POSTSCRIPT

POSTSCRIPT® PRINTER DESCRIPTION FILES Specification Version 3.0

January 16, 1989
PostScript® Developer Tools & Strategies Group

Adobe Systems Incorporated
1585 Charleston Road PO Box 7900
Mountain View, CA 94039-7900
(415) 961-4400

PN LPS5003

POSTSCRIPT® PRINTER DESCRIPTION FILES Specification Version 3.0

January 16, 1989
PostScript Developer Tools and Strategies Group

Adobe PostScript Printer Description files (also known as “PPD” files), are designed to be human-readable, machine-parsable text files useful for determining and using the special features on PostScript printers. These features include, for example, different paper sizes and font handling capabilities.

Printer Description files are intended to be parsed by a document manager in preparation for producing a print job for a particular printer. At this stage, presumably the decisions about paper size and fonts have been made (by the document composition software) and the spooler or document manager must invoke these features correctly for the chosen printer. The Printer Description files are intended to make that process easier. There is a particular relationship between these files and the Adobe Systems *Document Structuring Conventions, Version 3.0*. These comment conventions provide a mechanism for communicating a document’s needs to a spooler by specific *keywords* indicating paper sizes or other printer features. The spooler or document manager can then parse the Printer Description Files to satisfy those needs. It is highly recommended to read the document describing these structuring conventions.

NOTE:
Revision 2.1 of this document contains color keyword extensions. This revision also contains some substantive changes to the content, focused mostly on the query entries. In particular, there is a much more rigorous discussion of what should be returned by the queries, and in some cases it has actually changed from previous versions. Please be careful in this area.

1. USING PRINTER DESCRIPTION FILES

A Printer Description file provides very *device-specific* information about how to invoke particular features on a given PostScript printer. This information should never be used to compose the original PostScript language document file, since all of it is inherently device-dependent. A PostScript language file should be composed in a *device-independent* manner, permitting any size document, any font selections, and any of the “mainstream” PostScript language operators, but should contain *no explicit system-level PostScript language* fragments. Please refer to documentation on *Encapsulated PostScript* files for further guidelines.

When it is time to actually *print* a document, then it is appropriate to use a Printer Description file. The document file should be *parsed* to determine if any printer-specific features are needed (perhaps a particular paper tray or a font), and the Printer Description file should be consulted based on that need. Any device-specific setup code that is used from the Printer Description files should be carefully embedded in appropriate *comments*, and should be output *ahead* of the document file itself.

Here is an example print stream as sent to a PostScript printer, after having inserted code fragments from a Printer Description file:

```
%!PS-Adobe-3.0
%%Title: test.ps
%%EndComments
 % prologue definitions here
%%EndProlog
%%BeginSetup
%%BeginFeature: *PaperTray Ledger
 statusdict begin ledgertray end
%%EndFeature
%%EndSetup
%%Page: one 1
%%BeginPageSetup
 90 rotate 0 -612 translate
%%EndPageSetup
 % executable code for page 1
%%Trailer
```

Printer Description Files, as supplied by Adobe Systems or printer manufacturers, are *not* intended as an editable format. In particular, it is not appropriate to use the Printer Description file directly as a local “customization” file for a particular printer, nor to attempt to maintain the current state of a printer by updating the file. The list of fonts contained in the Printer Description file will be those that are *guaranteed* to be resident in the device when it is shipped from the factory (essentially, this means *only* ROM-resident fonts). Any fonts which may or may not be present (for instance, on the disk, in font cartridges, or on a font server) will not be listed in a Printer Description file. Please see Section 2.9 for a little more detail on customization.

Printer Description files are intended as a *starting point* for the task of managing a PostScript printer. The management task itself requires careful attention to the state of the machine, whether or not it is idle, busy, or out of paper, and whether or not it may contain a particular font or paper tray at any given moment. These issues are not directly addressed by the Printer Description files.

Each printing environment is expected to provide its own solution to *managing* a PostScript printing device. In that regard, the Printer Description files should be *parsed* and perhaps their contents should be used as the basis for an *editable* representation that may be used by a printing manager to maintain an accurate model of any particular instance of a PostScript printer. If used *directly* by printing managers, the files *should not be modified* from their original distribution format.

There are discussions of both local customization and foreign language customization later in this document.

2. THE FORMAT

Printer Description files provide three fundamental kinds of information: what the *options* are, what the *default* is, and *how to invoke changes*. This is addressed by a simple *line-oriented format* where the options, defaults, and invocation strings are made available through a regular set of *keywords*. Where applicable, there is also information supplied about how to find out the *current* setting of any of these configurable options (known as *querying* the printer). This can be useful for spoolers and document managers to determine the state of a printer and perhaps request operator intervention (for example, if the appropriate paper tray is not present).

The format of the Adobe Printer Description files is intended to be *extensible*. That is, additional keywords can be added to the format as necessary to address specific needs.

NOTE:
This document represents a keyword registry maintained by Adobe Systems. All valid keywords to be found in Printer Description Files are documented herein. As new keywords are added to extend the format, Adobe will update the document. We will send out updated lists if and when any keywords are added. Releases of the registry are uniquely identified by the date of the release. Please send requests to the address on page one of this document:

The design goal of the format is to be line-oriented and easily parsable using extremely simple searching strategies. As a minimal requirement, no single keyword will be wholly contained as a substring in another keyword, so that line-oriented searching programs like **grep** can be used without surprises. For example, there will not be two keywords such as “*Paper” and “*PaperSize”. All main keywords start with the same initial character (in particular, the **‘*’** symbol**,** which is decimal ASCII 42). This makes recognition of the keywords easier, and will help to reduce the possibility of the keywords being confused with PostScript language identifiers inside data segments.

2.1 DETAILS

The line length of any line in a Printer Description file shall be less than or equal to 255 characters, including line termination characters. Line termination in Printer Description files may consist of any combination of carriage return (decimal ASCII 13) and line feed (decimal ASCII 10). These two characters should be thought of as merely “white space” characters. Different computer systems may represent the line endings differently without affecting the actual data representation.

All main keywords will start with the leading special character **‘*’** (decimal ASCII 42). A complete list of keywords appears later in this document. Keywords for “querying” a printer are introduced by **‘*?’**, which is differentiated by the presence of the **‘?’** character (decimal ASCII 63).

Since the format is *line-oriented*, it is expected that all main keywords will start at the *beginning* of a line.

The basic format looks like this:

Default<paramtype***>: <***option***>**
<paramtype***> <***option***>: "***PostScript language code***"**
<paramtype***> <***option***>: "***some other PostScript language code***"**
?<paramtype***>: "***PostScript language query code***"**

An example entry:

```
*DefaultPaperTray: Letter
*PaperTray Letter: "lettertray"
*PaperTray Legal: "legaltray"
*?PaperTray: "[(Letter)(Legal)] currentpapertray get == flush"
```

The format conveys what the *possibilities* are for the particular feature that is wanted, what the *default* setting is for this feature, and how to *invoke* each of the options. *Any lines that start with the same keyword shall be contiguous in the file* (to make it easier to collect them).

