bx[↵]

3. Type the command:

   Type: beep

4. Use PFUNC to enter either the XC (Execute) or the Q2 (Execute) function call, depending on the command you used in Step 2:

   Type: [F5]pfunc Q2[↵]

Result: When you run the program, the computer emits an audible signal at the point where the command appears.

Switching Windows to an Open File GOFILE

FORMAT C-XYW GOFILE *filename*

filename is the name of the file to which you want to move.

This is an immediate command.

MENU Not a menu item.

PURPOSE **GOFILE** switches to the window containing the named file.

ACTION **Switching Windows to an Open File**
To switch to the window that contains a file named MYFILE:

1. Put the cursor at the point in your program where you want to switch windows.

2. Use either the BC (Blank the Command Line) or the BX (Blind Execute) function call:

   Type: F5 pfunc bc↵

3. Type the command:

   Type: gofile myfile↵

4. Use either the XC (Execute) or the Q2 (Execute) function call, depending on the command you used in Step 2:

   Type: F5 pfunc XC↵

Result: When you run the program, execution switches to the window that contains the file named MYFILE at the point where the command appears.

FORMAT

C:\XY4 LDPM *d:programfile,#* *(Option 1)*
C:\XY4 LDPM *d:programfile* *(Option 2)*

d: is the letter of the drive that contains *programfile.*
programfile is the existing program file you want to load.
# *(optional)* is the single letter (A–Z) or number (0–9) or two characters &A–@Z or &0–&9 where you want to save the program file.

This is an immediate command.

MENU

Advanced Programming Save to Macro Key

PURPOSE

LDPM (Load Program) loads a program file either onto the specified macro key *(Option 1)* or into memory *(Option 2).* Option 1 enables you to run the program file with an F2 key (rather than with the RUN command). Option 2 enables you to RUN a program directly from memory rather than from disk.

You can assign programs to two different kinds of macros:

- Macros A–Z and 0–9. You normally run this kind of program from the keyboard using F2# (which accesses the function calls @A–@Z or @0–@9).
- Macros &A–&Z and &0–&9. You run this kind of program from any key where you have assigned the corresponding function call &A–&Z or &0–&9, or with the FUNC command (such as FUNC &A).

ACTION
(Option 1)

Loading a Program onto a Macro
To load a program onto a macro:

1. **Load the program.** To load the program file EXERCISE.PGM onto macro X:

   Type: F5ldpm exercise.pgm,x↵

   Result. The program file is copied to the macro (in memory). You can now run the program file EXERCISE.PGM by pressing F2X.

2. **Verify.** (Optional) To verify that the file has indeed been loaded onto that key:

   Press: Ctrl F2

   After viewing the text, press Esc to return to the document.

3. **Store the Macro.** (Optional) If you want to keep this program file loaded on the macro key for use at future editing sessions (after you quit), use the STSGT (Store Macros) command.

NOTE

Additional Macros. You can load programs onto any of up to 72 keys. This includes the 36 ordinary macros [F2]A through [F2]Z (Function Calls @A–@Z) and [F2]0 through [F2]9 (Function Calls @0–@9). It also includes additional programming macros assigned to function calls &A through &Z and &0 through &9.

ACTION
(Option 2)

Loading a Program into Memory

To load a program directly into memory (rather than onto a macro), use LDPM with only the filename of the program. For example:

Type: [F5]ldpm exercise.pgm[↵]

Result: The program file EXERCISE.PGM is now stored in memory. When you use the RUN command, XyWrite checks to see if the program you specify is stored in memory before going to the disk.

PFUNC Program Function Calls

FORMAT

C:XY4 PFUNC ##

## is a two-character function call.

This is an immediate command.

MENU

Not a menu item.

PURPOSE

The PFUNC (Put Function Call) command lets you enter a XyWrite function call into your file when not in program mode or when the function call is not assigned to a key.

ACTION

Using the PFUNC Command

Let's suppose you are creating a program in which you want to clear the command line, but leave the cursor in the text area. The function call for this action is CH (Clear Header), which is not typically assigned to a key. To enter it in your program:

Type: F5 pfunc ch ↵

Result: A bold CH appears in your program file. When you run the program, XyWrite will clear the command line without moving the cursor there.

Running a Program RUN

FORMAT

C:\XY\ RUN *d:programfile,string*

d: is the letter of the drive that contains *programfile.*
programfile (optional) is the name of the program file you want to run.
string (optional) is any information you want to pass to the program.

This is an immediate command.

MENU

Advanced Programming Run Macro

PURPOSE

RUN causes the specified program file to execute. This means the commands (and text) stored in the program file are executed automatically, as if typed from the keyboard. When you include information as an argument to the RUN command, XyWrite automatically stores the information in macro 00. You can access it by including «is00» or «pv00» in the program, depending on the type of information and how you want to use it.

ACTION

Running a Program File
To run a program file — for example, EXERCISE.PGM:

Type: F5 run exercise.pgm ↵

Result: This command runs the program file named EXERCISE.PGM — the keystrokes stored in that file are automatically executed.

To stop the program (if necessary):

Press: Ctrl Break

ALSO SEE

Related Commands. You can also run a program file with an F2 key. Refer to LDPM (Load Program) to load the program file onto a macro key.

NOTE #1

Shortcut. XyWrite remembers the name of the last file run (with the exception of STARTUP.INT). Therefore, if you enter the RUN command without specifying the name of a program file, XyWrite reruns the most recently run file. If you have modified the program and saved the changes to disk, XyWrite detects the changes and runs the current version.

NOTE #2

Running a Program from STARTUP.INT. If you place a program into STARTUP.INT to be run on startup (with RUN *filename),* end that program with «EX» rather than ending it with «EX1». Since STARTUP.INT is itself a program, your program is a subroutine within it. EX1 in the subroutine stops execution after the subroutine without returning to STARTUP.INT.

STSGT — Store Macro Keys to Disk

FORMAT

C:\XY4 STSGT *filename*

filename identifies the file on disk to which the macro keys will be stored.

This is an immediate command.

MENU

Insert Text from a Macro Options...

PURPOSE

STSGT stores the set of current macro keys to the specified file on disk. This enables you to reload the keys for use at a later editing session. You can also save several sets of macro keys and load each one for a different purpose.

ACTION

Storing Macro Keys to Disk

This procedure saves macro keys to disk.

1. **View the Macro Keys (Optional).** To view the text that will be saved to disk:

   Press: Ctrl F2

   After viewing the text, press Esc to return to the document.

2. **Disk Save (Long-Term Save).** To save to disk all of the keys viewed in Step 1:

   Type: F5 stsgt proposal.sgt ↵

In this case, PROPOSAL.SGT is the filename to which the keys are stored. You can save to any filename you want, but we recommend that you use the SGT extension, and store the file in the same directory as SIG.EXE. If you follow those conventions, XyWrite will include the new macro file when it creates a list of macros.

XPL Operators and Functions

PURPOSE

XyWrite provides operators and functions in the following categories:

- Arithmetic operators
- Logical operators
- Relational operators
- String operators
- String functions

Arithmetic Operators

Use these operators to perform arithmetic on numeric values.

+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus

Example: «SX99,«PV01»*10+«PV02»»

Logical Operators

Logical operators perform logical (or boolean) operations on numeric or string expressions. You use logical operators within IF statements.

!	Or
&	And
@XOR	Exclusive Or
@NOT	Not

Example: «IF(«IS99»==«IS01»)!(«IS99»==«IS02»)»

Relational Operators

These operators let you compare two numeric expressions (with PV) or two string expressions (with IS).

<	Less Than
>	Greater Than
<=	Less Than or Equal (same as =<)
>=	Greater Than or Equal (same as =>)
<>	Not Equal
==	Equal

Example: «IF(«PV01»<«PV02»)»

String Functions

String functions operate on string expressions and return a value or another string.

@UPR Converts letters to uppercase. (Parentheses are required.)

@CNV Converts a function call to its two-letter mnemonic. (see Note #1)

@SIZ Returns a value equal to the number of characters in a string. (Parentheses are required.)

@NUM Strips all punctuation and letters, leaving only digits. (Note: @NUM does not process parentheses. Do not use it on expressions that contain them.)

@INT Eliminates fractions after decimal point

Example: «sx03,@UPR(«is03»)»

String Operators

A string operator operates on two string expressions.

+ Concatenation. Links two or more separate strings. Example: «SX99,«IS01»+«IS02»»

≡ Contains. Determines if a string exists anywhere within another string. Example: «IF«IS01»≡«IS02»»«GLA»«EI»

∈ Contains. Determines if a string exists within another string and where within the string it exists. If the first string is found to be part of the second string, XyWrite returns a numeric value equal to the number of characters that *precede* the occurrence of the first string within the second: returns a value of 0 if the characters of the first string are at the beginning of the second string; returns a value of *n* if the first string appears within the second string but not at the beginning (*n* is the number of characters in the second string that precede the first string; returns a value of -1 if the first string is not in the second string. Example: «IF(«IS01»∈«IS02»==0)» returns 0 if macro 01 = A and macro 02 = ABCDE.

NOTE #1 **Converting Function Calls.** We recommend that you use the @SIZ function to test input to be sure it is a function call before using @CNV to convert the input to a 2-letter mnemonic. If @SIZ=3, then the input is a function call.

Examples

This section contains examples of the following procedures:

- Creating and running a user program
- Testing for a carriage return
- Evaluating keyboard input and branching

Creating and Running a User Program

As an example, the following procedure writes a program to count the number of characters in a file and display the result.

1. **Plan the Program.** In order to count the number of characters in a file, you press [Ctrl] [End] to move the cursor to the end of the file, use the CP (Character Position) command to count the number of characters from the beginning of the file, and save the result in a macro key. Since you are asking XyWrite to make an evaluation, you use the SX (Save Expression) command to save the result to a macro and the PV (Put Variable) command to display the contents on the screen.

2. **Create the Program File.** Use the NE (New) command to create a new program file. For example:

   Type: [F5]ne count.pgm[↵]

   Result: XyWrite creates a new (empty) file with the name COUNT.PGM.

3. **Write the Program.** After turning on program mode, type all the keystrokes necessary to perform the routine. When you are finished, turn program mode off.

   Press: [Scroll Lock] to turn program mode on.

   Press: [Ctrl] [End]

   Result: A bold BF appears on the screen (for Bottom of File).

   Press: [Scroll Lock] to turn program mode off.

   Type: [F5]sx[F9]

   Result: A text entry screen opens.

   Type: 01,[F5]cp[F9]

   Press: [Shift] [F1] to close the window.
   (If you want to abort the command window, press [Esc].)

Result: The SX command displays in expanded view as «SX01,«CP»».

Press: Scroll Lock to turn program mode on again.

Press: F5

Result: BC appears in the program file, which moves the cursor to the command line.

Press: Scroll Lock to turn program mode off again.

Type: F5pv 01 F9

Result: This will display the contents of macro 01 (the value acquired by the CP command) on the command line when the program runs.

In draft view, the program file looks something like this: BF ▲BC ▲

In expanded view, the program file contains code that looks like this:

```
BF «sx01,«cp»»BC «pv01»
```

The program code functions as follows:

BF (Bottom of File) moves the cursor to the end of the file.

