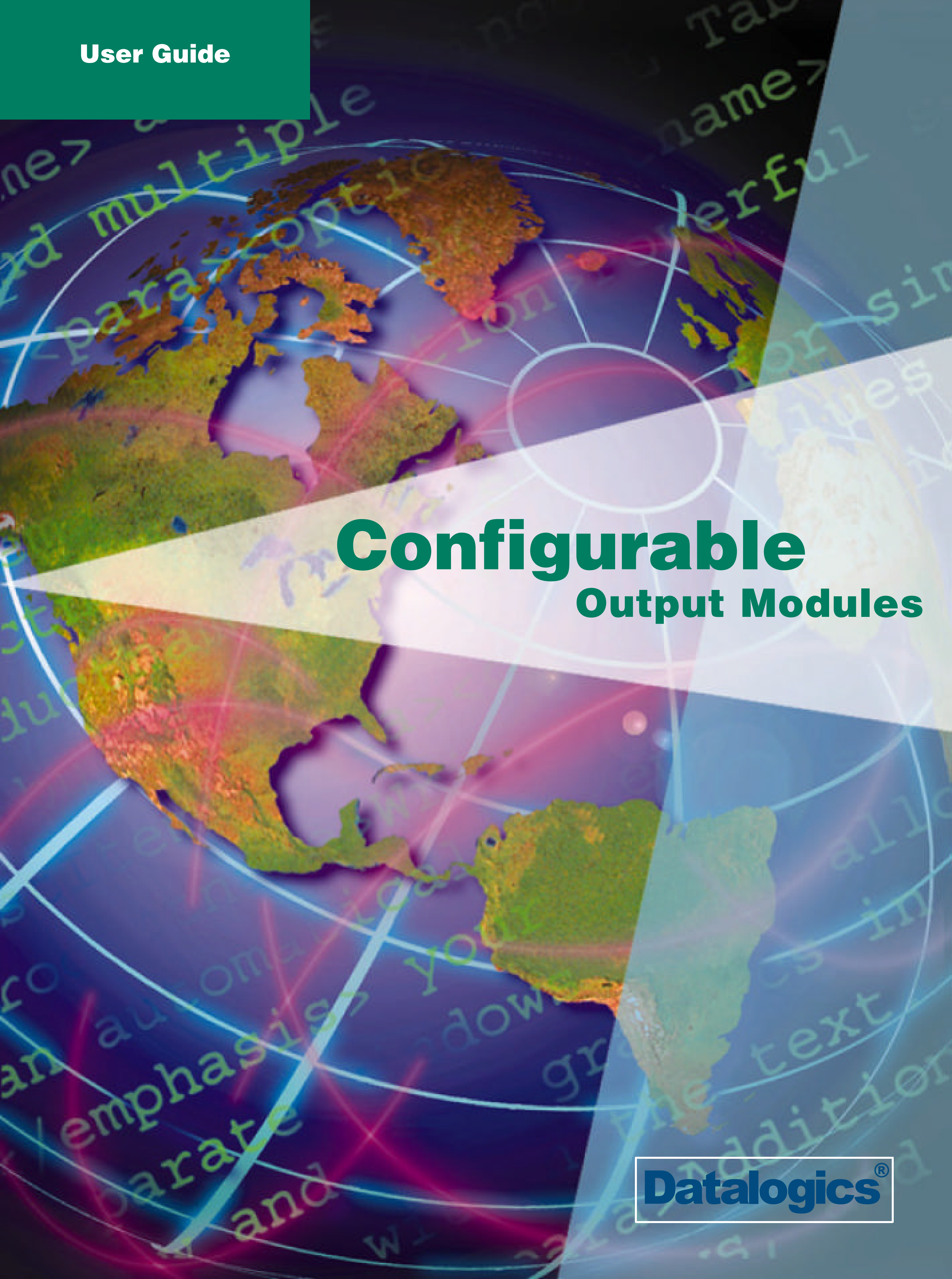
User Guide



Configurable

Output Modules

Datalogics®

Datalogics

DL PAGER CONFIGURABLE OUTPUT MODULES

User Guide

This Reference Guide supports the DL Pager Composition System, version 6.5
Use of Datalogics software is subject to the applicable license agreement.