The information is represented as “tuples.” They will typically either be *2-tuples* (keyword/value pairs) or *3-tuples* (keyword/option/value triplets). Where simple information is supplied, like the name of the printer, a simple key/value pair is used. Where there are *optional parameters*, triplets are used (as in the example above) to provide information about a specific option. A colon **‘:’** (decimal ASCII 58) is used to separate them. Any number of tabs and spaces are permitted between the colon and the value or option keyword. Some of the keywords will require a *value* that consists of more than one component. For instance, the ***Font** entry requires several pieces of information for each font. Please see the specific entry for each keyword for more details on the syntax. A simple key/value pair looks like this:

```
*MainKeyword:  value
```

and a 3-tuple typically looks like this:

```
*MainKeyword option:  value
```

If a *value* is in a form that is usable directly in the PostScript language (for instance, if the value is an integer constant that makes sense in the PostScript language, or a PostScript language string of some sort) then the value will be embedded in quotation marks in the manner of the more generalized PostScript language sequences. This allows values to be stripped out of the Printer Description file and placed directly into PostScript language files being generated. Keywords that are meaningful to a printing manager but which are not strictly PostScript language sequences will *not* appear in quotes. In particular, the values of the ***Default** parameters will be *keyword references* (rather than PostScript language strings), and should *not* appear in quotes. See the Example File at the end of this document for an illustration of this concept.

2.2 SEMANTICS OF MAIN KEYWORDS

There are several “kinds” of keywords. Keywords that start with **"*Default"** supply information about the default condition of one of the parameter types. The same keyword without the **Default** prefix provides either *information* about a feature or *invocation strings*, depending on the feature. Keywords that begin with **"*?"** indicate a PostScript language *query* that can be downloaded to a printer to obtain current state information about a particular feature. For example, the ***?ManualFeed** entry will contain a PostScript language fragment that returns either "**True**" or "**False**", depending on the current setting of the **manualfeed** feature on the printer. Other queries may return other values, depending on what keywords are appropriate for that option.

For most special features of a printer, there is no clear inverse operation. That is, "unsetting" something like a ledger-size paper tray will normally mean just "setting" whatever the default paper tray is. Therefore, it should be understood that explicitly setting the printer back to its default condition will "undo" the effects of having previously set a given feature. Unless there is a specific reason to do so, it is not necessary to *reverse* the effects of invoking printer-specific features for any particular print job, since the job server should provide that service.

2.3 OPTION KEYWORDS

Option keywords are provided whenever there may be several options to choose from for a particular feature. As an example, there may be many different paper sizes listed in the ***PaperTray** section. These options are specified using *option keywords*. These keywords are registered by Adobe Systems and their semantics are specified later in this document. The keywords themselves can optionally have *qualifiers* as extensions. These qualifiers will also be registered when appropriate, with the exception of the *serialization* extensions. These qualifiers are *appended* to the option keywords with a "dot" (or "period") separator character (decimal ASCII 46). Any number of these may be appended to a particular keyword, as appropriate. Here is an example:

```
*PaperTray Letter: "0 setpapertray"
*PaperTray Letter.Transverse: "1 setpapertray"
*PaperTray Letter.2: "2 setpapertray"
```

Currently these qualifiers are only defined for the *paper keywords*, to differentiate between several instances of a particular paper type that may differ slightly (as in this case, where **Letter** differs from **Letter.Transverse** only in the direction that the paper is fed into the printer). The *serialization* extensions are simply *integers* appended to the keywords to distinguish between otherwise identical features.

There is an optional foreign language translation string syntax available for option keywords. It is detected by the presence of a slash (**'/'**) character (decimal ASCII 47) after the keyword itself, and before the corresponding colon (**':'**). This string, if present, *must* be accounted for, and not confused with the PostScript language sequence that follows. These translation strings are used primarily in foreign language user interfaces.

2.4 HUMAN-READABLE COMMENTS

Comments are supported in the Printer Description files through the use of the keyword **'*%'**. Anything following this keyword (through the end of the line on which it appears) should be ignored by a parsing program. The **'*'** is the same introductory symbol used for all main keywords, and the **'%'** is borrowed from PostScript language syntax as its comment character. These comments will begin only in column one, for simplicity.

2.5 PARSING DETAILS

Here are some lexical rules that can be applied to writing a parser for Printer Description files:

- The **‘*’** metacharacter will only appear in the *first column* of a Printer Description file. That is, main keywords will always begin at the start of a new line.

- There shall be no spaces or other characters *before* the colon separator, but there may be spaces and/or tabs *after* the colon, as in most of the examples in this document.

- Entries that are PostScript language segments shall be enclosed in double quote characters (decimal ASCII 67). When an open quote is encountered, parsing should continue until the matching close quote is found, even if it crosses a line boundary. Line boundaries are considered to be *significant* “white space” within a PostScript language sequence. That is, lines will not be broken in the middle of PostScript language tokens. If a code segment breaks across a line, then the ***End** keyword should be found as the next entry in the Printer Description file after the close quote delimiter. If it is *not* found, this may be considered to be a *parse error* with a missing close delimiter. The ***End** keyword will only appear where code segments extend across a line boundary.

- The *case* of keywords is significant. That means that **letter** is distinct from **Letter**.

- Keywords may contain any *printable* ASCII characters within the range of decimal 33 to decimal 127.

- Delimiters between keywords shall be *spaces* or *tabs* (with the exception of the colon separator, discussed elsewhere).

- A PostScript language sequence contained in double quotes should be considered to be a single “token” when parsing Printer Description files.

- The *absence* of a keyword means that the feature does not exist (or does not make sense) on that particular printer.

- After the initial **‘*’** symbol is recognized, any characters through the next *tab*, *space*, or *colon* character is considered part of the main keyword.

- If a main keyword is not terminated with a *colon*, an *option* keyword can be expected. The option keyword is terminated either by a *colon* or by a *slash*, in instances where translation strings may be present. In this second instance, the translation string is terminated by the *colon* (spaces are part of the string).

- Keywords cannot contain spaces. This means that **"Quoted Keywords"** are *not* allowed. Again, quotation marks are only used where PostScript language sequences are recognized.

- Keywords will never exceed 40 characters in length.

- If a keyword is not recognized, it should be skipped.

- Foreign Language translation strings are *optional* in this format, and all parsers should be written to *permit* them without requiring them. English language versions of the files will be shipped *without* translation strings at first, but these may be added at any time if deemed necessary.

- A file included with the **"*Include:"** keyword should be treated as though it were "in-line" in the original file. Be prepared for nested includes. The semantics of repeated entries and keywords is up to the environment; the “generic” Printer Description files will *not* have **"*Include:"** in them.

2.6 POSTSCRIPT LANGUAGE SEQUENCES

The PostScript language strings that are supplied for invoking special features are always delimited by *double quotes* (67 decimal ASCII). This is a very simple and common delimiter, and happens to appear only very rarely in PostScript language sequences. By using this convention these files can be parsed with existing software tools quite simply.