«sx01,«cp»» saves the value of the current cursor location to macro 01. The CP (Character Position) command returns the number of characters from the beginning of the file.

BC (Blank Command Line) clears the command line.

«pv01» outputs the contents of macro 01 to the screen.

4. **Store the Program**

   Type: F5store↵

   Result: The program file COUNT.PGM is stored on the disk and disappears from the screen.

5. **Test the Program.** Test the program by calling a file in the current directory and counting the number of characters in it. To execute the program, call up any file and

   Type: F5run count.pgm↵

   Result: The program moves the cursor to the end of the file, counts the number of characters in the file, and displays the number on the command line.

6. **Load the Program onto a macro Key.** If you want to load COUNT.PGM onto a text key, refer to "Loading a Program" for information.

Testing for a Carriage Return

Testing for a carriage return requires two steps: (1) save a carriage return to a macro and (2) make an IF statement. The following example saves the carriage return to macro 91; and uses an IF statement to compare it to macro 00. Type this in with Scroll Lock turned off.

Type: F5sv↵

Type: 91,↵

Press: Shft F1

Result: The carriage return is saved to macro 91. In expanded view, the macro you just typed in looks rather odd, but nonetheless is correct:

```
«SV91,←
»
```

Then type in the following IF statement:

```
«IF«IS00»∈«IS91»==0»
```

If the content of macro 00 is a carriage return, this statement will be true.

Evaluating Keyboard Input and Branching

You can write programs that will pause in the middle of execution, ask a question and wait for you to respond. Therefore, you can stop and make choices. For example, you can modify your STARTUP.INT file to choose which printer file to load. The following program demonstrates the RC (Read Character) command and the IF statement. RC causes the program to pause, allowing you to select a printer by pressing a letter (D or L).

```
«LB-A»«PRDot matrix (D) or laser (L)?»«SX99,«RC»»
«IF«VA$KC»==38»BX setp 1Q2 «PRConfigured for laser printer.»
«EX»«EI»«IF«VA$KC»==32»BX setp 2Q2«PRConfigured for dot matrix
printer.»«EX»«EI»BX beepQ2«PRPress D or L»BX p Q2«GL-A»
```

Variables

INTRO

Variables indicate the status of XyWrite. You can check on the current value of any default setting, as well as many other system settings (for example, the name of the currently loaded keyboard file) that affect your use of XyWrite.

CONTENTS

Value of Variable

FORMAT **C:XY3 VA/nv** *nm*

/nv is an optional switch that displays the value on the status line.
nm is a variable

MENU Not a menu item.

PURPOSE **VA** obtains the current value (or string) of the variable you request (*nm*) and inserts it into the text. If you include the optional /NV switch, XyWrite displays the current value on the status line rather than in the text (see Note #1).

The VA command is useful for determining the current value of default settings. However, it is also a valuable troubleshooting tool. You can use the VA command to check the status of:

- Default settings, including format settings
- System settings (using the $ prefix)
- Text macros (using the @ prefix)
- Document Information items (using the ^ prefix)
- Display Modes (using the * and [prefixes)
- Keyboard location (using the _ prefix)

To obtain the value of a default setting, you simply issue the VA command with the default variable. For example, VA LM obtains the current Left Margin setting. Refer to "Default Settings" for a complete list of variables you can use. To obtain values for the other categories, you use the command forms described on the following pages.

ACTION **Obtaining a Value**
To get the current value of a XyWrite setting, say tabs (TS):

1. Move the cursor to the point in text where you want to find the current tab settings.
2. Enter the VA command on the command line. For example:

   Type: [F5]va ts[↵]

Result: The display will show the tabs settings at that location in the text:

▲.5,1.5,2.5,3.5

If you don't want these values to print, delete the marker before issuing the PRINT command.

NOTE #1 **Status Line Display.** If you use the /NV switch to display the value on the status line, cursor placement is important. If the cursor is on the command line when you press [F9], XyWrite displays the current default value of the variable. If the cursor is in the text, XyWrite displays the value in effect at that point in the file.

NOTE #2 **No File Open.** If you do not have a file open when you issue the VA command, XyWrite automatically displays the result on the status line — you do not need the /NV switch.

NOTE #3 **Separate Values.** If the variable you are checking includes more than one value, the VA command displays all the values. If you prefer, you can display only one value by including the numeric position of the value you want. For example, VA IP displays both values of the IP setting; VA IP1 displays only the first value; VA IP2 displays only the second value.

VA	Description	Value Returned (Example)
va $ac	Auto-Check	0
va $am	Available Modes	0
va $bd	Bad Words	10
va $bq	Border Query	0
va $bt	Black and White Trace	1
va $bw	Black and White Monitor	1
va $ca	Cartridges	WP
va $cm	Command Line	se /XyWrite/
va $cn	Cartridge Installed	2
va $co	Columns	3
va $cp	Code Page	437
va $cx	Cursor Column Position	40
va $cy	Cursor Row Position	20
va $dc	Define Column	0
va $de	Define Ended	0
va $df	Define Status	0
va $di	Directory Type	4
va $dl	Defaults	C:\XY4\SETTINGS.DFL
va $dn	Define End	100
va $do	Document Information	1
va $dp	Display Path	C:\DRAFT\CHAPTER1
va $dr	File in Directory	C:\MEMOS\VACATION
va $ds	Define Start	32
va $dt	Display Type	1
va $dv	Default Drive	C
va $ed	EXE Location	C:\XY4\XY4.EXE
va $er	Error Number	102
va $et	Elapsed Time	54
va $ex	File Extension	TMP
va $fb	File Begin	0
va $fe	File End	1
va $fl	Filename	CHAPTER
va $fm	Forms Mode	0
va $fp	Filename and Path	C:\DRAFT\CHAPTER1
va $fs	File Status	0
va $fx	Fixed Pitch	1
va $fz	Field Separator	◙
va $gc	GCI (Generalized Callable Interface) Status	0
va $hl	Help File	C:\XY4\XY4.HLP
va $hp	HP Printer File	0
va $in	Inside Define	0

VA	Description	Value Returned (Example)
va $kb	Keyboard File	C:\XY4\XY4.KBD
va $kc	Key Code	82
va $lg	Logged-On User	Rose
va $ln	Current Line Number	22
va $lt	Logon Notes Toggle	0
va $me	Memory Available	286
va $mn	Menu File	C:\XY4\XY4.MNU
va $mo	File Modified	1
va $mw	Mouse Window	6
va $mx	Mouse X Position	33
va $my	Mouse Y Position	12
va $m+	Memory Used By XyWrite	563
va $na	Non-Printable Area	.2,.3,.5,.3
va $nr	No Ruler	0
va $nu	Unused Printer Memory	2395
va $ov	Overflow File	C:
va $pa	Current Drive and Path	C:\WORKING
va $pc	PC Code	0
va $pf	Printer File Selection	LPT1 D:\XY4\POST35N.PRN
va $pg	Current Page	3
va $pr	Printer File	C:\XY4\POST35N.PRN
va $px	PC Code Explanation	Letter (8.5" x 11")
va $p?	PC Code Exist	0
va $p.	Truncated Path	\SI
va $p\	Path No Backslash	C:\XY4
va $ra	Read Attribute	27
va $rc	Resume Code	0
va $re	Read-Only File	1
va $rk	Record Keystrokes	1
va $rl	Redlining	0
va $rr	Return Error	
va $rs	Read Character	A
va $rz	Record Separator	←
va $sc	Scan Code	276
va $sd	Sort Directory	d,r
va $se	Search String	auto
va $sf	Soft Font List File	PS140b10.USP
va $sg	Text Macro File	C:\XY4\SAVE.GET
va $sm	Show Menus	0
va $so	Sort Setting	F1
va $sp	Spelling Dictionary	C:\XY4\PERS.SPL
va $ss	Styles	HEAD1
va $ss	Save Styles	DEFAULT
va $st	Startup	C:\XY4\STARTUP.INT
va $te	Type Effect	0
va $tr	Triangle Mnemonic	LM
va $tw	Text Window	0
va $tx	Cursor Location	0
va $uf	Use Typeface	TIMES
va $uo	Use Outline	0

VA	Description	Value Returned (Example)
va $ve	Version Number	1.0
va $wa	Window Availability	5
va $wc	Word Count	3782
va $wn	Window Number	3
va $wo	Windows Open	4
va $ws	Window Status	0
va *	Mode Number	15
va @	Text Macro	YES
va ^	Document Summary Items	07-08-91
va [	Additive Mode On	1
va |	Size of Macro	12
va §	Macro Contains Number	1
va _	Assigned Key	Alt+F7

DETAIL

This section contains a description of the variables listed above. These variables let you perform a variety of tasks, like saving the value of a setting so you can later restore it. Many of the VA settings are designed to be inserted within user programs.

VA $AC

Auto-Check — Indicates the status of Auto-Check (0=off, 1=on).

VA $AM

Available Modes — Indicates the availability of the specified mode in the current type family. The format is:

va $am*md*, where *md* is either BO, BI, or IT. XyWrite displays a 1 if the mode is available.

VA $BD

Bad Words — Displays the number of questionable words found by the SPELL command when it was last executed.

VA $BQ

Border Query — Indicates whether a particular border definition is present in the file. The format is:

va $bq*bo*, where *bo* is the name of a custom or standard border. XyWrite displays a 1 if the border definition is present, a 0 if it is not.

VA $BT

Black and White Trace — Indicates the value of the BW setting in the printer file. If you have a CGA monitor and set BW=1 in the printer file to get rid of "snow," $BT returns a value of 1. If BW=0 in the printer file, $BT returns a value of 0. If BW is not set in the printer file, $BT returns a non-zero.

VA $BW **Black and White Monitor** — Indicates the monitor type (0=monochrome; 1=color).

VA $CA **Cartridges** — Displays the printer cartridges currently loaded.

VA $CM **Command Line** — Displays the current contents of the command line.

VA $CN **Cartridge Installed** — Indicates whether or not a cartridge or a soft font is installed. The following values are reported:

0 No soft fonts or cartridges
1 Cartridges from the printer file
2 Soft fonts from the printer file
3 Cartridges and soft fonts from the printer file
4 HP printer file is loaded

VA $CO **Columns** — Displays the number of columns in use at the cursor.

VA $CP **Code Page** — Returns the value of the current code page.

VA $CX **Cursor Column Position** — Displays the column number (horizontal position) of the cursor.

VA $CY **Cursor Row Position** — Displays the row number (vertical position) of the cursor.

VA $DC **Define Column** — Indicates whether a column of text is currently selected (0=no, 1=yes).

VA $DE **Define Ended** — Indicates whether the select text sequence has been completed (0=no, 1=yes).

VA $DF **Define Status** — Indicates whether there is selected text in the current window (0=no, 1=yes).

VA $DI **Directory Type** — Indicates the type of directory currently displayed. The following values are reported:

1 Tree directory
2 Names-only directory
4 Normal directory
8 Long directory
64 Soft font directory
128 Cartridge directory

VA $DL **Defaults** — Displays the drive and path of SETTINGS.DFL.

VA $DN **Define End** — Displays the character position at the end of the selected block.