For additional information, contact:
Datalogics Incorporated
101 North Wacker Drive, Suite 1800
Chicago, Illinois 60606-7301
Phone: 312-853-8200
Fax: 312-853-8282
http://www.datalogics.com

Table of Contents

List of Tables

About *This Book*

This chapter describes the content of this book, and lists sources for additional information.

What You Should Know

Before working with *Configurable Output Modules* for the *DL Pager* Composition System, you should be familiar with the basic concepts of composition and typesetting within the *DL Pager* environment. This batch composition system uses a number of different files to produce text and graphics for output in electronic or hard copy typeset formats. Knowledge of the output format your document requires is also essential.

How This Book is Organized

Chapter 1: ''About This Book'' describes the content of this book.

Chapter 2: ''Overview of Configurable Output Modules'' provides an overview of the *Configurable Output Modules* and highlights some important things to consider when configuring output modules.

Chapter 3: ''Installation Procedures'' explains the software installation procedure, lists the requirements and describes the general use of the product.

Chapter 4: ''Output Module Parameters'' defines the relevant terms and describes the parameters available for each of the configurable output modules.

Document Conventions

These notation conventions appear throughout this book.

- Product and program names are set in *Initial Cap Italic* type.
- Directory names and filenames are set in `monospace`.
- Menu and screen options or fields are set in **bold**.
- A command syntax in text appears in `Courier Monospace`. Optional parameters appear in braces ({ }).
- The names of all keyboard keys appear in a lightface roman. A plus sign (+) between keys indicates that the keys should be pressed at the same time. A slash (/) between keys indicates that the first key—the key to

the left of the slash—should be pressed and released before pressing the second key.

- Examples are separated from the text and set in `monospace`
- In text, tags appear in angle brackets and are set in `monospace`.
- An OR operator (|) signifies one command qualifier or the other.
- Page numbers in this guide do not correspond to page numbers in the PDF file. The numbering schema (*e.g.*, *4.1* or *A.10*) represents the chapter number *(4)* or appendix letter *(A)* first, followed by the page number (*1* or *10*), and these are separated by a period.

Related Documentation

The following table lists sources for further information on using the *DL Pager* composition system. All of these books are available from your Datalogics support representative.

The first table lists all the documentation titles and their corresponding filenames. Additional tables include details about what type of information is in each book.

Table 1-1 Complete *DL Pager* Documentation Set

Title	Filename
DL Pager Charting System	dlchart.pdf
DL Pager Language Reference Guide	dlpager.pdf
DL Pager NT Migration	migration.pdf
DL Publisher Suite	dlpublisher.pdf
Edgar OM	edgarom.pdf
Fonter, Styler, Dictor	fsd.pdf
Integrated Graphics System	igs.pdf
Input/Output (User-Oriented)	userio.pdf
Math Facilities	math.pdf

Table 1-1 Complete *DL Pager* Documentation Set (Cont.)

Title	Filename
Page Index Files	pif.pdf
Page Squeeze	squeeze.pdf
PDF Output Module	pdfom.pdf
PostScript Output Module	psom.pdf
PPrint	pprint.pdf
REPROC/Loose-leaf, Page, and PSPLIT	reproc.pdf
Reservation Layout	reslay.pdf
Rules (Vertical and Horizontal)	rules.pdf
SEC HTML Export	sechtml.pdf
The Spell Program	spell.pdf
Tables (DL Pager Tables)	tables.pdf
Xerox Output Module	xeroxom.pdf
XML and SGML Related Features	sgmlxml.pdf

Table 1-2 Composition and Commands

Title and Overview	Filename
Composition System Overview Job Files, Compose Command, Pager Keyboard Commands (PKC), and Interpret composition log files	composition.pdf
Concepts and Facilities Overview of DL Pager composition engine, ancillary composition components and graphics facilities	concepts.pdf
DL Pager Language Reference Guide DL Pager Markup and Formatting Commands	dlpager.pdf
Fonter, Styler, Dictor User guides for Fonter, Styler, and Dictor programs	fsd.pdf