For *multiple-line* PostScript language listings, the optional keyword ***End** is used as an extra delimiter to help line-extraction programs (like *grep* or *awk* under Unix), and for better human readability (the quote delimiter is sometimes hard to see at the end of a long string of code). This extra keyword will not be needed by most parsers, since the delimiting quotation marks uniquely define the code segment. It will only be used in the case where the code requires more than one line in the Printer Description file. The keyword ***End** should be ignored by all parsing software. Here are two examples, one of which fits on one line, the other of which is an “extended” code fragment:

```
*PaperTray Legal: "serverdict begin legaltray end"
```

```
*?PageRegion: " newpath clippath pathbbox 4 -1 roll = 3 -1 roll = exch = = flush "
*End
```

The PostScript language fragments supplied in the Printer Description files are only *guaranteed* to work on the printer for which the file was prepared. The fragments assume the *default state* of the interpreter. Only **userdict** and **systemdict** are assumed to be on the dictionary stack. The operand stack and dictionary stack will be left undisturbed after the execution of any given PostScript language sequence from a Printer Description file. There will be no memory use (**save** and **restore** are used where appropriate) except as in setting frame buffers, where memory use is necessary.

2.7 PAPER HANDLING

Probably the most common use of Printer Description files will be to be able to take advantage of many different paper sizes. There are many paper-handling features available on different PostScript language devices that currently exist, and undoubtedly more will come. In order to deal with these in a fairly general way, several different keywords have been devoted to paper handling in the Printer Description files.

It is assumed that in many instances what is desired is, at a user level, “please print this on ledger paper.” Seldom will the user need or care to know about how the paper was actually provided through the underlying PostScript language mechanism. Toward this end, there is a keyword provided (***PageSize**) with those (vaguely imprecise) semantics: “please give me ledger paper” (or other paper sizes). The actual invocation of that paper type may vary from one device to another—it may require use of **setpageparams** on one device, and perhaps **setpapertray** on another.

Unless there are special paper handling needs, it is suggested that the ***PageSize** keyword be used to set a paper type.

For more sophisticated control over the paper handling capabilities, there are sections of the Printer Description files for controlling the *paper trays* directly, the *input* and *output bins* (which may be useful if the user has specifically placed letterhead in the top tray, for example), the *output order* of the pages, and the *imageable area*. Each of these has a specific use that may be needed beyond the simplistic notion “please give me ledger paper.”

In general, it is best to invoke a *paper tray* operator to get the desired size of paper. Most of the invocation strings provided in the ***PageSize** section will in fact invoke the paper tray operators directly). This may provoke a **rangecheck** error if the paper tray is not inserted, which can be caught by the application and a message can be generated. If **manualfeed** is used, the *page region* operators may be used to invoke a particular imageable area for the manually fed sheet. Selecting a particular *input slot* may occasionally be useful for specialized applications which may have letterhead in one slot and regular paper in the other.

NOTE:
*Only one of these keywords should be used to set the desired size of paper used. Under normal circumstances, the *PageSize operator is the right choice, unless manualfeed is used or there is a specific need for a paper tray operator.*

In addition to the *invocation* information for the various paper types, information is provided about each nominal size. For instance, the physical *paper dimensions* are provided through the ***PaperDimension** keyword, and the actual area on the page which is “writable” by the PostScript interpreter is provided through the ***ImageableArea** keyword.

2.8 FOREIGN LANGUAGE CUSTOMIZATION: TRANSLATION STRING SYNTAX

There are some entries in the Printer Description files that may be encountered at the user interface level, including the option keywords and the printer messages. There is an *optional* syntax to provide translation strings for these keywords and messages that is based on the slash character (**‘/’**) (decimal ASCII 47). The only purpose of this is to provide an alternative string that may be used to display messages to human users of the systems, especially those in non-English-language environments. There will be a keyword entry in the printer description file that marks the particular language, to make clear in which language these strings are expressed. All information in a Printer Description file *except* these translation strings shall be *identical* for each foreign language version of a given Printer Description file. There should be no other tangible differences. Here is an example of the translation string syntax:

```
*LanguageVersion: French

*PaperSize Ledger/Papier Ledger: "statusdict begin ledgertray end"
*PrinterMessage: "%%[ PrinterError: out of paper ]%%" / "%%[ Il n’y en a plus de
papier. ]%%"
```

The idea behind this is that the keywords themselves should not change, or a parsing program would have to learn a new set of keywords for each language. Instead, the normal keyword searches can be carried out, and the optional translation strings can be presented to the users instead of (or in addition to) the keywords.

2.9 LOCAL CUSTOMIZATION AND "*Include:"

A Printer Description file is a *static* representation of the features available on a PostScript printer. It contains information on the features available on a printer as it is shipped from the factory. The task of *managing* a PostScript printer is a dynamic issue that requires keeping track of fonts downloaded to disk, error handlers, RAM-based fonts and procedure sets, and so forth. This kind of printer management is beyond the scope of the Printer Description files. However, the need to “customize” the files to adapt them to local instances of printers is recognized.

The proper way to do this is to permit *local copies* of Printer Description files. In any given computing environment, there should be a single Printer Description file for each type of PostScript printer in use. If particular software products (or individual users) wish to add to or modify the contents of a Printer Description file, they should do so by *copying* the original file and making local changes, so that the “master” copy is not altered. To facilitate this, there is also a provision for *including* files through the keyword **"*Include: *filename*"** which can appear anywhere in a Printer Description file. This permits small changes or additions to Printer Description files without having to make an entire local copy (updates to the “included” file are also automatically incorporated with this scheme).

When a Printer Description file is included into another file, the parsing details change somewhat. In particular, there may be several instances of the same keyword in the “composite” file. There are basically two choices:

- The *last* instance of a keyword is the correct one.
- The *first* instance of a keyword is correct.

This is left to the particular environment. Files shipped from Adobe will *not* contain the **"*Include:"** keyword. The syntax is specified here merely to provide the capability in a standard way.

3. COLOR EXTENSIONS

Color separations are very device-dependent. A color separation is a monochrome print that represents a single color plate which is later printed in combination with other plates on a full color press system. In this sense, a color separation may be one of the four standard *process colors* (cyan, magenta, yellow, and black) from which all other colors are simulated by mixing, or it may be a particular *spot color*, which is simply an ink of a particular color.

In order for color separations to work well, it must be possible to print several layers one on top of the other on a color printing press. The way the color mixing is optimized is to print each color plate with a different *halftone* screen, usually rotated at some specific angle to minimize both dot interference with other plates and to avoid moiré patterns. The selection of these halftone screens is typically done carefully by hand for a particular device, taking resolution and other device characteristics into account (even variations in the speed of paper travel). Once a good set of screen parameters have been established, they are used for almost all separations on that machine, unless screens of different granularity are desired, in which case the process is repeated.

In addition to the halftoning process necessary for producing separations, there are issues of color matching which are equally device-dependent. As an example, many companies have specific *names* for their entire range of colored inks. These colors can be simulated or approximated with various color technologies (screen phosphors or process inks) but it may not be possible to render them exactly. There is usually a color mapping table that associates a particular combination of process inks (or screen phosphor intensities) to one of the named colors. This is, of course, very device-specific.

Toward this end, Adobe has added some extra parameters to the existing Printer Description File format so that a printer description file can contain necessary information for color matching and producing good color separations.

This document documents only the extensions to the existing format specification. Please refer to the original document on the Printer Description File format for further information. These extensions will be folded into that document for future distribution, but are isolated into a separate document for easier reference as the world of process color unfolds.