VA $DO — **Document Information** — Indicates whether document summary information is attached to the displayed file (0=no, 1=yes).

VA $DP — **Display Path** — Displays drive, path, and filename of the current file.

VA $DR — **File in Directory** — Displays the name of the file that is currently highlighted in the directory.

VA $DS — **Define Start** — Displays the character position at the beginning of the selected block.

VA $DT — **Display Type** — Indicates the current view (as opposed to the default setting, which is obtained with VA DT). The following values are reported:

0	Expanded view
1	Draft view with markers but no page breaks
2	Draft view with page breaks and markers
4	Graphic view with markers
9	Draft view without markers or page breaks
10	Draft view without markers but with page breaks
12	Graphic view without markers

VA $DV — **Default Drive** — Displays the letter of the current drive (without :\).

VA $ED — **EXE Location** — Displays the drive and subdirectory of XY4.EXE.

VA $ER — **Error Number** — Displays the error number of the last error that occurred.

VA $ET — **Elapsed Time** — Displays the amount of time that has elapsed since you issued the ZT (Zero Time) function call. If you have not issued the ZT function call since starting XyWrite, $ET displays the current time.

VA $EX — **File Extension** — Displays the extension, if any, of the current file.

VA $FB — **File Begin** — Indicates whether the cursor is at the beginning of the file (0=no, 1=yes).

VA $FE — **File End** — Indicates whether the cursor is at the end of the file (0=no, 1=yes).

VA $FI — **Filename** — Displays the name of the current file.

VA $FM — **Forms Mode** — Indicates whether or not you are in forms mode (0=no, 1=yes).

VA $FP **Filename and Path** — Displays the drive, path, and name of current file.

VA $FS **File Status** — Displays the status of files (0=no files are open, non-zero=at least one file is open).

VA $FX **Fixed Pitch** — Indicates whether the current font is fixed pitch or proportional (0=fixed pitch, 1=proportional).

VA $FZ **Field Separator** — Indicates the field separator as XyWrite displays it, rather than the 3-byte sequence XyWrite uses to store the field separator internally, which is the value returned by VA FX.

VA $GC **GCI (Generalized Callable Interface) Status** — Indicates whether XyWrite has been invoked by a program using the GCI protocol. You might use the variable if you are logged on to a host machine and you are running XyWrite as a TSR (Terminate and Stay Resident) program. The format for $GC is:

va $gc*n* where *n* is one of the following arguments:

1 Returns 1 if GCI is running; 0 if it is not.
2 Returns 1 if a merge option from the host is available; 0 if it is not.
3 Returns 1 if a save or store option is available; 0 if it is not.
4 Returns 1 if printing to the host is available; 0 if it is not.

If you do not specify *n*, values for all four arguments are displayed.

VA $HL **Help File** — Displays the name and location of the current help file.

VA $HP **HP Printer File** — Indicates whether the current printer file is for a Hewlett Packard LaserJet (0=no, 1=yes).

VA $IN **Inside Define** — Indicates whether the cursor is within a selected block (0=no, 1=yes).

VA $KB **Keyboard File** — Displays the name and location of the current keyboard file.

VA $KC **Key Code** — Displays the key code number of the last key pressed.

VA $LG **Logged-On User** — Displays the name of the user currently logged on.

VA $LN **Current Line Number** — Displays the line number where the cursor is currently located. This works only when Page-Line numbers are displayed.

VA $LT Logon Notes Toggle — Indicates whether text display of captured redlining login information is suppressed (0=off; 1=on).

VA $ME Memory Available — Shows the amount of memory (RAM) currently available, in kilobytes, after XyWrite is loaded. Thus, 312 means 312K of memory is available for more files or other programs.

VA $MN Menu File — Displays the name and location of the current menu file.

VA $MO File Modified — Indicates whether the current file has been modified since being opened (0=no, 1=yes).

VA $MW Mouse Window — Displays the number of the window in which the mouse is currently located.

VA $MX Mouse X Position — Displays the pixel position of the mouse in graphic view; displays the row position in draft view.

VA $MY Mouse Y Position — Displays the pixel position of the mouse in graphic view; displays the column position in draft view.

VA $M+ Memory Used by XyWrite — Displays the amount of memory currently used by XyWrite, expressed in kilobytes.

VA $NA Non-Printable Area — Displays the portions of the page the printer does not use when printing. For example, laser printers do not begin printing at the very top of the page. $NA displays four values to indicate the left, right, top, and bottom non-printable areas. You could include $NA in a program to ensure that files are portable to different types of printers.

VA $NR No Ruler — Indicates whether a ruler is displayed at the top of the screen (0=no, 1=yes).

VA $NU Unused Printer Memory — Displays the amount of memory currently available in the HP LaserJet. This value reflects any soft fonts that have been downloaded.

VA $OV Overflow File Drive — Displays the location of the overflow file, if one exists.

VA $PA Current Drive and Path — Displays the current drive letter and path.

VA $PC **PC Code** — Indicates if there are any PCs in the current PC (Printer Control) table within the range of the number specified (0=no, 1=yes). The format of $PC is:

> **va $pc#** where # is a PC number.

VA $PF **Printer File Selection** — Displays the current SETP selection.

VA $PG **Current Page** — Displays the page number where the cursor is currently located. This works only when Page-Line numbers are displayed.

VA $PR **Printer File** — Displays the name and location of the current printer file.

VA $PX **PC Code Explanation** — Displays a description of the specified PC. *VA $PX can only be used in a dialog box.* The format of $PX is:

> **va $px#** where # is a PC number.

VA $P? **PC Code Exist** — Indicates whether the specified PC code exists in the current PC table (0=no, 1=yes). The format of $P? is:

> **va $p?#** where # is a PC number.

VA $P. **Truncated Path Name** — Displays the truncated path name as established by the P. default.

VA $P\ **Path No Backslash** — Indicates the current drive letter and path the same as $PA does, with the exception that $P\ displays a backslash in the root directory, whereas $PA does not.

VA $RA **Read Attribute** — Displays the number of the current type style. This value only reflects absolute character modes, not additive modes.

VA $RC **Resume Code** — Indicates which of the valid keystrokes a user pressed in response to a WM (Wait Message) or BR (Break) command (0=first valid key, 1=second valid key, etc.; 65533=Enter (on numpad); 65534=F9 or Ctrl Break; 65535=Esc).

VA $RE **Read-Only File** — Indicates whether the current file is a read-only file (0=no, 1=yes).

VA $RK **Record Keystrokes** — Indicates whether Record Keystrokes mode is on or off (0=off, 1=on).

VA $RL **Redlining** — Indicates whether redlining is on or off (0=off, 1=on).

VA $RR **Return Error** — Displays any errors generated while running a program in DOS via the DO command.

VA $RS **Read Character** — Displays the character under the cursor.

VA $RZ **Record Separator** — Indicates the record separator as XyWrite displays it, rather than the 3-byte sequence XyWrite uses to store the record separator internally, which is the value returned by VA RS. $RZ is used primarily in programs to search for a carriage return/line feed. Although carriage returns are represented on the screen as a single character (←), they are actually stored internally in a set as carriage return and line feed.

VA $SC **Scan Code** — Displays the value of the next key pressed, plus the value of any current shifted state (Caps Lock = 256, Shift = 512, Alt = 1024, Ctrl = 2048, NumLock = 32768). For example, if ? and Caps Lock are pressed, $SC displays 275 (19 + 256). (Note: If you change the order of the tables in the definition section of the keyboard file, then the values associated with each table change accordingly.)

VA $SD **Sort Directory** — Displays current DSORT (Directory Sort) settings.

VA $SE **Search String** — Displays last search string.

VA $SF **Soft Font List File** — Displays the name of the soft font list file loaded into memory.

VA $SG **Text Macro File** — Displays the name and location of the current text macro file.

VA $SM **Show Menus** — Indicates whether a menu is currently displayed (0=no, 1=yes).

VA $SO **Sort Setting** — Displays the value of the SO (Sort) setting.

VA $SP **Spelling Dictionary** — Displays the name and location of the current personal spelling dictionary.

VA $SS **Save Styles** — Displays the names of the styles defined in the current file. The format is:

va $ss*n*, where *n* is a number that refers to the order in which the style was defined. For example, VA $SS1 displays the name of the first style defined in the file; VA $SS2 displays the name of the second style defined; etc. If *n* is omitted, displays the name of the current style.

VA $ST **Startup** — Displays the drive and path of STARTUP.INT.

VA $TE **Type Effect** — Displays the status of the different EF (Special Effect) settings (0=off, 1=on). The format is:

va $te*n*, where *n* is the number of the printer effect. (See the description of the EF command in the *Command Reference Guide* for more information.)

VA $TR **Triangle Mnemonic** — Displays the mnemonic of the embedded command on which the cursor is located.

VA $TW **Text Window** — Indicates whether a command window is currently open (0=no, 1=yes).

VA $TX **Cursor Location** — Indicates whether the cursor is in the text area or in the header (0=header, 1=text).

VA $UF **Use Typeface** — Displays the name of the current typeface.

VA $UO **Use Outline** — Indicates whether the printer supports outline fonts (0=no, 1=yes).

VA $VE **Version Number** — Displays the version number of XyWrite that is currently loaded.

VA $WA **Window Availability** — Displays the number of the next available window that XyWrite will use when a CALL or NEW command is executed. (The NW default must be set to 1, 2, or 3.) SWA displays 0 if all 9 windows are in use.

VA $WC **Word Count** — Displays the total number of words counted by the WC, WCB, or SPELL command (whichever was most recently executed).

VA $WN **Window Number** — Displays the number of the window that is currently active (0-9).

VA $WO **Windows Open** — Displays the number of windows currently open.

VA $WS **Window Status** — Displays the status of specified window. The format is:

va $ws*n*, where *n* is the window you want to check. The following values are returned:

0	No file open, window is empty.
1	File is open.
2	Directory is displayed.

VA *

Mode Number — Displays the color number of the specified display mode. The format is:

va **md* where *md* is the display mode.

VA @

Text Macro — Displays the contents of the specified text macro. The format is:

va @# where # is the text macro you want to display.

VA [

Additive Mode On — Indicates whether the cursor is in text that uses the specified additive mode (0=no, 1=yes). The format is:

va [*md*, where *md* is the mode (for example, va [bo).

VA ^

Document Summary Items — Displays the contents of the specified document summary item. The format is:

va ^*mn* where *mn* is one of the following:

au	Author
lg	Logon of person who last modified the file
cd	Creation date
cm	Comments
rv	Revision number
ky	Keyword
rp	Retention period
pj	Project number

VA ¦

Size of Macro — Displays the size of the macro. XyWrite returns 65535 if macro does not exist; 0 if macro exists but is empty. The format is:

va ¦#, where # is the macro.

VA §

Macro Contains Number — Indicates whether the contents of the specified macro is a number (0=no, 1=yes). The format is:

va §#, where # is the macro.

VA _

Assigned Key — Displays the key to which the specified function is assigned in the current keyboard file. The format is:

va *_function*

where *function* is the function call, character assignment, JM command, etc. for which you want the key assignment.

This command requires a frame called 1A in the dialog box file. This frame, which lists the keycodes and the key labels, is in XY4.DLG you received with XyWrite. If you create a new DLG file and want to use this VA command, be sure to include the 1A frame in your file.