Table 1-3 *DL Pager* Suite

Title and Overview	Filename
DL Pager Charting System Create and compose charts to certain graphic-enabled output modules	dlchart.pdf
DL Pager NT Migration Migrate composition jobs designed for the VMS version of DL Pager to DL Pager NT.	migration.pdf
DL Publisher Suite A full featured batch composition System for Windows NT	dlpublisher.pdf
Integrated Graphics System Manage and integrate graphics within the DL Pager composition system	igs.pdf
SEC HTML Export Export an HTML for SEC representation of a DL Pager composition job	sechtml.pdf

Table 1-4 Output Modules

Title and Overview	Filename
AFP Output Module Advanced Function Printer (AFP) output file format	afpom.pdf
Configurable Output Modules User-configurable output modules which read parameters specified with a configuration file at run-time	configom.pdf
Edgar OM Electronic Data Gathering Analysis and Retrieval (EDGAR) output file format	edgarom.pdf
PDF Output Module Portable Document Format (PDF) output file format	pdfom.pdf
PostScript Output Module PostScript output file format	psom.pdf
Xerox Output Module Compose to Xerox 9700-series printers and 4800-series color printers	xeroxom.pdff

Table 1-5 Applications and Technical References

Title and Overview	Filename
Input/Output (User-Oriented) Create indexes, tables of contents and other listings	userio.pdf
Math facilities Typeset mathematical characters and position them into formulas	math.pdf
Page Index Files Per page output information for custom page manipulation	pif.pdf

Overview *of Configurable Output Modules*

This chapter provides an introduction to configurable output modules, explaining terms such as Class File and Startup File and also highlights some considerations which should be taken into account.

Background

The *DL Pager* output modules (OM) have historically been configured by Datalogics Support Representatives to suit the needs of individual customers. Released as a separate executable to be used with *DL Pager* on VMS, or built directly into the *DL Pager* executable on UNIX and Windows NT, the OMs were defined with a fixed configuration; modification to OM configuration values meant rebuilding executables.

To increase user flexibility and reduce delivery time of the *DL Pager* system, user-configurable output modules have been created. The user-configurable output modules read parameters specified within a Class File at run-time. A Class File can contain more than one *class* or output module. Currently, the PostScript, AFP, Xerox, and Edgar output modules are configurable, the PDF output module does not require configuration, and the output module for Copyfit is not configurable.

Considerations

- On VMS systems, existing pre-configured output modules may be used with new versions of *DL Pager*, although this is not recommended.
- Datalogics Support Representatives will generate OM Class Files to replace the existing pre-configured output modules.
- Implementing the new OM Class Files with older *DL Pager* versions is possible but may produce unexpected results.
- If a Class File is already in use, new configurations can be merged into the existing file.
- If no Class Files are specified for an OM, default configuration values will be used. These values are listed in “Understanding Startup Files and Class Files”.

Installation *Procedures*

This chapter contains the installation and setup information needed to work with configurable output modules.

Introduction

DL Pager versions 5.047 and higher include configurable output modules for PostScript, AFP, Xerox, and Edgar output. The OMs are set to the default values listed in "Understanding Startup Files and Class Files". On NT, a Class File with the default settings is included in the installation of *DL Pager* and can be modified to meet specific requirements. If you are using a modified Class File, the file should be saved with a new file name to prevent its loss in the case of a new installation of *DL Pager*. Because performing a new installation on an NT machine where *DL Pager* exists requires uninstalling the previous version, the modified Class File will be removed and replaced with one containing the default settings. If the modified class file with the new file name is to be used, a reference to this file will need to be added to the *DL Pager* Startup File. On VMS, the class (or configuration) files are delivered separate from the installation files. Modifications to the Class File, therefore, will not be at risk of being overwritten if *DL Pager* is reinstalled on a VMS system.

Output module configurations can be specified in several ways. Older *DL Pager* executables may have specific OM parameter values predefined within them, and these values previously could not be changed without rebuilding the *DL Pager* executable. Current *DL Pager* versions have the configurable output modules set to default values. All OM parameter values, whether set with a default Class File or built into the *DL Pager* executable, can now be redefined by reading an additional OM Class File from either the *DL Pager* Startup File or the PKC file used during composition.

A simple Startup File on VMS might look like:

```
ON MS=3
mem paragr:100000
RC "PAG$DISK:[PAGER.EXE]DEFAULT.CLASS"
```

where switches and memory values are set, and the Class File is specified with a directory path and is embedded in quotation marks.