4. LIST OF KEYWORDS

Here is a list of the currently defined keywords and a description of their use. The keywords are grouped according to their type, and there is also a list of *standard values* for the defined keywords. The format of this section of the document is to show the *keyword*, what the possible *options* are, and a pseudo-code syntax to illustrate its *value*. Here is what the format of the examples in the section will look like:

*MainKeyword Option1/Option2/Option3: "invocation"

This indicates that for the keyword ***MainKeyword**, there are three viable options (**Option1**, **Option2**, **Option3**), and the appropriate syntax for the *value* of the tuple is a PostScript language invocation string enclosed in quotation marks. This notation is used throughout this section except where otherwise noted.

† Note that a few of the keywords may require exiting the server loop for correct execution. These have been flagged with the dagger at right. The value suppled by the ***Password** entry should precede the code, or an alternate **exitserver** password as supplied by system software or the user. The code contained in these flagged entires will check for the existence of a valid password on the operand stack when executing.

4.1 GENERAL DEFAULTS AND INFORMATION KEYWORDS

*LanguageVersion: *languagekeyword*

This provides a keyword indicating the natural language used within the body of the Printer Description file. The language (for instance, **French** or **German**) affects *only* the human-readable comments and the *translation strings* provided in the PrinterMessages section and for the option keywords. See previous discussion in this document for more details.

*FormatVersion: *"string"*

This provides the format version number for the Printer Description file format. If the files change for some reason, this version number will reflect any incompatible differences. A standard version numbering scheme will be employed, where digits to the left of the decimal imply incompatible changes, and digits to the right of the decimal imply more minor revisions. Any revisions to the format should be accompanied by appropriate documentation, in any case. To conform to the specification detailed in this document, the string should be **“3.0“**.

*FileVersion: *"number"*

This keyword is used simply to identify the version number of the file itself. It is used only to distinguish between releases of the same file, not to distinguish one file from another. For example, all of the released printer description files will start out with 1.0 in this field, and if they are re-released for any reason (bug fixes), this version number will be increased in the new file. This permits the files to be otherwise identical in most ways (including file name) but still be distinguished easily from one another.

*PSVersion: *"(string) integer"*

This corresponds to the PostScript interpreter's *version* number as returned by the PostScript language **version** operator and the interpreter's *revision* number as returned by the **revision** operator. The values are presented in PostScript language form in order that they may be compared with the actual values in the printer to determine whether or not the Printer Description file matches the printer. There may be *more than one instance* of the ***PSVersion** keyword if the Printer Description file is valid for more than one version (and revision) of the interpreter. Also, please note that it is neither necessary nor sufficient for the version number to match. That is, two versions may have equivalent Printer Description files, and, more importantly, two interpreters with the same version number and product name may in fact require *different* Printer Description files. An example of this is a manufacturer that may use a single controller to drive several different marking engines.

The real responsibility of matching Printer Description files with physical printers lies with the system administrator and/or system software that uses these files. It may simply ask the user to select one, or the installation process makes the proper association.

*Product: *"(string)"*

This corresponds exactly to the product string of the printer, as stored in **statusdict**. Its value should be the same as returned by the fragment:

```
statusdict begin product == flush end
```

There may be *more than one instance* of the ***Product** keyword if the Printer Description file is valid for more than one product.

*NickName: *"(any name)"*

This is used simply to give a human-readable name for the printer. It is used primarily at the user interface level when selecting a printer, or to distinguish between two otherwise indistinguishable printers (for example, if a single RIP is used to drive more than one type of marking engine). The contents of this field are originally set to be some "reasonable" description of the printer (for example, **(Linotronic L300)**, **(DataProducts LZR 1260)**, **(Apple LaserWriter II NTX)**, etc.).

*DefaultResolution: *resolutionkeyword*

This entry notes the default nominal resolution of the printer, in pixels per linear inch in both X and Y dimensions. Square pixels are assumed, and there is only *one* value given in this entry. It is actually a *keyword*, in that the ***resolutionkeyword*** can be used to determine a PostScript language invocation string by consulting the ***SetResolution** entry. The possible ***resolutionkeyword*** values are constructed from the number of device pixels per linear inch with the characters **"dpi"** appended (for example: **300dpi**, **1270dpi**). Only **dpi** is supported as a unit of measure. Be careful not to use these to produce device-dependent PostScript language files!

† *SetResolution *resolutionkeyword*: *"invocation"*

In PostScript language implementations which support resolution changes from software, this entry will provide the proper invocation string for each resolution supported by the device. There may be several lines of these, if the PostScript device will support them. The possible ***resolutionkeyword*** values are constructed from the number of device pixels per linear inch with the string **dpi** appended (like **300dpi**, **1270dpi**, etc.).

*?Resolution: *"invocation"*

This is used to query the PostScript server for the current resolution. The returned value shall be a ***resolutionkeyword*** as in ***DefaultResolution** above.

*ColorDevice: True/False

This entry provides a flag that indicates whether or not the device supports color. The absence of this keyword implies a black and white device.

4.2 SYSTEM MANAGEMENT

† *PatchFile: *"arbitrary PostScript language"*

This is used to represent a (perhaps large) PostScript language sequence representing a downloadable patch to ROM code. It may be used if there are any known bugs in existing PostScript devices, or to provide some initial state to all jobs. A program that is attempting to manage a PostScript printer should make every attempt to guarantee that this information is resident in the PostScript interpreter VM.

*Throughput: *"integer"*

This is the nominal throughput in pages per minute. It should represent the marking engine capacity for throughput. It may perhaps be used to determine the fastest of a number of printers if there are many to choose from, but should not be construed as any kind of "benchmark" figure. In the case of roll-fed machines, the number indicates the number of 8-1/2 inch sections of paper can be fed in one minute by the marking engine. If the value is fractional, it should be rounded up to the nearest number (it should not be 0 unless the marking engine is broken).

*FreeVM: *"integer"*

This is the value as returned by the PostScript language fragment **"vmstatus exch sub == pop"** when a printer is first powered on. It may be used by a document manager to determine which of several printers has more VM built in to it.

† *Reset: *"invocation"*

This is a PostScript language sequence that will perform a "soft" restart of the PostScript interpreter. It may be used by a printing manager to reboot the printer under some circumstances.

† This keyword requires the *Password value to be supplied in front of the invocation.

*Password: *"arbitrary PostScript language"*

This provides the default **exitserver** password for the printer. It should be used in conjunction with the ***ExitServer** keyword.

† *ExitServer: *"invocation"*

This provides the appropriate PostScript language sequence to exit the job server loop (typically using the **exitserver** operator). This should be used very carefully if at all by a printing manager. Its sole purpose is to cause effects to the memory of the printer to be “permanent” until the printer is turned off. It is usually only appropriate for error patches or to change the system defaults on a printer. The value of ***Password** should precede this string in the PostScript language invocation.

*FileSystem: True/False

This provides a **True** or **False** value depending on whether or not the PostScript device has a built-in file system. Normally this means the presence of a hard disk or SCSI controller on the printer. This information may be used by a printing manager to determine the capability for internal file system support. Some devices may have the capability for a file system but may not in fact have a disk installed. The ***?FileSystem** query may be used to dynamically determine this. The absence of this keyword implies no file system, although if the file system query is available, that is a better way to find out.

*?FileSystem: *"query"*

This is used to query the PostScript device to see if a file system is currently installed and active. The query will return either **True** or **False**.

*DeviceAdjustMatrix: *"[transformation matrix]"*

This entry provides a device-specific transformation matrix to compensate for any anamorphic scaling or offset problems inherent in the underlying mechanical marking device. This entry will normally be shipped as the identity matrix [1 0 0 1 0 0], but may be modified locally for a particular printer to compensate for slight shrinkage or magnification caused by motor speeds, paper thicknesses, and so forth.