VA \

Error Message — Displays the message associated with the specified error number. The format is:

va \nnn

where *nnn* is an error number.

Appendix A Advanced Keyboard Functions

INTRO

The standard XyWrite keyboard file, XY4.KBD, includes several key assignments that are considered to be "advanced." Some of these keys provide diagnostic tools that you will find to be useful as you start customizing XyWrite. Other keys are just handy shortcuts that you can use in your everyday work with XyWrite. The table on the next page describes the advanced key set.

Keys	Function
Ctrl Alt Shift A	**Alternate Menu.** Loads an alternate menu file or runs a program. Refer to the notes section at the beginning of XY4.MNU for more information about this feature (search for the word "Developers:" to move quickly to this information). Note: If you are loading both a MNU file and a DLG file, then you must have alternates for both.
Ctrl Alt Shift B	**BC.** Inserts the BC function call at the cursor location.
Ctrl Alt Shift C	**Copy.** Copies the currently selected text to the command line.
Ctrl Alt Shift F	**Find Latest.** Displays a list of files sorted by date, with the most recent files at the top. You can use this key to display a list of files in the current directory, in the current directory and its subdirectories, or on the current disk. This is useful if you have lost a file that you recently created and don't remember its exact name.
Ctrl Alt Shift K	**Keyboard Diagnostics.** Helps you customize keyboard files. To use this key, first display the keyboard file (XY4.KBD, for example) and press Ctrl Alt Shift K. XyWrite presents three options: (1) you can identify the key code (1-104) associated with the next key you press; (2) you can move the cursor onto a key definition and identify the key associated with that code; or (3) you can jump to one of the tables in the keyboard file.
Ctrl Alt Shift L	**Load a File.** Displays a dialog box that gives you four options: (1) you can load the displayed file; (2) save the displayed file and then load it; (3) copy the displayed file to a temporary file named LOAD.TMP and then load the temporary file; or (4) load a file other than the one currently displayed.
Ctrl Alt Shift M	**Move.** Moves the selected text to the command line.
Ctrl Alt Shift N	**Next File.** When you issue the Call command with a global filename (for example, CALL *.DOC), this key combination displays the next file that matches the specification. You can edit the file (if you wish) and press this key combination again to store the current file and open the next one.

(cont'd)

Advanced Keyboard Functions

Keys	Function
Ctrl Alt Shift P	**Print Expanded.** Prints the current file in expanded view, with embedded commands printed in angle brackets. This key is useful for printing a hard copy of program files. It is also useful for diagnosing embedded commands in any document.
Ctrl Alt Shift Q	**Quick Build.** Displays a dialog box for verification. This key is useful when customizing menus, because it lets you verify a new or revised dialog box without saving and reloading the menu file. To use this key, display the menu file, create or edit a dialog box (type K frame), move the cursor anywhere past the keyword, and then press Ctrl Alt Shift Q.
Ctrl Alt Shift R	**Resume.** Displays a dialog box with two choices: Save settings and Restore settings. When you save settings, XyWrite creates an XPL program containing information about every window you have open: window number, filename, view (draft, graphic, etc.), and cursor position. If you had a macro file in effect, information about it is captured, too. When you restore settings, XyWrite runs the XPL program, calling each file, putting it into the appropriate view, and restoring its latest cursor position. Your macro file is loaded into memory too. Resume can be used to capture the status of your work before you store it to disk at the end of the day; when you restore settings, XyWrite returns you to where you left off. Window sizes are currently not recorded or restored.
Ctrl Alt Shift S	**Status.** Displays a dialog box that lists the currently loaded XyWrite custom files. It also gives you the option of loading/running or opening one of the custom files.
Ctrl Alt Shift T	**Time/Date.** Displays a message box that contains the current date and time.
Ctrl Alt Shift V	**Values.** Displays a list of formatting values in effect at the current cursor location.
Ctrl Alt Shift X	**Programming Diagnostics.** Displays a dialog box you can use to enter function calls into your program files or to turn on the DB (Debugging) default setting.

Keys	Function
Ctrl Alt Shift Z	**Last Commands.** Displays the result of the last commands executed through the menus. For example, when you import text into a file, Last Commands will show you the command that actually created the link. This feature has been implemented only for some of the more complex operations, such as importing text, and for default commands.
Ctrl Alt Shift ?	**Help.** Displays a list and brief description of the advanced keys.

Appendix B Terminate & Stay Resident Programs

INTRO

Terminate and Stay Resident (TSR) programs are programs that are loaded into memory and can be instantly activated by pressing a hot key. For example, you might run a calendar TSR program that keeps track of your appointments; you don't need the program to be active all the time, but you want to be able to access it quickly so you can check your schedule. 3270 emulation programs are another common example of TSR programs; the emulation programs allow you to communicate with mainframe computers. This appendix describes the special setup needed to avoid conflicts between XyWrite and TSR programs.

Terminate & Stay Resident Programs (cont'd)

PURPOSE

To use a TSR program with XyWrite, you must complete a few simple steps. Otherwise, you may find that the TSR interferes with the XyWrite operation. The steps are:

1. Determine that you have enough memory.
2. Modify your XyWrite keyboard file.
3. Load the TSR and XyWrite.

These steps are explained in detail below. Note that the procedures assume that both the TSR and XyWrite are already installed on your computer.

ACTION

Evaluating Memory Requirements

To determine if you have enough memory to run a TSR and XyWrite simultaneously:

1. Determine how much conventional memory is available on your computer. If you aren't sure, there are utility programs available that will give you information about your hardware setup.
2. Subtract 384KB from the available memory. XyWrite requires a minimum of 384KB (512KB is recommended).
3. Compare the result of step 2 with the memory requirements of the TSR.

Result: If there is enough memory, go to the next procedure. Otherwise, you cannot run both programs simultaneously.

ACTION

Modifying the XyWrite Keyboard File

TSR programs are activated by pressing a hot key. To accommodate the hot key, you need to modify the XyWrite keyboard file. There are two different modifications, depending on how the TSR program interacts with the keyboard.

If the TSR program uses a different keyboard map than XyWrite or changes the keyboard interrupt handling in any way, you need to assign the KF function call to the hot key (see Note #1). The procedure is:

1. Open the XyWrite keyboard file you want to modify. For example:

   Type: F5 call xy4.kbd ↵

2. Refer to the diagrams at the top of the file to determine the key code of the hot key.

3. Search for the table you want to modify. For example:

   Type: F5se /table=alt/↵

4. Search for the key code of the hot key. For example, if the hot key is Backspace:

   Type: F5se /14/↵

5. After the equals sign, type the KF (Keyboard Flip) function call. The line will look like this:

   14=KF←

6. Store and load the keyboard file.

If the TSR program doesn't modify keyboard handling, you don't need the KF function call. Instead, assign a null character to the key. The procedure is:

1. Perform steps 1-3 from the previous procedure.

2. After the equals sign, type a null character.

   Type: Ctrl Alt 0

   Result: A blank space appears after the equals sign.

3. Store and load the keyboard file (see Note #2).

ACTION

Loading the TSR Program and XyWrite

The last step in the setup process is to load the TSR program and XyWrite. The important thing to remember about this step is that the TSR must be loaded into memory *before* you start XyWrite. The proper sequence is: quit XyWrite, run the TSR program, then run XyWrite.

If the TSR and XyWrite are part of your standard setup, you may want to modify the AUTOEXEC.BAT so that it automatically loads the TSR before XyWrite. To do that:

1. Open AUTOEXEC.BAT.

   Type: F5call \autoexec.bat↵

2. Search for the line that loads XyWrite.

   Type: F5se /editor/↵

3. Immediately *above* the EDITOR line, enter the command to load the TSR.

4. Store the file.

5. Reboot the computer.

Result: Both programs are now in memory. You can activate the TSR by pressing the appropriate hot key.

NOTE #1 **Predefined Hot Keys.** The standard XyWrite keyboard file, XY4.KBD, preassigns the KF function call to Alt Esc, Ctrl Esc, and Alt Tab. If the TSR you want to run is activated by one of these hot keys, you don't have to modify the keyboard file. Alternatively, if you don't need these hot keys, you can assign a different function call to one or more of them.

NOTE #2 **Startup.** You can have your keyboard file automatically loaded at startup by adding a line to your STARTUP.INT file. For example, the following entry loads a keyboard file named XY4.KBD.

```
BC load xy4.kbd←
```

Refer to "Startup File" for more information on modifying STARTUP.INT.

Appendix C Foreign Language Support

INTRO

XyWrite was designed to be used in a variety of languages, as indicated by its extensive character set and its support of foreign-language spelling, hyphenation, and synonyms. This appendix describes two XyWrite functions that enhance the foreign language support: the UC (Uppercase) string and the Accent Table.

CONTENTS

Uppercase String UC

FORMAT UC<*nNnNnN*...

n is the lowercase version of a character
N is the uppercase version of a character

MENU Not a menu option

PURPOSE XyWrite uses 191 standard uppercase/lowercase character pairs when it executes the UC (Uppercase), LC (Lowercase), CC (Change Case), and CF (Capitalize First) commands. These character pairs are built into XyWrite, but if you want to customize them, you can enter the UC string into SETTINGS.DFL.

When you enter the UC string, *you must include all the characters you want XyWrite to recognize as pairs,* not just the ones you are adding. Because of this, we recommend that you get a soft copy of the built-in character pairs from Technical Services and use that as a basis for the new string.

ACTION

Modifying a UC String

To create a customized UC (Uppercase) string, you should first request a copy of the standard string from Technical Services. Then:

1. Call SETTINGS.DFL.

   Type: [F5]call settings.dfl[↵]

2. Move the cursor to the end of the file.

   Press: [Ctrl] [End]

3. Merge the file that contains the default UC string. For example:

   Type: [F5]merge case[↵]

4. Make the desired modifications to the UC string in SETTINGS.DFL.
5. Store the file.

   Type: [F5]store[↵]

6. Load SETTINGS.DFL.

   Type: [F5]load settings.dfl[↵]

NOTE Errors. Even if you input the UC string incorrectly, you may still be able to load SETTINGS.DFL successfully. If you get the message "Command is not recognized" when entering standard XyWrite commands, you may have omitted an uppercase/lowercase pair from the UC string. Similarly, if you incorrectly input a pair, the search and change case commands will not function properly. Should you encounter such errors, review the UC string and make the appropriate corrections.

* a-z pairs must be included.

AC Accent Table

trigger

TRIGGER
0-9
\/+-( ). '' |

FORMAT

AC:*n*
string1 = \acc AA
string2
.
.
.
stringn

n is the number of lines in the table.
string1...stringn defines the floating accents and the characters associated with them.

MENU

Not a menu option

PURPOSE

The XyWrite character set has an extensive selection of accented characters. Most of these characters are not assigned to keys. If you are working in a foreign language, you can remap the keyboard so that you can easily access the characters you use most frequently. Otherwise, you can enter the accented characters by pressing Ctrl Alt and typing the character number or using the Insert Other option from the Options menu.

The AC (Accent) Table gives you another alternative for entering accented characters. The AC Table defines floating accents and the characters associated with the accents. To use the information in this table, you press the key assigned to the floating accent and then the key of the character you want to accent. XyWrite automatically enters the accented character in your document.