The syntax in a PKC file is:

```
RC "thisfile.class"
```

where the filename, which can include an absolute or relative path, is enclosed in double quotes.

When including more than one Class File, the same output modules should not be specified in each file. OM parameters are set using the last configuration encountered.

A likely scenario is one where a Class File is referenced in the *DL Pager* Startup File in order to override the default values, and an individual user may wish to make further modifications without changing the Startup File which could affect other users. To do this, simply include an `RC` command in the `PKC` file that is used for composition. If a Class File is called in both the startup and the `PKC` file, the last-called class file is what is used by *DL Pager*. However, if the output modules configured by the configuration file referenced at start-up are distinct from the output modules in the Class File referenced by the `PKC`, all output module configurations will be used. In addition, if a particular configuration parameter has a default value and is not specified differently in the Class File referenced by the `PKC`, the default value is used.

If a Class File is referenced in the Startup File and a user wishes to replace this file with a local version, they must redefine `PAG_STARTUP` to reference a new start-up file in their area. This local Startup File can read the Class File they wish to use. This local Class File must include all parameters being configured to values other than the default.

> **NOTE:** Each Class File can be a subset of the normal class file (subset meaning blocks that begin with CLASS "om-name" and end with ENDCLASS), so you can have a Class File for each OM. Then, each OM can be called one-by-one in the startup file.

If using the *DL Publisher* interface to *DL Pager*, `PAG_STARTUP` can be easily defined using the **Define Variable** option under the tree path **Environment, Format Options, Variable Tracking**.

New DL Pager Installations

Upon installation, a default OM Class File will be placed in the *DL Pager* executable area. A copy of this file can be modified to include specific output requirements. If no modifications are desired, no further action is required. If Class Files are already in use, they can be used instead of the default file by reading them with the RC command in the Startup File. To use a modified Class File, follow the steps provided below.

Defining a Modified OM Class File as the Default at Start-up

- The default OM Class File resides in the *DL Pager* executable areas:
  VMS: `PAG$DISK:[PAGER.EXE]DEFAULT.CLASS`
  UNIX: `$PAG_DISK/pager/bin/default.class`
  Windows: `C:\Datalogics\PagerNT\Default.class`
- Modify a copy of this Class File and save it with a new name (`MODIFIED.CLASS`).
- Modify the Startup File referenced by `PAG$STARTUP ($ PAG_STARTUP)` by adding a `RC` (Read Class) command which references the new Class File:
  VMS: `RC "PAG$EXE:MODIFIED.CLASS"`
  UNIX: `RC "$PAG_EXE/modified.class"`
  Windows: `RC "C:\Datalogics\PagerNT\Modified.class"`
  The Startup File is typically `startup.pgr` in the `PAG$EXE ($PAG_EXE)` directory.
- If another file is to be used to configure an additional output module, another `RC` command can be specified in the `.pkc` file.

Overriding the Default Configurations with a Local Class File on VMS and UNIX

- Copy a version of the default Class File to a local area and modify as desired.
- Include an `RC` command in the `PKC` file specifying the Class File location:
  VMS: `RC USER$DISK:[COMPANY.USER]USER.CLASS"`
  UNIX: `RC "/some/local/area/user.class"`

Overriding the Default Configurations with a Local Class File on Windows

- Currently, a local Class File can be used only by redefining the Startup File to point to it.
- Redefine PAG_STARTUP to point to a local Startup File that includes the read class command:

  ```
  RC "C:\some\local\area\user.class"
  ```
- Using the *DL Publisher* client, PAG_STARTUP can be redefined with the **Define Variable** option under the tree path **Environment, Format Options, Variable Tracking**. If using the *DL Publisher* client in a network environment, all filenames must be specified with UNC paths.

Adding OM Class Files to Existing DL Pager Systems

In *DL Pager* systems that include pre-configured output modules, OM Class Files can be added to provide users with the ability to change configurations as necessary without having to rebuild the executable.

When using a Class File to change or update configurations, the file must include a complete listing of all the configuration values in use other than the default settings. Place the Class File in the PAG$EXE ($PAG_EXE) directory and modify the *DL Pager* Startup File startup.pgr to include an RC command as defined in a previous section titled: “Defining a Modified OM Class File as the Default at Start-up”.