NOTE:
The ImageableArea figures given in the Printer Description files will no longer be exactly accurate if the device matrix is adjusted. Bear in mind, if this field is changed, that any operations that are sensitive to the page boundaries may have to be recomputed slightly (or the results may be off the page, in the worst case).

4.3 GRAY LEVELS & HALFTONING

*ScreenFreq: *"integer"*

This is the second argument returned by the **currentscreen** operator. It is the halftone screen frequency.

† This keyword requires the *Password value to be supplied in front of the invocation.

*ScreenAngle: *"integer"*

This is the first argument returned by the **currentscreen** operator after powering on the device. It represents the halftone screen angle.

*DefaultScreenProc: Dot/Line/Ellipse/Cross/Mezzo

This represents the procedure returned by the **currentscreen** operator (the default halftone spot function). Any of these keywords may also have a **.Invert** qualifier which would invert the color of the spot function.

*ScreenProc: Dot/Line/Ellipse/Cross/Mezzo: *"{ procedure }"*

This represents a procedure body appropriate for use as a “spot function” with the **setscreen** operator. These are used to specify an alternate shape for the halftone spot. There may be one or more of these spot shapes present in the Printer Description file. Any of these keywords may also have a **.Inverse** qualifier which would invert the color of the spot function.

*DefaultTransfer: Null/Normalized

This is the default transfer function, as returned by the **currenttransfer** operator. The transfer function may be a “null” function (like this: **"{}"**) or it may be a *normalized* transfer function. A normalized transfer is one which corrects for the characteristics of the marking engine or display technology to obtain “true” optical gray densities. A normalized transfer function is expected to return accurate results at the 10% increments, and should return reasonable values at any point between **0** and **1**. Either of these may also have the **.Invert** qualifier as may the ***ScreenProc** entries. The inversion is typically performed by appending **"1 exch sub"** to the existing transfer function, but an inverse normalized function may be more complex.

*Transfer Null/Normalized: *"{ procedure }"*

This keyword provides possible transfer functions which may be invoked with the **settransfer** operator. When these are used at the PostScript language level, be careful to always *concatenate* the transfer function with the existing one, rather than replacing it. Either of these may also have the **.Invert** qualifier as may the ***ScreenProc** entries. The inversion is typically performed by appending **"1 exch sub"** to the existing transfer function, but an inverse normalized function may be more complex.

4.4 PAPER HANDLING

Many of the paper-handling capabilities of PostScript language implementations are predicated on a fixed notion of particular sizes or classes of paper. In the Printer Description files, these page sizes are represented by specific keywords, each of which has unique semantics. The files are organized in such a way that if a particular paper size is desired, it is easy enough to parse out an appropriate invocation string to set that paper type. An additional goal of the format is to be able to determine a *list* of all paper types (even if they were not known beforehand) and to be able to determine the salient features of each page size (for instance, the paper dimensions, the orientation). There will be a ***PageSize** entry for every page size supported by the printer, for example, and it is easy to enumerate them. Here some examples of keywords that are valid wherever ***PaperKeyword*** is used in this format:

A3	A4	A5	B3
B4	B5	Executive	Folio
Ledger	Legal	Letter	LetterSmall
Quarto	Tabloid	Statement	10x17

Each of these options may be further *qualified* by an extension to indicate a slightly distinct treatment of the paper size. The only currently defined qualifier is **Transverse**, which indicates that the paper is fed in an orientation that is rotated 90 degrees from the orientation of the base paper size. The keyword qualifiers are appended with a period, like this: **Letter.Transverse**, **A5.Transverse**, etc.

NOTE:
Feeding the paper transversely does not affect the relationship of user space to the physical page, but it does mean that the page is oriented differently with respect to device space. This will probably only make a difference with asymmetric halftone screens (or patterns) and (in older printers) the speed of printing when using the image operator.

Any of the paper keywords may have a *serialization* extension which is used to distinguish between two otherwise equivalent instances of the same keyword. For example, if there are two Letter-size paper trays, they may simply be numbered to differentiate them (as in **Letter.1**, **Letter.2**). These extensions may be combined with the **Transverse** or other qualifiers, as appropriate.

4.5 PAPER SIZE INVOCATION

*DefaultPageSize: *PaperKeyword*

This is a keyword indicating the default page size for the printer when powered on. It will correspond to one of the possible pages sizes listed in the ***PageSize** section of the Printer Description file. See the discussion under ***PageSize** and the list of option keywords at the end of this document. If the default page size is the desired one, it is best not to set it explicitly.

*PageSize *PaperKeyword*: *"invocation"*

Provides information on available page sizes and their corresponding invocation strings. Although there may be several ways to set a particular paper type, the invocation in this section will always provide the most “reasonable” or “standard” approach to providing the right paper size. The invocation may in fact be *identical* to the value of some other paper keyword, but ***PageSize** is intended to be used in all but very specific circumstances (like when using **manualfeed** or when direct control over the paper handling is necessary).

*?PageSize: *"query"*

Provides a PostScript language sequence which, when sent to the printer, will return the appropriate *keyword* corresponding to the current page size. If the current page size cannot be determined, or is not one of the existing keywords, the word **Unknown** should be returned.

*DefaultPageRegion: *PaperKeyword*

This is a keyword indicating the default page size for the printer when powered on. It will correspond to one of the possible pages sizes listed in the ***PageRegion** section of the Printer Description file.

*PageRegion *PaperKeyword*: *"invocation"*

This is intended to set the *imageable area* to the appropriate paper type without explicitly specifying the source of the paper (as with specifying the input bin or paper tray). It is intended to be used in conjunction with **manualfeed** so that the imageable area is appropriate for the paper to be fed.

*DefaultPaperTray: *PaperKeyword*

Provides a keyword indicating the default paper tray, although this is to some extent meaningless, since the printer is undoubtedly shipped with the paper trays in separate pieces of styrofoam, and any of them may be inserted by the erstwhile system administrator. Proceed with caution if making assumptions about the default paper tray. It is likely to be **None** in most Printer Description Files.

*PaperTray *PaperKeyword*: *"invocation"*

Provides information on selectable input paper tray types, and their corresponding invocation strings. See discussion in the introductory text on paper handling for more information.

*?PaperTray: *"query"*

Provides a query to determine the currently set paper tray. *Note:* it is not always possible for the PostScript interpreter to determine the current paper tray, depending on the physical interface to the printer. If it is not possible to determine the current tray, this query entry will not be present. The query should return a keyword matching one of the valid ***PaperTray** keywords, else the word **Unknown**.

4.6 INFORMATION ABOUT PAPER SIZES

*DefaultImageableArea: *PaperKeyword*

This provides the keyword for the default imageable area. The value should always be **Letter**.

*ImageableArea *PaperKeyword*: *"int int int int"*

This provides the *bounding box* of the imageable area for each given page size. It is represented as four integers representing the X and Y coordinates of the *lower left* and *upper right* corners of the region, respectively. It takes into account any offset from the edges of the physical paper. The imageable region is defined to be the part of the page where marks can actually be made. On many printers, there are “margins” imposed by the paper transport mechanism in the marking engine that may prevent marks from being made close to the edges of the paper. The ***ImageableArea** will supply a region that represents a “reliable” area of the page in which marks can be made. This may or may not exactly correspond to the *clipping path* set by the PostScript interpreter. See also the ***PaperDimension** keyword section.