This method saves you from the burden of remembering character assignments and minimizes the number of keys you have to dedicate to accented characters.

XyWrite has a built-in AC Table that defines the following floating accents:

Acute (S1)
Grave (S2)
Umlaut (S3)
Circumflex (S4)
Ring Overdot (S5)
Tilde (S6)
Macron Underline (S7)

SP
8 Cedilla (XX,,)),
9 Stroke/slash (XX,/) /,
10 Double acute (XX,") ''
~~Macron (XX,=)~~
11 Ogonek (XX,.) (
12 Caron (XX,-) \
13 Breve (XX,-) !
14 Dot accent .

Each accent has a function call associated with it, and the function calls need to be assigned to keys in order for the information in the AC Table to be in effect. The function calls are shown in parentheses above, and the procedure for modifying the keyboard file is provided below.

ACTION

Modifying the Accent Table

The built-in AC Table includes all the accented characters in XyWrite's character set. It is unlikely that you will ever need to modify it, but the procedure below explains the necessary steps should you find a reason to. To create a customized AC Table, you should first request a copy of the standard table from Technical Services. Then:

1. Call SETTINGS.DFL.

   Type: [F5]call settings.dfl[↵]

2. Move the cursor to the end of the file.

   Press: [Ctrl] [End]

3. Merge the file that contains the default AC Table. For example:

   Type: [F5]merge accents[↵]

4. Make the desired modifications to the AC Table in SETTINGS.DFL. To define a new floating accent:

   - Go to the end of the table.
   - Type the accent, a comma, the *unaccented* version of the first character, and then the *accented* version of the character.
   - Type all the character pairs (unaccented followed by acented) associated with this accent.

5. Store the file.

   Type: [F5]store[↵]

6. Load SETTINGS.DFL.

   Type: [F5]load settings.dfl[↵]

Result: The AC Table is loaded in memory, but you must complete the following procedure before you can use the floating accents.

ACTION

Modifying the Keyboard File

You must modify the keyboard file in order for XyWrite to recognize floating accent characters in the AC Table. If you want to use all the floating accents, you may want to add a new table to the keyboard file so that you have an entire set of keys from which to select the key assignments. To add a new table for the floating accents:

1. Decide which keys you want to assign to the floating accents. For example, you might assign the acute accent to A, the grave accent to G, etc.

2. Call your keyboard file to the screen. For example:

   Type: [F5]call xy4.kbd[↵]

3. Refer to the diagram at the top of the file. Make a note of the key codes for the keys you selected in step 1.
4. Move to the end of the file.

   Press: Ctrl End
5. Enter the first line of the new table. For example, to enter a table for CTRL+CAPS:

   Type: TABLE=CTRL+CAPS←
6. Enter the assignment for the first key. For example, to assign the acute accent (function call S1) to key A (key code 30):

   Type: 30=s1←
7. Repeat step 6 for each floating accent.
8. Store and load the keyboard file.

Result: You can now use floating accents to create accented characters. For example, if you assigned the acute accent to key A and entered want to enter

NOTE

Printer Output. The AC table affects the screen display only. If the accented character is not part of the printer's standard character set, you must make the appropriate entry in the character substitution file.

NOTES

Appendix D Reserved XyWrite Elements

This appendix lists ID numbers, keywords, and macros reserved for use by XyWrite.

Reserved Keywords

The following keywords, which can be used only with Type K frames, are reserved for special functions in the XyWrite menu file:

/g	Macro directory
/m	Memory menu
/s	Spelling menu
/h	HP menu to decode soft font files
/t	Expanded names for commands
/y	Thesaurus
/u	File summary

If you use one of these reserved keywords, no other keywords can appear between the pair of double curly braces {{ }} in the label.

Reserved ID Numbers

The following control ID numbers are reserved for special functions.

1	Typically the OK pushbutton
2	Typically the CANCEL pushbutton
50	Remove item from printer queue
51	Move item to top
52	Stop printing
53	Resume printing
54	See JB codes in PC menu
55	Set JB codes
56	See PB codes
57	Set PC codes
58	Rewind printer queue
59	Look up synonyms
60	Return to original synonyms
61	Print a directory (directory must be in an LF box)
62	HELP pushbutton for a dialog box
98	Path that gets updated
90-99	Don't interpret ampersand (&) as an accelerator
100	Text box where path is located. Used with /LS, so user can switch directories from text box and still allow /LS to work.
251-256	Don't strip trailing spaces (used in Search and Replace)

RESERVED MACROS

100-999	Reserved for user; they must not be used in menu programs.
1000-1399	Unreserved - can be used by any item
1400-1499	Reserved for third-party developer's permanent (session) storage.
1700-1799	Reserved for future expansion.
1800-1899	Reserved temporarily for operations performed in this session only.
1900-1999	Reserved for use by XyWrite; currently they have the following uses:

1900 -	Dir for documents (files that are NOT loaded or run), including *.TPL, *.FRM and MEMOPAD. (Set at installation.)
1901 -	Dir for files that are loaded or run. Original location of SETTINGS.DFL. (Set at installation.)
1902 -	
1903 -	
1904 -	Dir for COLOR SET files
1905 -	Dir for MACRO files
1906 -	Dir for PRINTER files
1907 -	Dir for DATABASE Condition files
1908 -	Dir for IBM2TIF.EXE
1909 -	
1910 -	Dir for TEMPLATE files
1911 -	Dir for FORMS files
1912 -	Dir and name of MEMOPAD
1913 -	
1914 -	
1915 -	Dir for PRINTER FONT bitmap files C:\XY4\PRNFONTS\*.*
1916 -	Dir for SOFT FONT LIST files C:\XY4\PRNFONTS\ *.SFL
1917 -	
1918 -	
1919 -	
1920 -	(reserved)
1921 -	List of filters for saving text files
1922 -	List of filters for spreadsheets
1923 -	List of filters for database
1924 -	List of filters for graphics
1925 -	List of filters for opening text files
1926 -	
1927 -	
1928 -	
1929 -	
1930 -	

1931 - Marker no. used for TOC if diff. from "1" (Can be 1-9)

1932 - Marker no. used for Index if diff. from "2" (Can be 1-9)

1933 - Marker used for Footnote set if diff. from "1" (Can be 1-3)

1934 -

1935 -

1936 -

1937 -

1938 -

1939 -

1940 - User's choice of "Recorded keystrokes" or "Stored file"

1941 - Cursor position (CP) of MEMOPAD file

1942 -

1943 - Cursor position (CP) of PERS-DIR file

1944 -

1945 - Currency format for spreadsheet import (LEADING/TRAILING)

1946 - Currency format for spreadsheet import (PERIOD/COMMA)

1947 -

1948 - Counter (1-9) for multiple-save backup (not currently used)

1949 - Number of backups user has requested (not currently used)

1950 - Prefer Command Line (cm) or Dialog Box (null), used for Search, Replace and Use Style

1951 -

1952 -

1953 -

1954 -

1955 -

1956 -

1957 -

1958 -

1959 -

1960 -

1961 -

1962 - Copy to screen (s) or clipboard (c) (not currently used)

1963 -

1964 - Flag for dir listing in Open: "OpenComments"/"OpenNoComments"

1965 - Display subdir filenames "OpenSubdir"/"OpenNoSubdir" (not used)

1966 -

1967 - Name of default template in New

1968 -

1969 -

1970 -

1971 - Custom date format

Reserved XyWrite Elements (cont'd)

1972 - Check box for saving changes to Macro file. (1=STSGT, 0=No save)

1973 - Insert TOC and Index and text (1) or marker (0)

1974 - Check box in New for "Include Default Style"

1975 - Directory command and switches for Manage Files (DIR/FI/LO)

1976 -

1977 - Check box for "Offset for facing pages" in Page Setup

1978 - Check box for "Display document info" in Manage Files

1979 -

1980 - Top margin and page length for portrait sizes : letter, legal, executive, A4 and A5. Values are separated by vertical bars (¦).

1981 - Page width, form depth, top margin and page length for portrait custom. Values are separated by vertical bars (¦)

1982 - Top margin and page length for landscape sizes : letter, legal, executive, A4 and A5. Values are separated by vertical bars (¦).

1983 - Page width, form depth, top margin and page length for landscape custom. Values are separated by vertical bars (¦)

1984 - Check box for "Include page number" in "NewH/F".

1985 - Value for "Blank lines following header" in "NewH/F"

1986 - Check box for "Switch to new file" in Save As.

1987 - Sort criteria for DSORT in Manage Files.

1988 -

1989 -

1990 - Envelopes: Top and Left margins for Non-Laser printers

1991 - Envelopes: Top and Left margins for Laser-Manual feed, from center of tray

1992 - Envelopes: Top and Left margins for Laser - Manual feed, from edge of tray

1993 - Envelopes: Top and Left margins for Laser - Tray/Feeder

1994 -

1995 - (reserved)

1996 - (reserved)

1997 - (reserved)

1998 - (reserved)

1999 - Filename for alternate menus

Appendix E XyWrite Character Set

PURPOSE

XyWrite has a unique character set that is designed to support a wide range of languages. The following pages show all 909 XyWrite characters.

ACTION

Typing Special Characters
To type a XyWrite character:

1. Press and hold: Ctrl Alt
2. Type the character number using the numeric keypad. For example, to enter character 174:

   Press: 174

ACTION

Displaying Special Characters
Not all characters from the character set can be displayed on the screen. If a character is not displayable, its character number appears in the text inside square brackets.

In formatted view, the characters that are displayed depend on the code page that you are using (the default is 437). In graphic view, characters that are part of the Bitstream character set are displayed. In the table on the following pages, the letter N indicates characters that are not supported by the Bitstream character set. Other characters are not available in the Bitstream set, but are represented by similar characters that are available. The letter P in the table indicates that such a substitution has been made.

ACTION

Printing Special Characters
The ability to print special characters is printer-dependent. Some printers have an image mode, which means that anything that can be displayed in graphic view can be printed. Other printers offer a wide range of character sets and can include characters that are not part of the character set.

To determine if your printer can produce a special character, print the sample file CHARSET provided with XyWrite. Be sure to enter the name of the typeface (and symbol set if you are using a Hewlett-Packard laser printer) you want to test for character availability.

NOTE

Using the Menus. The menus allow you to enter a special character into a file without identifying it by number. Choose Other from the Insert menu, then choose Special Char.