Output
Module Parameters

This chapter explains the parameters available for working with each of the configurable output modules.

Understanding Startup Files and Class Files

Startup File The Startup File references the Class File. Class Files are called with the RC command in either the Startup File (defined by PAG_STARTUP for NT and UNIX or PAG$STARTUP for VMS) or in the PKC file. This variable will usually point to a file called startup.pgr in the *DL Pager* executable directory. There can be more than one RC command in either the Startup File or the PKC file.

For example, the following line in the .com file:

```
define PAG$STARTUP          SYS$DIR:PAGER-STARTUP.PGR
```

references the following Startup File:

```
ON MS=3
memparagr:100000
rc "pag$exe:ps.class"
rc "pag$exe:afp.class"
```

The first two lines set default values for certain switches. The lines beginning with rc reference Class Files; pag$exe:ps.class defines values for PostScript and pag$exe:afp.class defines values for AFP.

Class File The Class File contains information about each output module. It is read into the composition via the Read Class (RC) PKC command. A single Class File may be used for multiple OM classes, or each OM class can be specified by individual Class Files. When reading multiple Class Files in the Startup File or the PKC file, each Class File is referenced with an RC command.

A Class File lists each OM class to be configured. The OM classes are configured by pairs of name/value parameters. The parameters are initially set with the default values (these are listed within this chapter). A Class

File that includes Edgar, Xerox, and PostScript classes might be set up like:

```
CLASS "EDGAR"
DBLQUO      1
FRMFED      1
ENDCLASS

CLASS "Xerox"
Bpnull      0    ! nulls at the start of the page
Even        1    ! even records to a word boundary
Imgrot      1
ENDCLASS

CLASS "PostScript"
gdln1 "%DDatalogics %T  Job %j  %u/%p/%U %d %t"
allclr      1    ! to enable color output
ENDCLASS
```

Code Page

This file maps each text character to a glyph (the actual bit pattern of the character image) in a character set using a two-digit hex code point and an 8-byte data string character ID. This code point is used when referencing the character in other files. The maximum number of characters available on a single-byte Code Page is 256. Most characters have been identified by IBM with an EBCDIC string. The Code Page name is identified in the composition proceedure `.com` file.

Code Pages are only used in AFP OM Class Files. In the example below, `T1DLCHAR` identifies the Code Page. Notice this follows an exclamation point (`!`) which means that it is a comment and, therefore, not read by the processor. `A2EC129` references a specific character on that Code Page. `66` represents the EBCDIC character to be used in the translation. For more information, see “AFP Output Module” on page 4.7.

```
CLASS "AFP"
A2EC129 66   !ff ligature     T1DLCHAR
A2EC130 69   !ffi
A2ES128  220  !copyright
ENDCLASS
```

PostScript Output Module

In addition to the configuration parameters, guideline (or banner line) values can be set within the PostScript Class File. Guidelines contain information that describes the page and/or job; they can contain comments and information that has been defined by the operating system or by the Job File. Guidelines are printed on the typesetter page either above or below the composed page.

Part of a PostScript Class File might look like:

```
CLASS "PostScript"
gdln1 "%D Datalogics %T Job %j %u/%p/%U %d %t "
gdln2 "Style: %S File: %i  "
gdln3 "Seq: %s Color: %Cc: %cFormat: %fFree lead
%l%Fpoints, Next %n, Vjust %v"
allclr      1    ! to enable color output
clrno       1    ! to enable color number output
clrsep      1    ! to enable color separation
commnt      1    ! allow comment field
defoff      3    ! Default offset (P5 switch)
. . .
ENDCLASS
```

In the above example, the first three lines create guidelines that contain information passed by the “%” parameters. For example, %D passes the system date and %T passes the system time. There is no constraint on the length of an individual guideline field, and the length of the guidelines themselves are only constrained by the width of the page. Datalogics recommends keeping the guideline length to less than 512 characters.

Up to three guidelines may be specified within the PostScript class, and the number of guidelines to be used is given by the gidnmb configuration parameter. Guidelines can also contain comments in any position. Within the Class File, the OM configuration parameters follow the guideline definitions and are specified by parameter name and value, and may have a comment preceded by an exclamation point (!).