*?ImageableArea: *"query"*

This provides a query to determine the current page region information. It returns four integers representing the bounding box of the imageable area. Since it is virtually impossible to determine hardware restrictions from software polling, this query will most likely return the default clipping region for the page size in effect. In general, it is better to use the values supplied in the ***ImageableArea** keyword section, since they may be adjusted by hand for particular hardware constraints.

*DefaultPaperDimension *PaperKeyword*

This provides the keyword for the default paper dimension. The value should always be **Letter**.

*PaperDimension *PaperKeyword: "integer integer"*

This provides absolute dimensions for a particular paper size, independent of the imageable area of the page. There are only two integers specified, which represent the *width* and *height* of the paper, respectively. See also the ***PageRegion** keyword.

4.7 PAPER HANDLING FEATURES

*VariablePaperSize: True/False

This has a value indicating whether the device supports infinitely variable paper sizes. This corresponds to the existence of the **setpageparams** operator in the particular PostScript language implementation. This may be used by a document manager to determine if it is feasible to specify an unusual imageable area by using the **setpageparams** operator. Most normal paper sizes will also be included in the Printer Description file under the other standard paper keywords.

*DefaultInputSlot: Upper/Middle/Lower/None

This is the default value for the ***InputSlot** category.

*InputSlot Upper/Middle/Lower: *"invocation"*

This provides information on selectable input slots for paper trays. If present, this keyword implies that paper can be selected by specifying, for instance, the upper or the lower slot, and accepting whatever is found there.

*?InputSlot: *"query"*

This query will return the currently chosen input slot (one of **Upper**, **Middle**, **Lower**, or **None**).

*DefaultManualFeed: True/False/None

This is the default condition of the manualfeed operation. See ***ManualFeed** also. This can be used to determine the initial setting of the manualfeed condition. If the default state is the desired state, it should not be explicitly changed. This entry may be **None**.

*ManualFeed True/False/None: *"invocation"*

This provides a mechanism to turn manual feed on and off. Manual feed is interesting in that it involves setting a variable, rather than invoking a procedure. The state is in effect until the variable is reset to **false**. This feature may not be available on all printers, which will be indicated by the ***DefaultManualFeed** keyword having **None** as its value, or by the omission of the **ManualFeed** keywords altogether. If ***DefaultManualFeed** is **None**, this section should be ignored.

*?ManualFeed: *"query"*

This provides a query that will return the current state of the **manualfeed** condition of the printer. It will return **True**, **False**, or **None**.

*DefaultOutputBin: Upper/Middle/Lower/OnlyOne

This provides the Printer Description keyword for the default output bin (see above).

*OutputBin Upper/Middle/Lower/OnlyOne: *"invocation"*

This provides information on selectable output paths for paper. This entry may be absent form the Printer Description file, but if present it will provide invocation strings for selecting different output bins. Typically the only difference between output bins is whether the pages land face up or face down, but there may be other differences (such as paper capacity) which are harder to clearly define. They will be identified by keywords like **Upper** or **Lower**. These entries carry no meaning as to what will happen if you select the upper or lower output bin, but are provided so that they can be selected if desired. To invoke a particular *stacking order* for the pages, use the values provided under the ***OutputOrder** entry in the Printer Description file.

*?OutputBin: *"query"*

This provides a *query* string which will return the *Printer Description File keyword* corresponding to the current setting of the printer's output bin.

*DefaultOutputOrder: Normal/Reverse

Provides the keyword for the default output order.

*OutputOrder Normal/Reverse: *"invocation"*

This provides information about how to invoke a specific page stacking order (usually by setting a particular output bin on a printer). The keyword **Normal** means that the pages will end up in *1-n* order if they are sent to the printer in that order (this usually means that each page will land *face down* in the output tray). **Reverse** means that the pages will end up in *n-1* order if they are sent in *1-n* order (which might be thought of as backwards). This usually means the pages land *face up* in the output tray. This keyword section may be extended to address “signature” and other page ordering schemes, as appropriate.

*?OutputOrder: *"query"*

This query will return the current output order of the PostScript engine. It should return one of the valid keywords for the ***OutputOrder** main keyword: **Normal** or **Reverse**.

*DefaultCollator: *Keyword*

This provides a keyword to indicate the default state of the collator when the printer is turned on.

*Collator *Keyword: "invocation"*

If the printer is equipped with a collator, this will provide invocation information. The exact values of the keywords will depend on the eventual implementation of these features.

*?Collator: *"query"*

This will provide a query to determine the current state of the collator mechanism. It will return one of the defined keywords.

*DefaultDuplex: *Keyword*

Default condition of duplex mechanism when printer is powered on.

*Duplex *Keyword: "invocation"*

If duplex capability is supported by the printer, the invocation information will be contained here. The exact values of the keywords will depend on the eventual implementation of these features.

*?Duplex: *"query"*

This will provide a query as to the current state of the duplex mechanism. It will return one of the defined keywords.

4.8 FONT RELATED

*Font *FontName*: Standard/Special/ISOLatin1 *"(version)"*

One line for each ROM-resident font. ***FontName*** is the correct PostScript language name of the font. The next field is a keyword indicating the type of encoding (**Standard** or **Special**, or **ISOLatin**1, for instance), and the last field is the version number of the font (as found under the key **version** in the **FontInfo** dictionary that is a subdictionary of the particular font dictionary). There may, in the future, be additional information added to the Printer Description files to indicate character sets contained in the fonts.

*DefaultFont: *FontName*

This gives the font name which is the default font provided by **findfont** if the requested font is not available. Note that in some devices this may not be well-defined (especially where there might be a network font server, for instance), and this field may not be present. For many printers this field will contain the name **Courier**. If this value is **Error**, it means that an execution error will occur if the font is not found. Any other value implies that a font substitution will take place (as in substituting Courier).

*?FontList: *"query"*

Provides a PostScript language sequence to return a list of all available fonts. It should consult the FontDirectory dictionary as well as any mass storage devices available to the device. The list need be in no particular order, but each name should be returned separated by a slash **"/"** character. This is normally the way the PostScript == operator will return a font name. All white space characters should be ignored. The end of the font list should be indicated by a trailing **"*"** sign on a line by itself (decimal ASCII 42). Here is a look at two valid returns from the query:

```
/Optima/Optima-Bold/Optima-Oblique/Optima-BoldOblique/Courier/Symbol
*
```

```
/Courier
/Symbol
/Times-Roman
*
```

Note:
In previous versions of this document, it was recommended to use ***flush*** *to separate names into packets. This turns out to result in major performance degradation, and is hereby and subsequently disrecommended.*

*?FontQuery: *"query"*

This provides a PostScript language query that should be combined with a particular list of font names being sought. It looks for any number of names on the stack, and will print a list of values depending on whether or not the font is known to the PostScript interpreter. The font names should be provided on the operand stack by the Document Manager, This is done by simply emitting the names, with leading slash **"/"** characters, before emitting the query itself.