XyWrite Character Set

(cont'd)

Character Code #	Character Name	Character	Bitstream Supported
01	Smiling face	☺	
02	Smiling face, reverse image	☻	
03	Heart suit symbol	♥	
04	Diamond suit symbol	♦	
05	Club suit symbol	♣	
06	Spade suit symbol	♠	
07	Bullet	•	
08	Bullet reverse	◘	
09	Open circle	○	
10	Open circle reverse	◙	
11	Male symbol	♂	
12	Female symbol	♀	
13	Eighth note	♪	
14	Two sixteenth notes	♫	
15	Sun symbol	☼	
16	Right arrowhead	►	
17	Left arrowhead	◄	
18	Up down arrow	↕	
19	Double exclamation point	‼	
20	Paragraph symbol	¶	
21	Section symbol	§	
22	Solid horizontal rectangle	▬	
23	Up down arrow perpendicular	↨	
24	Up arrow	↑	
25	Down arrow	↓	
26	Right arrow	→	
27	Left arrow	←	

Character Code #	Character Name	Character	Bitstream Supported
28	Right angle symbol	∟	
29	Left-right arrow	↔	
30	Solid triangle	▲	
31	Solid triangle, inverted	▼	
32	Space		
33	Exclamation point	!	
34	Quotation marks	"	
35	Number sign	#	
36	Dollar sign	$	
37	Percent sign	%	
38	Ampersand	&	
39	Apostrophe	'	
40	Left parenthesis	(	
41	Right parenthesis	)	
42	Asterisk	*	
43	Plus sign	+	
44	Comma	,	
45	Hyphen/minus sign	-	
46	Period/full stop	.	
47	Slash	/	
48	Zero	0	
49	One	1	
50	Two	2	
51	Three	3	
52	Four	4	
53	Five	5	
54	Six	6	

XyWrite Character Set

(cont'd)

Character Code #	Character Name	Character	Bitstream Supported
55	Seven	7	
56	Eight	8	
57	Nine	9	
58	Colon	:	
59	Semicolon	;	
60	Less than	<	
61	Equal sign	=	
62	Greater than	>	
63	Question mark	?	
64	At sign	@	
65	A	A	
66	B	B	
57	C	C	
68	D	D	
69	E	E	
70	F	F	
71	G	G	
72	H	H	
73	I	I	
74	J	J	
75	K	K	
76	L	L	
77	M	M	
78	N	N	
79	O (Cyrillic)	O	
80	P	P	
81	Q	Q	

Character Code #	Character Name	Character	Bitstream Supported
82	R	R	
83	S	S	
84	T	T	
85	U	U	
86	V	V	
87	W	W	
88	X	X	
89	Y	Y	
90	Z	Z	
91	Left bracket	[	
92	Back slash	\	
93	Right bracket	]	
94	Circumflex accent	^	
95	Underline/continuous underscore	_	
96	Grave accent	`	
97	a	a	
98	b	b	
99	c	c	
100	d	d	
101	e	e	
102	f	f	
103	g	g	
104	h	h	
105	i	i	
106	j	j	
107	k	k	
108	l	l	

XyWrite Character Set (cont'd)

Character Code #	Character Name	Character	Bitstream Supported
109	m	m	
110	n	n	
111	o (Cyrillic)	o	
112	p	p	
113	q	q	
114	r	r	
115	s	s	
116	t	t	
117	u	u	
118	v	v	
119	w	w	
120	x	x	
121	y	y	
122	z	z	
123	Left brace	{	
124	Vertical line/logical OR	|	
125	Right brace	}	
126	Tilde accent	~	
127	Small house	⌂	
128	C Cedilla	Ç	
129	u umlaut	ü	
130	e acute (Greek)	é	
131	a circumflex	â	
132	a umlaut	ä	
133	a grave	à	
134	a overcircle	å	
135	c cedilla	ç	
136	e circumflex	ê	

Character Code #	Character Name	Character	Bitstream Supported
137	e umlaut	ë	
138	e grave	è	
139	i umlaut	ï	
140	i circumflex	î	
141	i grave	ì	
142	A umlaut	Ä	
143	A overcircle	Å	
144	E acute (Greek)	É	
145	ae diphthong	æ	
146	AE diphthong	Æ	
147	o circumflex	ô	
148	o umlaut	ö	
149	o grave	ò	
150	u circumflex	û	
151	u grave	ù	
152	y umlaut	ÿ	
153	O umlaut	Ö	
154	U umlaut	Ü	
155	Cent sign	¢	
156	Pound sterling sign	£	
157	Yen sign	¥	
158	Peseta sign	₧	
159	Florin sign	ƒ	
160	a acute small (Greek)	á	
161	i acute	í	

XyWrite Character Set (cont'd)

Character Code #	Character Name	Character	Bitstream Supported
162	o acute	ó	
163	u acute	ú	
164	n tilde	ñ	
165	N tilde	Ñ	
166	Ordinal indicator/feminine	ª	
167	Ordinal indicator/masculine	º	
168	Question mark, inverted	¿	
169	Start of line symbol	⌐	
170	Logical NOT/end of line symbol	¬	
171	One half	½	
172	One quarter	¼	
73	Exclamation point, inverted	¡	
74	Left angle quotes	«	
175	Right angle quotes	»	
176	Fill character, light	░	
177	Fill character, medium	▒	
178	Fill character, heavy	▓	
179	Center box bar vertical	│	
180	Right middle box side	┤	
181	Right box side double to single	╡	
182	Right box side single to double	╢	
183	Upper right box corner single to double	╖	
184	Upper right box corner double to single	╕	
185	Right box side double	╣	
186	Center box bar double	║	
187	Upper right box corner double	╗	
188	Lower right box corner double	╝	

Character Code #	Character Name	Character	Bitstream Supported
189	Lower right box corner single to double	╜	
190	Lower right box corner double to single	╛	
191	Upper right box corner	┐	
192	Lower left box corner	└	
193	Middle box bottom	┴	
194	Middle box top	┬	
195	Left middle box side	├	
196	Center box bar horizontal	─	
197	Box intersection	┼	
198	Left box side single to double	╞	
199	Left box side double to single	╟	
200	Lower left box corner double	╚	
201	Upper left box corner double	╔	
202	Middle box bottom double	╩	
203	Middle box top double	╦	
204	Left box side double	╠	
205	Center box bar horizontal double	═	
206	Box intersection double	╬	
207	Middle box bottom single to double	╧	
208	Middle box bottom double to single	╨	
209	Middle box top double to single	╤	
210	Middle box top single to double	╥	
211	Lower left box corner double to single	╙	
212	Lower left box corner single to double	╘	
213	Upper left box corner single to double	╒	
214	Upper left box corner double to single	╓	
215	Box intersection single to double	╫	
216	Box intersection double to single	╪	

Character Code #	Character Name	Character	Bitstream Supported
217	Lower right box corner	┘	
218	Upper left box corner	┌	
219	Solid fill character	█	
220	Solid fill character, bottom half	▄	
221	Solid fill character, left half	▌	
222	Solid fill character, right half	▐	
223	Solid fill character, top half	▀	
224	Alpha small (Greek)	α	
225	Sharp s (small)	ß	
226	Gamma capital (Greek)	Γ	
227	Pi small (Greek)	π	
228	Sigma capital (Greek)	Σ	
229	Sigma small (Greek)	σ	
30	Mu small (Greek)	μ	
231	Tau small (Greek)	τ	
232	Phi capital (Greek)	Φ	
233	Theta capital (Greek)	Θ	
234	Omega capital (Greek)	Ω	
235	Delta small (Greek)	δ	
236	Infinity symbol	∞	
237	Phi small (Greek)	φ	
238	Epsilon small (Greek)	ε	
239	Intersection symbol	∩	
240	Identity symbol, almost equal	≡	
241	Plus or minus sign	±	
242	Greater than or equal to	≥	
243	Less than or equal to	≤	
244	Upper integral symbol section	⌠	

Character Code #	Character Name	Character	Bitstream Supported
245	Lower integral symbol section	⌡	
246	Divide sign	÷	
247	Nearly equals symbol	≈	
248	Degree symbol	°	
249	Product dot	·	
250	Middle dot	•	
251	Radical symbol	√	
252	n superscript	^{n}	
253	Two superscript	^{2}	
254	Solid square/histogram square bullet	■	
255	Em space		
256	Superior copyright mark	©	
257	Superior registered trademark	®	
258	Ellipsis	…	
259	En dash	–	
260	Em dash	—	
261	En space		
262	Left single angle quote	‹	
263	Right single angle quote	›	
264	Left double quote	“	
265	Right double quote	”	
266	Left single quote	‘	
267	Right single quote	’	
268	Right parenthesis, small	)	
269	Left parenthesis, small	(	
270	Franc sign	Fr	
271	Numeric space		

XyWrite Character Set

(cont'd)

Character Code #	Character Name	Character	Bitstream Supported
272	Peseta sign	₧	
273	Middle box bottom single to double	╧	
274	Middle box top double to single	╤	
275	Upper right box corner single to double	╖	
276	Upper right box corner double to single	╕	
277	Lower right box corner double to single	╛	
278	Lower left box corner double to single	╙	
279	Lower left box corner single to double	╘	
280	Upper left box corner double to single	╓	
281	Upper left box corner single to double	╒	
'82	Reserved		
33	Right box side double to single	╡	
.84	Middle box bottom double to single	╨	
285	Middle box top single to double	╥	
286	Box intersection double to single	╪	
287	Box intersection single to double	╫	
288	Left box side double to single	╟	
289	Right box side single to double	╢	
290	Lower right box corner single to double	╜	
291	Solid fill character, left half	▌	
292	Left box side single to double	╞	
293	Solid fill character, right half	▐	
294	Alpha small (Greek)	α	P
295	Alpha capital (Greek)	A	P
296	Identity symbol, almost equal	≡	
297	Gamma small (Greek)	γ	N

Character Code #	Character Name	Character	Bitstream Supported
298	Gamma capital (Greek)	Γ	P
299	Greater than or equal to	≥	
300	Pi small (Greek)	π	P
301	Pi capital (Greek)	Π	N
302	Less than or equal to	≤	
303	Sigma small (Greek)	σ	P
304	Sigma capital (Greek)	Σ	P
305	Upper integral symbol section	⌠	
306	Mu small (Greek)	μ	P
307	Mu capital (Greek)	M	P
308	Nu small (Greek)	ν	N
309	Nu capital (Greek)	N	P
310	Lower integral symbol section	⌡	
311	Tau small (Greek)	τ	P
312	Tau capital (Greek)	T	P
313	Nearly equals symbol	≈	
314	Phi capital (Greek)	Φ	P
315	Theta capital (Greek)	Θ	P
316	Product dot	·	
317	Delta small (Greek)	δ	P
318	Radical symbol	√	
319	Infinity symbol	∞	
320	Phi small (Greek)	φ	N
321	Epsilon small (Greek)	ε	P
322	Epsilon capital (Greek)	E	P

XyWrite Character Set

(cont'd)

Character Code #	Character Name	Character	Bitstream Supported
323	Intersection symbol	∩	
324	d stroke	đ	
325	L acute	Ĺ	
326	Double acute accent	˝	
327	I acute	Í	
328	N acute	Ń	
329	Caron accent	ˇ	
330	Overdot accent	˙	
331	Ogonek accent	˛	
332	a ogonek	ą	
333	A ogonek	Ą	
334	n acute	ń	
335	Breve accent	˘	
336	u overcircle	ů	
337	L caron	Ľ	P
338	N caron	Ň	
339	n caron	ň	
340	c acute	ć	
341	I caron	ľ	P
342	Z caron	Ž	
343	A breve	Ă	
344	S caron	Š	
345	S acute	Ś	
346	z caron	ž	
347	E caron	Ě	
348	a breve	ă	
349	s caron	š	