Guideline Values

Up to three guidelines can be set. Parameters that can be passed are:

Table 4-1 PostScript Guideline Parameters

Parameter	Description
%c	Comment
%C	Color number
%d	Date at start of composition
%D	Default style
%f	Format file name
%F	Page break flag
%I	Input file name
%j	Job number
%l	Free lead
%L	Note free lead
%n	Next lead
%N	Note next lead
%p	Project number
%s	Page sequential number
%S	Style file name
%t	Time at start of composition
%T	Typesetter identification
%u	User name
%U	UIC (user number/user identification code)
%v	Vertical justification information string
%%	A single %

Configuration Options

Table 4-2 PostScript OM Configuration Options

Parameter Name	Default Value	Description
allclr	1	Enables color output
clrno	1	Enables color number output
clrsep	1	Enables color separation
commnt	1	Allow comment field in guideline
defoff	3	Default offset (P5 switch)
defrot	-1	Default rotation (P1 switch)
defsca	5	Default scaling (P3 switch)
dmrflg	1	Execute predefined string, when extended [DM n]
fillen	40	Maximum file name character length
filnam	1	File name
fmtnam	1	Format name
frlead	1	Free lead
gidnmb	3	Number of guidelines defined, ‘1’, ‘2’, or ‘3’
imgrot	1	Enable image rotation
logfnt	1	Enable logical fonts
multsy	1	Allow multiple symbol fonts
outlay	1	Enable layered output
pbflag	1	Page break flag (m,f,*)
stynam	1	Style file name
vjust	1	Vertical justification flag

Xerox Output Module

Configuration Options

Table 4-3 Xerox OM Configuration Options

Parameter Name	Default Value	Description
bpnull	0	Nulls at the start of the page
even	1	Even records to a word boundary
imgrot	1	Enable image rotation
nulls	24	Nulls at end of job
phoml	24	Minimum length record
pnulls	0	Nulls at the end of the page
spool	1	Automatic spooling of output

AFP Output Module

AFP fonts can be named in the Class File rather than the font metrics file. The AFP OM uses T1DLCHAR and T1DLSYMB as the default Code Pages for all fonts and uses the internal ASCII to EBCDIC translation tables. The Class File allows a user to:

- specify different Code Pages for different fonts while using the defaults for any font not specified
- change the internal translation table(s)

- view the table(s) in the *DL Pager* Log File before and after changes, via the DUMP parameter.

```
CLASS "AFP"
DUMP "YES"
FONT0 "PHELN CHAR=(C0H200#0,T1000361) SYMB=(C0H201#0,T1001087)"
FONT1 "PTIME CHAR=(C0N200#0,T1000361) SYMB=(X0N201#0)"
FONT2 "PCOUR CHAR=(C0C200#0) SYMB=(C0C201#0,T1001087)"
FONT3 "PSYMB CHAR=() SYMB=(C0H201#0,T1001087)"
A2EC129 66   !ff ligature     T1DLCHAR
A2EC130 69   !ffi
A2EC131 70   !ffl
A2EC132 67   !fi
A2EC133 68   !fl
A2ES128  220  !copyright
A2ES129  255  !trademark
ENDCLASS
```

AFP OM Class File Encoding

Each position (32 through 255) in each translation table (character and symbol) have been assigned a value that can be included in the Class File to change the resultant translation. The Class File variables are named in the form:

```
A2EXaa  ee
```

where:

- A2E represents *ASCII to EBCDIC*
- X can be either C for *character table* or S for *symbol table*
- aa is a number between 32 and 255 that represents the ASCII value to be translated
- ee is the decimal value of the resultant EBCDIC character.

For example, an exclamation point has an ASCII value of 33 and an EBCDIC value of 4F, which is decimal 79. A question mark has an ASCII value of 63 and an EBCDIC value of 6F, which is decimal 111. To cause all of the exclamation points in the input to be translated to question marks in the output and all of the question marks in the input to be translated to exclamation points in the output, the Class File would need the following two lines:

```
A2EC33  111
A2EC63  79
```

NOTE: Be careful to consider the printed output when making these character translations. In the above example, the setwidth of a question mark is greater than that of an exclamation point (except when a monospace font is in use) and this may cause undesirable results.

To dump the internal translation tables to the Log File, set the `DUMP` parameter to **Yes**.