To keep the Document Manager from having to keep track of the precise order in which the values are returned, and to guard against errors from dropped information, the syntax of the returned value will be **/FontName:Yes** or **/FontName:No**, where each font in the list is returned in this manner. The slashes delimit the individually returned font names, although newlines should be expected (and ignored) between them: A final **'*'** character will follow the returned values.

```
/Times-Roman:Yes
/Optima:Yes
/CircleFont:No
/Adobe-Garamond:No
*
```

This query is in many cases preferable to the above ***?FontList** query, since that query may return a very long list of fonts in some devices (with mass storage available to the PostScript interpreter such as built-in hard disks or network font servers).

4.9 PRINTER MESSAGES

It is often desirable to arm yourself with a collection of fairly common messages that may be produced by the printer spontaneously. These messages may be recognized by the Printing Manager and some more readable message displayed to the user. They are enumerated in the Printer Description file for this purpose.

*PrinterError: *"string"*

This provides a list of possible **PrinterError** messages, of the form:

```
%%[PrinterError: cover open]%%
%%[PrinterError: paper exit misfeed]%%
```

In this case, the string **"cover open"** would be the value associated with the ***PrinterError** keyword. The brackets, percent signs, and the word "PrinterError" are *not* included in the Printer Description file.

*Status: *"string"*

This lists the possible responses to a status query (accomplished typically by sending ^T (control-T, decimal ASCII 20) over a serial connection or by a special status packet if a network protocol is used (for instance, AppleTalk)). The status message may be composed of up to *three* parts. There is always at least the word **"status:** *message***"** with an appropriate status message (those messages are listed in this section of the Printer Description file). There may also be two other sections, listing the currently executing job name (**"job:** *name***"**) as defined by the variable **jobname** in **statusdict**, and a *source* field, like this: **"source:** *connection***"**. Here are some examples of the format itself

```
%%[status: warming up]%%
%%[status: busy; source: AppleTalk]%%
%%[job: userjob; status: waiting; source: serial25]%%
```

The entries in the Printer Description File will not have the brackets or the percent signs:

```
*Status: "warming up"
*Status: "busy"
*Status: "waiting"
```

*Source: *"string"*

This lists the possible sources for print jobs. These correspond to the **"source:"** field in the status message (as shown under the ***Status** section). Here are some example entries:

```
*Source: "serial25"
*Source: "serial9"
*Source: "AppleTalk"
*Source: "Centronics"
```

Unfortunately, The status message in which the source is found may contain other fields (as in the example under ***Status** above) depending on the values of **/jobname** in **statusdict** and whether or not there is an active job (in which case the **source** is listed). Just the strings for the **status** field are provided in this section. Parse carefully.

*Message: *"string"*

This provides a list of possible printer messages that do not fit into the above categories. The text of these messages may or may not contain the **"%%[ ]%%"** syntax; therefore, the strings in this section *will* contain the delimiters (if they exist) as well as the text of the message. Here are three examples:

```
*Message: "%%[exitserver: permanent state may be changed]%%"
*Message: "%%[Flushing: rest of job (to end-of-file) will be ignored]%%"
*Message: "\FontName\ not found, using Courier"
```

Notice the **\FontName** notation in the last example, with the backslashes. This is a special situation, in that the exact text of the message depends on which font was requested (with **findfont**) by the user program. This backslash notation is a meta-syntax that indicates that any arbitrary PostScript language name may be found as the beginning of that message (substituted for **\FontName**). Not all software packages necessarily can make use of this, but at least the messages can be recognized and dealt with using this notation.

4.10 COLOR SEPARATION KEYWORDS

The following keywords provide suggested values for manipulating the PostScript language halftone machinery to provide good color separations. Each separate process color should be printed with a different screen angle and perhaps different transfer functions or at various screen frequencies. The ***colorsepkey*** keywords are detailed in the next section, **Standard Option Values for Main Keywords**.

*DefaultColorSep: *colorsepkey*

This keyword provides the default color separation, in the form of a colorsepkey keyword. This is used in conjunction with the other entries listed below.

*ColorSepScreenFreq *colorsepkey*: *"PostScript code"*

This keyword provides the appropriate screen frequency for a color separation keyed to the given colorsepkey.

*ColorSepScreenAngle *colorsepkey*: *"code"*

This entry gives the halftone screen angle for the given color separation.

*ColorSepScreenProc *colorsepkey*: *"code"*

This provides the halftone spot function for the specified color separation.

*ColorSepTransfer *colorsepkey*: *"code"*

This entry provides the transfer function appropriate for the given color separation keyword.

*CustomCMYK *inkname*: "cyan magenta yellow black"

This entry provides the CMYK equivalents for a named custom color. These may be user-defined or names used in a commercial color matching system which may provide CMYK approximations for particular marking technologies. The keyword CustomCMYK is kept deliberately brief because there may be many hundred entries of this sort in a printer description file. For some printers, in fact, these entries may be put into a separate file (considered to be a “customization” of the standard file) and reference the original with the *Include: convention mentioned in the document PostScript Printer Description Files. The file naming convention for the separate color files may have a slightly different extension (.CPD instead of .PPD) to distinguish them.

*InkName *inkname/alias*

This keyword provides an alternative name for one of the inkname keywords used in the *CustomCMYK section. Its purpose is to provide slightly more human-readable versions of the keywords that may be presented in a user interface (the keywords themselves cannot contain spaces). Here is an example:

```
*InkName p305/COLORNAME 305
```

4.11 SPECIAL KEYWORDS

*End

This is used to close multiple-line format. The double quotes used in the **"PostScript language string"** notation are still used to delimit the strings, but as an extra measure, the ***End** keyword can be included after a multiple-line PostScript string. This keyword should generally be ignored by parsing software.

5. STANDARD OPTION VALUES FOR MAIN KEYWORDS

These “option values” can appear in several different places in an Printer Description file, but they should always refer to the same thing. These option keywords will always appear after a high-level keyword like ***PageSize**. For example, in the following Printer Description File entry, the word **Ledger** is an option keyword:

```
*PageSize Ledger: "statusdict begin ledgertray end"
```

Following is list of all the standard values for these keywords. Adobe Systems is keeping track of these keywords, and maintains a registry of acceptable values to be found in Printer Description files. This is to ensure that there will never be two instances of the same keyword in two different Printer Description files where the keywords have different semantics.

5.1 SPECIAL KEYWORDS

None

Indicates the *absence* of options for a particular category.

Unknown

This is returned from *queries* if the correct information can not be determined, or none of the valid keywords can be returned.

5.2 FOREIGN LANGUAGE KEYWORDS

These keywords are used as arguments to the ***LanguageVersion** entry in the Printer Description file. See discussion at the beginning of this document on the syntax of translation strings and option keywords in foreign languages. These keywords are all the *English* words for the foreign languages in question, for simplicity.

English	French	German	Italian
Russian	Spanish	Chinese	Japanese

others defined as encountered.

5.3 PAPER KEYWORDS

Letter

612 x 792 point paper size. This keyword is used wherever this standard paper type is referred to.

LetterSmall

This is a reduced-size imageable region based on the Letter paper size. Its imageable area is 553 x 731.5 points centered on an 8.5 x 11 inch page.

Tabloid

This is 792 x 1224 point paper (11 x 17 in.), oriented in “portrait” or “tabloid” mode (the Y axis is on the longer edge of the paper).

Ledger

This is 1224 x 792 point paper (17 x 11 in.), oriented in “landscape” mode (the Y axis is on the shorter edge of the paper).