Character Code #	Character Name	Character	Bitstream Supported
350	I stroke	ł	
351	s acute	ś	
352	E ogonek	Ę	
353	S cedilla	Ş	
354	e caron	ě	
355	e ogonek	ę	
356	O double acute	Ő	
357	o double acute	ő	
358	z acute	ź	
359	U double acute	Ű	
360	u double acute	ű	
361	C caron	Č	
362	N macron	Ň	
363	Z acute	Ź	
364	L stroke	Ł	
365	s cedilla	ş	
366	Z overdot	Ż	
367	T cedilla	Ţ	
368	r caron	ř	
369	R caron	Ř	
370	z overdot	ż	
371	U overcircle	Ů	
372	t cedilla	ţ	
373	C acute	Ć	
374	c caron	č	
375	g small (Cyrillic)	г	N
376	Eta small (Greek)	η	P

Character Code #	Character Name	Character	Bitstream Supported
377	Eta capital (Greek)	H	N
378	Theta small (Greek)	θ	P
379	Phi small (Gr.)	ϕ	P
380	Omicron small (Greek)	ο	P
381	Omicron capital (Greek)	O	P
382	Omega capital (Greek)	Ω	P
383	Upsilon umlaut capital (Greek)	Ÿ	P
384	Delta capital (Greek)	Δ	P
385	Beta small (Greek)	β	P
386	Beta capital (Greek)	B	P
387	One-third	⅓	
388	Two-thirds	⅔	
389	Fraction bar	-	
390	Hyphen dash	-	
391	Dagger	†	
392	Double dagger	‡	
393	Radical, tape mark	√	
394	Solid diamond	♦	
395	Trademark symbol	™	
396	Open square	□	
397	Lozenge	⌑	
398	One-fourth	¼	
399	One-half	½	
400	Three-fourths	¾	
401	n superscript	^{n}	
402	Solid square, small	■	
403	Three-eighths	⅜	
404	One-eighth	⅛	

Character Code #	Character Name	Character	Bitstream Supported
405	Five-eighths	⅝	
406	Seven-eighths	⅞	
407	D stroke, capital	Đ	
408	o (Cyrillic)	о	P
409	O (Cyrillic)	О	P
410	U breve (Cyrillic)	Ў	P
411	G capital (Cyrillic)	Г	P
412	Omicron acute small (Greek)	ό	P
413	Omicron acute capital (Greek)	Ό	N
414	Start of line symbol	⌐	
415	D caron capital	Ď	N
416	D caron small	ď	N
417	R acute capital	Ŕ	N
418	r acute small	ŕ	N
419	T caron capital	Ť	N
420	t caron small	ť	N
421	d special small (Cyrillic)	ђ	N
422	kh small (Cyrillic)	х	P
423	KH capital (Cyrillic)	Х	P
424	lj small (Cyrillic)	љ	N
425	LJ capital (Cyrillic)	Љ	N
426	l small (Cyrillic)	л	N
427	L capital (Cyrillic)	Л	N
428	m small (Cyrillic)	м	N
429	M capital (Cyrillic)	М	P
430	YA capital (Cyrillic)	Я	N
431	D special capital (Cyrillic)	Ђ	N

Character Code #	Character Name	Character	Bitstream Supported
432	y small (Cyrillic)	ы	N
433	g special small (Cyrillic)	ґ	N
434	nj small (Cyrillic)	њ	N
435	a small (Cyrillic)	а	P
436	b small (Cyrillic)	б	N
437	e small (Cyrillic)	е	P
438	n small (Cyrillic)	н	N
439	N capital (Cyrillic)	Н	P
440	r small (Cyrillic)	р	P
441	R capital (Cyrillic)	Р	P
442	s small (Cyrillic)	с	P
443	S capital (Cyrillic)	С	P
444	Y capital (Cyrillic)	Ы	N
445	G special capital (Cyrillic)	Ґ	N
446	NJ capital (Cyrillic)	Њ	N
447	A capital (Cyrillic)	А	P
448	B capital (Cyrillic)	Б	N
449	E capital (Cyrillic)	Е	P
450	c special capital (Cyrillic)	ћ	N
451	ts small (Cyrillic)	ц	N
452	C special capital (Cyrillic)	Ћ	N
453	TS capital (Cyrillic)	Ц	N
454	t small (Cyrillic)	т	N
455	T capital (Cyrillic)	Т	P
456	u small (Cyrillic)	у	P
457	U capital (Cyrillic)	У	N
458	sh small (Cyrillic)	ш	N

Character Code #	Character Name	Character	Bitstream Supported
459	ye small (Cyrillic)	є	N
460	YE capital (Cyrillic)	Є	N
461	e umlaut small (Cyrillic)	ё	P
462	E umlaut capital (Cyrillic)	Ё	P
463	k special small (Cyrillic)	ќ	N
464	d small (Cyrillic)	д	N
465	k small (Cyrillic)	к	N
466	SH capital (Cyrillic)	Ш	N
467	K special capital (Cyrillic)	Ќ	N
468	D capital (Cyrillic)	Д	N
469	i small (Cyrillic)	и	N
470	I capital (Cyrillic)	И	N
471	i special small (Cyrillic)	і	P
472	I special capital (Cyrillic)	І	P
473	i umlaut small (Cyrillic)	ї	P
474	I umlaut capital (Cyrillic)	Ї	P
475	j small (Cyrillic)	ј	P
476	J capital (Cyrillic)	Ј	P
477	K capital (Cyrillic)	К	N
478	s special small (Cyrillic)	ѕ	P
479	S special capital (Cyrillic)	Ѕ	P
480	e special small (Cyrillic)	э	N
481	E special capital (Cyrillic)	Э	N
482	u breve small (Cyrillic)	ў	P
483	p small (Cyrillic)	п	N
484	zh small (Cyrillic)	ж	N
485	ZH capital (Cyrillic)	Ж	N

Character Code #	Character Name	Character	Bitstream Supported
486	shch small (Cyrillic)	щ	N
487	SHCH capital (Cyrillic)	Щ	N
488	dz special small (Cyrillic)	џ	N
489	DZ special capital (Cyrillic)	Џ	N
490	f small (Cyrillic)	ф	N
491	F capital (Cyrillic)	Ф	P
492	v small (Cyrillic)	в	N
493	V capital (Cyrillic)	В	P
494	z small (Cyrillic)	з	N
495	Z capital (Cyrillic)	З	N
496	ch small (Cyrillic)	ч	N
497	yu small (Cyrillic)	ю	N
498	CH capital (Cyrillic)	Ч	N
499	YU capital (Cyrillic)	Ю	N
500	i special small (Cyrillic)	й	N
501	J special capital (Cyrillic)	Й	N
502	P capital (Cyrillic)	П	N
503	Soft sign small (Cyrillic)	ь	N
504	Hard sign small (Cyrillic)	ъ	N
505	ya small (Cyrillic)	я	N
506	Soft sign capital (Cyrillic)	Ь	N
507	Hard sign capital (Cyrillic)	Ъ	N
508	Numero sign (Cyrillic)	№	N
509	Iota acute capital (Greek)	Ί	N
510	Iota umlaut small (Greek)	ϊ	P
511	Iota umlaut capital (Greek)	Ϊ	P
512	Zeta small (Greek)	ζ	N
513	Zeta capital (Greek)	Ζ	P

Character Code #	Character Name	Character	Bitstream Supported
514	Iota acute and umlaut small (Greek)	ΐ	N
515	Upsilon small (Greek)	υ	N
516	Upsilon capital (Greek)	Υ	P
517	Upsilon acute small (Greek)	ύ	N
518	Iota small (Greek)	ι	P
519	Iota capital (Greek)	Ι	P
520	Psi capital (Greek)	Ψ	N
521	Kappa small (Greek)	κ	N
522	Kappa capital (Greek)	Κ	N
523	Chi small (Greek)	χ	N
524	Chi capital (Greek)	Χ	P
525	Upsilon acute capital (Greek)	Ύ	N
526	Lambda small (Greek)	λ	N
527	Alpha acute capital (Greek)	Ά	N
528	Lambda capital (Greek)	Λ	N
529	Psi small (Greek)	ψ	N
530	Acute and umlaut accent	΅	N
531	Omega acute capital (Greek)	Ώ	N
532	Xi small (Greek)	ξ	N
533	i umlaut small (Greek)	ϊ	P
534	I overdot capital (Greek)	İ	
535	Omega small (Greek)	ω	N
536	Alpha acute small (Greek)	ά	N
537	Rho small (Greek)	ρ	N
538	Reserved		N
539	Upsilon umlaut small (Greek)	ϋ	N
540	Epsilon acute small (Greek)	έ	N
541	Epsilon acute capital (Greek)	Έ	N

Character Code #	Character Name	Character	Bitstream Supported
542	Xi capital (Greek)	Ξ	N
543	Omega acute small (Greek)	ώ	N
544	Eta acute small (Greek)	ή	N
545	Eta acute capital (Greek)	Ή	N
546	Iota acute small (Greek)	ί	P
547	g breve small	ğ	N
548	G breve capital	Ğ	N
549	Long dash/throughscore	—	
550	Sigma small-final form (Greek)	ς	N
551	Minutes symbol	′	N
552	Seconds symbol	″	N
553	Upsilon acute and umlaut small (Greek)	ΰ	N
554	Integral symbol	∫	N
555	Five subscript	$_5$	
556	Zero subscript	$_0$	
557	Zero superscript	^{0}	
558	Four superscript	^{4}	
559	Five superscript	^{5}	
560	Six superscript	^{6}	
561	Six subscript	$_6$	
562	Seven superscript	^{7}	
563	Seven subscript	$_7$	
564	Eight superscript	^{8}	
565	Eight subscript	$_8$	
566	Nine superscript	^{9}	
567	Nine subscript	$_9$	
568	Not equal sign	≠	
569	Not assigned		

Character Code #	Character Name	Character	Bitstream Supported
570	Union symbol/Logical sum symbol	∪	N
571	Is not element of symbol	ε	N
572	Upper left parenthesis	⎛	N
573	Upper right parenthesis	⎞	N
574	Lower left parenthesis	⎝	N
575	Upper summation symbol	⎲	N
576	Parallel symbol	‖	N
577	Permille symbol	‰	
578	Substitute blank		N
579	Slash superscript	⁄	N
580	Angle symbol	∠	N
581	Bottle symbol	●	N
582	Lower right parenthesis	⎠	N
583	Not assigned		
584	Approximately equal to	≈	N
585	Delta superscript	▵	N
586	Del	∆	N
587	Congruent to symbol	≅	N
588	Right arrow superscript	→	N
589	Is included in symbol	⊂	N
590	Left angle bracket superscript	>	N
591	Lower summation symbol	⎳	N
592	Includes symbol	⊃	N
593	Right angle bracket superscript	<	N
594	Solid triangle rotated	►	
595	Circle plus, closed sum	⊕	N
596	Lower right, upper left brace	⎰	N
597	Upper left, lower right brace	⎱	N
598	Prescription symbol	℞	N

XyWrite Character Set

(cont'd)