AFP OM Class File Syntax

In the AFP OM Class File, the following syntax rules apply:

- All values must be supplied in upper-case
- Quotes, spaces, commas, and parenthesis are significant
- All information on the same line following an exclamation point is considered to be a comment
- `A2E` entries should be grouped according to the table (character or symbol) they reference; all `A2EC` entries together and all `A2ES` entries together
- There is a maximum of 256 entries (`FONT0` through `FONT255`) for a job
- `FONT#` entries (`FONT0` through `FONT255`) should be consecutively numbered
- Code pages cannot be specified unless a font value is also specified
- Any value that is acceptable in the **NAME** and **FONTNAME** entries is acceptable here
- *DL Pager* font metrics file names are limited to 6 characters, AFP font names are limited to 8 characters.

More information about AFP OM Class Files, Code Pages and the `.FNT` file is available in the *DL Pager AFP Output Module User Guide*.

Edgar Output Module

Edgar Decode Table Values (Character Mapping)

The Edgar Output Module may be configured with as many as 11 character translation tables, providing the ability to use different character mappings for different compositions without changing the output module configuration. The Edgar translation tables listed in this section are the default character mappings. For a discussion on accessing specific tables during composition, see the *DL Pager Edgar Output Module User Guide, Chapter 5, The OE Command.*

There can be as few or as many `Dectb` mappings as necessary. The `Dectb` values are set using the syntax:

```
dectbl "145:0"
dectbl "146:1"
dectbl "148:2"
```

These particular mappings are defined in the following table. The number preceding the colon (:) is the ASCII character number that is mapped to the character that follows the colon (:).

Table 4-4 Edgar OM dectbl Character Mappings

Reference	Character	Reference	Character	Reference	Character
124	--	129	ff	130	ffi
131	ffl	132	fi	133	fl
145	0	146	1	148	2
149	3	151	4	153	5
158	6	160	7	161	8
162	9	168	0	194	1/3
199	2/3	202	1/8	203	3/8
204	5/8	205	7/8	206	1/5

Table 4-4 Edgar OM dectbl Character Mappings (Cont.)

Reference	Character	Reference	Character	Reference	Character
208	1/2	209	1/4	210	3/4
212	2/5	237	3/5	244	1
245	2	246	3	247	4
248	5	249	6	250	7
251	8	252	4/5	254	9

Table 4-5 Edgar OM dectb1 Character Mappings

Reference	Character	Reference	Character	Reference	Character
48	zero	49	one	50	two
51	three	52	four	53	five
54	six	55	seven	56	eight
57	nine				

Table 4-6 Edgar OM dectb2 Character Mappings

Reference	Character	Reference	Character	Reference	Character
48	null	49	eins	50	zwei
51	drei	52	vier	53	fuenf
54	sechs	55	sieben	56	acht
57	neun				

Table 4-7 Edgar OM dectb3 Character Mappings

Reference	Character	Reference	Character	Reference	Character
48	null	49	odin	50	dva
51	tri	52	cheteri	53	pyat

Table 4-7 Edgar OM dectb3 Character Mappings (Cont.)

Reference	Character	Reference	Character	Reference	Character
54	shest	55	sem	56	vosem
57	devyat				

Table 4-8 Edgar OM dectb4 Character Mappings

Reference	Character	Reference	Character	Reference	Character
48	null	49	odin	50	dva
51	tri	52	cheteri	53	pyat
54	shest	55	sem	56	vosem
57	devyat				

Configuration Options

Table 4-9 Edgar OM Configuration Options

Parameter Name	Default Value	Description
DBLQUO	1	Sets two single quotes to one single quote
FRMFED	1	Form feed after each page
IMGROT	1	Enable image rotation
MAXLEN	78	Maximum length of text line
TABLEN	131	Maximum length of table line

NOTE: The DBLQU2 configuration parameter is not included in the default Class File but is necessary for converting two single left-hand quote characters into one double quote character. If this functionality is needed, the ASCII number 96 must be mapped to a single quote (‘) and the default Class File will have to be modified to include:

```
DBLQU2   1  ! Sets a single quote character generated from
            ! combining two single quotes into one double
            ! quote character.
```

NOTE: A WRN 100 message means the following:

page > 100 lines.

Index