Legal

This refers to 612 x 1008 point paper (11 x 14 in.), oriented in “portrait” mode.

Statement

This refers to 396 x 612 point paper (5.5 x 8.5 in.), oriented in “portrait” mode.

Executive

This refers to 540 x 720 point paper (7.5 x 10 in.), oriented in “portrait” mode.

A3

This refers to the ISO / JIS A3 paper size, which is 842 x 1190 points in “portrait” orientation.

A4

This is 595 x 842 point paper in “portrait” orientation.

A4Small

This is a reduced-size imageable area version of the A4 paper size. The imageable area is 7.47 x 10.85 inches centered on an A4 page.

A5

This is 420 x 595 point paper in “portrait” orientation.

B4

This corresponds to the JIS Standard B4 paper size, which is 729 x 1032 point paper in “portrait” orientation.

B5

This refers to 516 x 729 point paper in “portrait” orientation.

Envelope

This is a page size reserved for printing envelopes. The imageable region of this paper size will depend on the values of the variables **envelopeheight** and **envelopewidth** as defined in **statusdict**. It is normally in “landscape” orientation. This may be expanded later to encompass specific standard envelope sizes. This keyword parallels a particular existing implementation.

Folio

This refers to a 567 x 903.5 point imageable area in “portrait” orientation, centered on an 8.5 x 13 inch sheet (folio sheet).

Quarto

This is an imageable region of 567 x 744 points in “portrait” orientation centered on a 610 x 780 point sheet of paper (quarto sheet).

10x14

This refers to 720 x 1008 point paper (10 x 14 in.) in “portrait” orientation.

5.4 PAPER TRAY AND BIN KEYWORDS

Many of the keywords that apply for input paper tray selection are the same ones listed above under ***Paper Keywords***. There are a few specialized tray keywords that are listed below.

LargeCapacity

This is used to refer to a large capacity paper tray, such as an input paper tray which can hold more than one ream of paper.

LargeFormat

This refers to an input slot that can hold "large format" paper trays (ledger paper, for instance).

Lower

This is used for any option for which there is no other particular distinguishing feature other than it is lower than another just like it. For instance, if there are two paper input slots that are identical, but there are two of them and they need to be distinguished somehow.

Middle

See "lower"

Upper

See "middle" or "lower"

OnlyOne

This keyword is used where **Upper**, **Middle**, and **Lower** make little sense (that is, if there is only one output bin or paper tray anyway).

AnySmallFormat

This is used as an option for ***PaperTray** to indicate a paper tray which can hold *any* of the smaller format papers. This includes any paper size that is up to (and including) 11 inches on the longer side.

AnyLargeFormat

This option to ***PaperTray** allows selection of a “universal” paper tray which may contain any of the large format paper sizes (those with one dimension greater than 11 inches).

Normal

This keyword is used in the ***OutputOrder** section to refer to “normal” sorting (usually face down in the output tray). If the pages are transmitted to the printer in 1-*n* order, then they should be in 1-*n* order when they are picked up from the output tray. Compare with **Reverse**.

Reverse

This keyword indicates *reverse* sorting (usually face down in the output tray). Pages in the output tray will be in the opposite order from the order in which they were transmitted to the printer.

5.5 FONT KEYWORDS

Standard

This keyword is used with the ***Font** entry in an Printer Description file to indicate a font which uses the Adobe **StandardEncoding** vector.

Special

This indicates a font with nonstandard encoding (for instance, the *Sonata* font).

ISOLatin1

This is used to indicate a font with the ISO-Latin-1 standard encoding vector.

Error

This is used only with the ***DefaultFont** keyword, to indicate that you don’t get a default (substituted) font at all, you get an execution error. This is true on some printers from Digital Equipment Corporation, for example.

5.6 HALFTONE KEYWORDS

Null

This is provided to indicate a null procedure body (typically for the transfer function). A null procedure body is represented in the PostScript language like this: "**{}**".

Normalized

This indicates a transfer function which has been adjusted to correct for marking engine characteristics to come as close as possible to “true” perceived gray levels. It will provide good results for the *default* halftone screen at the 10% gray levels, and will provide at least reasonable results for gray levels at in-between levels.

Dot

This keyword represents a standard “dot”-shaped halftone screen function. This is the default shape for the halftone cell on many PostScript language implementations, and basically consists of small black circles that vary in size with the gray level. This keyword also encompasses more sophisticated functions which may also produce circular dots (for example, as found on higher-resolution devices), but which might slightly differ from the most basic dot screen.

Line

This keyword represents a “line screen” halftone function. Gray levels will be rendered by parallel lines which vary in thickness according to the gray level.

Ellipse

This keyword provides an “elliptical spot” screen, which is similar to a “dot screen” except that the dots are elliptical rather than circular.

Cross

This provides a “crosshatch screen” halftone function.

Mezzo

This provides a pseudorandom “mezzotint” screen function for the halftone mechanism.

5.7 COLOR KEYWORDS

Color separation keywords (the notation colorsepkey in the keyword listings) are designed to reflect particular combinations of separation characteristics. For example, a given separation typically is designed for a particular process color (the cyan separation, say), at a certain halftone screen frequency, for a particular resolution device. To this end, the color separation keywords are complex and modular, but they can be made more human-readable through use of the general translation string mechanism provided in the file format.

A colorsepkey consists of a name which can optionally have qualifiers (sub-components) separated by dots (‘.’, decimal ASCII 46). The key is typically a color name, and a typical set of qualifiers for a color separation might be screen frequency and resolution, in the following relationship: colorname.frequency.resolution. The idea is to be able to associate many different components of a color separation “package” by keyword. Since the color separation components are tuned for particular screen frequencies (and device resolutions), it is convenient to be able to recognize them this way. The keywords are arbitrary, but the structured qualifiers makes it possible for an application to separate the components if necessary, to allow a user to choose from several frequencies, optional resolutions, etc. Otherwise, these keywords behave like any other option keywords in printer description files. For printers where the resolution cannot be varied (most of them), the resolution qualifier will usually be omitted from the colorsepkey keyword.

Here are several examples, to help illustrate the format more clearly:

```
*ColorSepScreenAngle cyan.60lpi.1270dpi: "37"
*ColorSepScreenAngle magenta.60lpi.1270dpi: "45"
*ColorSepScreenAngle yellow.60lpi.1270dpi: "75"
*ColorSepScreenAngle black.60lpi.1270dpi: "0"
*ColorSepScreenFreq black.60lpi.1270dpi: "60"
*ColorSepScreenProc black.60lpi.1270dpi: "{ pop }"
*ColorSepTransfer black.60lpi.1270dpi: "{ 1 exch sub}"
```

```
*ColorSepScreenFreq cyan.90lpi.1270dpi: "90"
*ColorSepScreenFreq cyan.60lpi.600dpi: "60"
```

Again, these keywords are used only to identify a particular set of separation parameters, like other option keywords, but they are designed so that there is a visible substructure for software that might need to construct them or know something about the individual components.

5.8 INK NAMES

For the ***CustomCMYK** keyword, inks can be named arbitrarily. The idea is to associate any given named ink (whether it be from a commercial color matching system or a local custom color) with a set of process color values to approximate it:

```
*CustomCMYK HarvestGold: "0 .01 .9 .01"
```