Character Code #	Character Name	Character	Bitstream Supported
599	Perpendicular symbol	⊥	N
600	Zero slash	Ø	N
601	Increase	/	N
602	Therefore symbol	∴	N
603	Circle x, closed product	⊗	N
604	Decrease	\	N
605	Total symbol	♀	N
606	And symbol	∧	N
607	Or symbol	∨	N
608	Double overline	‗	N
609	Partial differential symbol	∂	N
610	Sine curve	∿	N
611	Liter symbol	ℓ	
612	Not assigned		
613	Theta small open form	ϑ	N
614	Square box with slash	⍁	N
615	Equal sign superscript	$^=$	N
616	Minus sign superscript	$^-$	N
617	Not assigned		
618	Right angle bracket	>	N
619	Left angle bracket	<	N
620	One subscript	$_1$	
621	Two subscript	$_2$	
622	Three subscript	$_3$	
623	Four subscript	$_4$	
624	Infinity symbol superscript	$^\infty$	N
625	Pi superscript	$^\pi$	N
626	Solid vertical rectangle	▮	

Character Code #	Character Name	Character	Bitstream Supported
627	Macron accent	¯	N
628	Fraction slash	⁄	N
629	Contour integral	∮	
630	Proportional to	∝	
631	Square root extension	—	
632	Minus plus	∓	
633	Approximately	~	
634	Right angle	∟	
635	Large bullet	●	
636	Integral sign (midsect)	⎮	
637	Epsilon, alternate small	ε	
638	Solid square, large	■	
639	Superscript a	^{a}	
640	Superscript e	^{e}	
641	Superscript i	^{i}	
642	Superscript l	^{l}	
643	Superscript m	^{m}	
644	Superscript o	^{o}	
645	Superscript r	^{r}	
646	Superscript s	^{s}	
647	Superscript t	^{t}	
648	Backslash (ASCII)	\	
649	Superscript circle	$^{\circ}$	
650	Superscript dot	$^{\cdot}$	
651	Exists in symbol	∈	
652	Superscript minus sign	$^{-}$	
653	Superscript plus sign	$^{+}$	
654	Foot symbol	′	

Character Code #	Character Name	Character	Bitstream Supported
655	Inch symbol	″	
656	Slash (ASCII)	/	
657	fi	ﬁ	
658	fl	ﬂ	
659	OE ligature capital	Œ	
660	OE ligature small	œ	
661	ij	ĳ	
662	Ij	Ĳ	
663	ffl	ﬄ	N
664	ffi	ﬃ	N
65	ff	ﬀ	
6	Open unhappy face	☹	
667	d caron small, alternate	ď	
668	d macron small	đ	
669	macron capital	Đ	
670	e macron small	ē	
671	E macron capital	Ē	
672	g cedilla small	ģ	
673	G cedilla capital	Ģ	
674	i macron small	ī	
675	I macron capital	Ī	
676	i overdot capital	İ	
677	i ogonek small	į	
678	I ogonek capital	Į	
679	k cedilla small	ķ	
680	K cedilla capital	Ķ	
681	l macron small	l̄	
682	L macron capital	L̄	

Character Code #	Character Name	Character	Bitstream Supported
683	l cedilla small	ļ	
684	L cedilla capital	Ļ	
685	n cedilla small	ņ	
686	N cedilla capital	Ņ	
687	r cedilla small	ŗ	
688	R cedilla capital	Ŗ	
689	S macron capital	Š	
690	s cedilla small, alternate	ş	
691	S cedilla capital, alternate	Ş	
692	t macron small	t̄	
693	T macron capital	T̄	
694	u macron small	ū	
695	U macron capital	Ū	
696	w circumflex small	ŵ	
697	W circumflex capital	Ŵ	
698	y circumflex small	ŷ	
699	Y circumflex capital	Ŷ	
700	t caron small	ť	
701	T caron capital	Ť	
702	a macron small	ā	
703	A macron capital	Ā	
704	D caron, alternate	D'	
705	g caron small	ǧ	
706	G caron capital	Ǧ	
707	n macron small	n̄	
708	r grave small	r̀	
709	R grave capital	R̀	

Character Code #	Character Name	Character	Bitstream Supported
710	s macron small	ş	
711	I tilde capital	Ĩ	
712	i tilde small	ĩ	
713	n small high comma	ŉ	
714	u ogonek small	ų	Capital?
715	u tilde lower	ũ	
716	U tilde upper	Ũ	
717	LL mid dot upper	L·L	N
718	ll mid dot lower	l·l	N
719	j dotless	ȷ	
720	L mid dot upper	Ŀ	
721	l mid dot lower	ŀ	
722	acute accent lower	´	
723	acute accent upper	´	
724	grave accent lower	`	
725	grave accent upper	`	
726	circumflex accent lower	ˆ	
727	circumflex accent upper	ˆ	
728	umlaut accent lower	¨	
729	tilde accent lower	˜	
730	tilde accent upper	˜	
731	caron accent lower	ˇ	
732	Macron accent lower	¯	
733	Macron accent upper	¯	
734	Breve accent lower	˘	
735	Overcircle accent lower	˚	
736	Overcircle accent upper	˚	

Character Code #	Character Name	Character	Bitstream Supported
737	Overdot accent lower	˙	
738	Double acute accent lower	˝	
739	Over comma accent	ʼ	
740	Under dash	ˍ	
741	Under dot	.	
742	Under comma accent	,	
743	Sub-comma	,	
744	Sub-double comma	„	
745	Mirrored under comma accent	,	
746	High comma	ʼ	
747	Over comma accent up	ʼ	
748	Apostrophe, up beside	ʼ	
749	Acute accent lower	´	
750	Acute accent upper	´	
751	Grave accent lower	`	
752	Grave accent upper	`	
753	Circumflex accent lower	ˆ	
754	Circumflex accent upper	ˆ	
755	Umlaut accent lower	¨	
756	Umlaut accent upper	¨	
757	Tilde accent lower	˜	
758	Tilde accent upper	˜	
759	Caron accent lower	ˇ	
760	Caron accent upper	ˇ	
761	Cedilla accent	¸	
762	Macron accent lower	¯	
763	Macron accent upper	¯	
764	Breve accent lower	˘	

Character Code #	Character Name	Character	Bitstream Supported
765	Breve accent upper	˘	
766	Overcircle accent lower	·	
767	Overcircle accent upper	·	
768	Dot accent lower	·	
769	Dot accent upper	·	
770	Double acute accent lower	·	
771	Double acute accent upper	·	
772	Over comma accent lower	·	
773	Over comma accent upper	·	
774	Under macron accent	_	
775	Under dot	.	
776	Under comma accent	.	
777	Mirrored comma accent	.	
778	Ogonek accent	.	
779	Apostrophe, lower beside	·	
780	Apostrophe, upper beside	·	
781	Solid circle	●	
782	C Cedilla	Ç	
783	u umlaut	ü	
784	e acute (~~Greek~~)	é	
785	a circumflex	â	
786	a umlaut	ä	
787	a grave	à	
788	a overcircle	å	
789	c cedilla	ç	
790	e circumflex	ê	
791	e umlaut	ë	
792	e grave	è	

Character Code #	Character Name	Character	Bitstream Supported
793	i umlaut	ï	
794	i circumflex	î	
795	i grave	ì	
796	A umlaut	Ä	
797	A overcircle	Å	
798	E acute (Greek)	É	
799	ae diphthong	æ	
800	AE diphthong	Æ	
801	o circumflex	ô	
802	o umlaut	ö	
803	o grave	ò	
804	u circumflex	û	
805	u grave	ù	
806	y umlaut	ÿ	
807	O umlaut	Ö	
808	U umlaut	Ü	
809	o slash	ø	
810	Pound sterling sign	£	
811	O slash	Ø	
812	Multiply sign	×	
813	Florin sign	ƒ	
814	a acute small (Greek)	á	
815	i acute	í	
816	o acute	ó	
817	u acute	ú	
818	n tilde	ñ	
819	N tilde	Ñ	
820	Ordinal indicator/feminine	ª	N

XyWrite Character Set (cont'd)

Character Code #	Character Name	Character	Bitstream Supported
821	Ordinal indicator/masculine	º	N
822	Question mark, inverted	¿	
823	Registered trademark symbol	®	
824	Logical NOT/end of line symbol	¬	
825	One half	½	
826	One quarter	¼	
827	Exclamation point, inverted	¡	
828	Left angle quotes	«	
829	Right angle quotes	»	
830	Fill character, light	░	
831	Fill character, medium	▒	
32	Fill character, heavy	▓	
3	Center box bar vertical	│	
,34	Right middle box side	┤	
835	A acute	Á	
836	A circumflex	Â	
837	A grave	À	
838	Copyright symbol	©	
839	Right box side double	╣	
840	Center box bar double	║	
841	Upper right box corner double	╗	
842	Lower right box corner double	╝	
843	Cent sign	¢	
844	Yen sign	¥	
845	Upper right box corner	┐	
846	Lower left box corner	└	
847	Middle box bottom	┴	
848	Middle box top	┬	

Character Code #	Character Name	Character	Bitstream Supported
849	Left middle box side	├	
850	Center box bar horizontal	─	
851	Box intersection	┼	
852	a tilde	ã	
853	A tilde	Ã	
854	Lower left box corner double	╚	
855	Upper left box corner double	╔	
856	Middle box bottom double	╩	
857	Middle box top double	╦	
858	Left box side double	╠	
859	Center box bar horizontal double	═	
860	Box intersection double	╬	
861	International currency symbol	¤	
862	Eth Icelandic small	ð	
863	Eth Icelandic capital	Ð	
864	E circumflex	Ê	
865	E umlaut	Ë	
866	E grave	È	
867	i dotless	ı	
868	I acute	Í	
869	I circumflex	Î	
870	I umlaut	Ï	
871	Lower right box corner	┘	
872	Upper left box corner	┌	
873	Solid fill character	█	
874	Solid fill character, bottom half	▄	

Character Code #	Character Name	Character	Bitstream Supported
875	Right vertical, bar graphic	¦	
876	I grave	Ì	
877	Solid fill character, top half	▀	
878	O acute	Ó	
879	Sharp s (small)	ß	
880	O circumflex	Ô	
881	O grave	Ò	
882	o tilde	õ	
883	O tilde	Õ	
884	Micro symbol	µ	N
885	Thorn Icelandic small	þ	
86	Thorn Icelandic capital	Þ	
7	U acute	Ú	
88	U circumflex	Û	
889	U grave	Ù	
890	y acute	ý	
891	Y acute	Ý	
892	Overline	‾	
893	Acute accent	´	
894	Syllable hyphen	-	
895	Plus or minus sign	±	
896	Double underscore	‗	
897	Three quarters	¾	
898	Paragraph symbol	¶	
899	Section symbol (USA) paragraph symbol (Eur)	§	
900	Divide sign	÷	
901	Cedilla accent	¸	

Character Code #	Character Name	Character	Bitstream Supported
902	Degree symbol	°	
903	Diaeresis/umlaut accent	¨	
904	Middle dot	·	
905	One superscript	^{1}	
906	Three superscript	^{3}	
907	Two superscript	^{2}	
908	Solid square/histogram square bullet	■	
909	Em space		

NOTES

Index

Notes

Notes

XYQUEST®

A Division of The Technology Group, Inc.
36 South Charles Street
Baltimore, Maryland 21201
(410) 576-2040   Fax (410) 576-1968