ADOBE
Systems
Incorporated

March 12, 1987

EPSF Version 1.2

Encapsulated PostScript File Format

A file format for all imported PostScript Illustrations and Images

The following specifies the format required for import of Encapsulated PostScript (EPS) Files into an application. This specification suggests a standard for importing PostScript files in all environments, and contains specific information about both the Apple Macintosh and MS-DOS environments. This format conforms to Adobe Systems' PostScript Document Structuring Conventions, version 2.0.

Introduction

The rules that should be followed in creating importable PostScript files are a subset of the structuring conventions proposed by Adobe Systems Incorporated; refer to the PostScript Language Reference Manual, Appendix C, and PostScript Document Structuring Conventions, version 2.0, available from Adobe Systems. Files must also be "well-behaved" in their use of certain PostScript operators, manipulation of the graphics state, and manipulation of the PostScript stacks and global dictionaries. These conventions are designed to allow cooperatvie sharing of files between many systems using PostScript.

Fundamentally, an Encapsulated PostScript file is merely a standard PostScript file with a bitmap screen dump optionally included in the format. Their purpose is to be included into other document makeup systems as illustrations, and the screen representation is intended to aid in page composition only, and the bitmaps is normally discarded when printing is done, and the PostScript segment of the file is used instead. Typically any manipulation of the screen image that is performed by the user (such as scaling, translating, or rotation) should be tracked by the page layout application and an appropriate PostScript transformation should precede the encapsulated PostScript when sent to the printer.

Encapsulated PostScript File GUIDELINES

An EPS file should conform to version 2.0 of the PostScript Document Structuring Conventions. This does not explicitly require any of the structuring comments to be employed, but if used, they should be in accordance with that specification. Additionally, an EPS file is required to contain the %%BoundingBox comment, and is required to be "well-behaved" (see pages 3-4). An EPS file may optionally contain a bitmap image suitable for WYSIWYG screen display, as discussed herein.

Structure Comments

The structure of an EPS file is marked by PostScript comments, according to the PostScript Document Structuring Conventions. These are covered briefly here for reference. Structuring comment lines must begin with "%!" or "%%" and terminate with a newline (either return or linefeed) character. EPS file conventions require that a comment line be no longer than 256 bytes. A comment line may be continued by beginning the continuation line with "%%+". The EPS file should begin with a "header" of structuring comments, as specified in the PostScript Structuring Conventions.

Required Comments

The first comment in the header (and the first line in the file) should be the version comment:

%!PS-Adobe-2.0 EPSF-1.2
This indicates to an application that the PostScript file conforms to this standard. The version number following the word "Adobe-" indicates the level of adherence to the standard PostScript Document Structuring Conventions. The version number following the word "EPSF" indicates the level of EPSF-specific comments.

The following comment must be present in the header; if it is not present then an importing application may issue an error message and abort the import:

%%BoundingBox: llx lly urx ury
The values are in the PostScript default user coordinate system, in points (1/72 of an inch, or 2.835 mm), with the origin at the lower left corner.

General Informational Comments

The following header comments are strongly recommended:

%%Creator: creator_name
%%Title: included_document_title
%%CreationDate: date_and_time
Creator, Title, and CreationDate information may be used by an application or spooler to provide human-readable information about a document, or to create a screen representation of the PostScript segment if no screen representation is included in the file.

%%EndComments
This comment indicates an explicit end to the header comments

Font Management Comments

If fonts are used, the following two comments (which are defined in version 2.0 of the PostScript Document Structuring Conventions) should be included in the header of the EPS file:

%%DocumentFonts: font1 font2 ....
%%+ font3 font4
The DocumentFonts comment is a full list of all fonts used by the file. Font names should refer to non-reencoded printer fonts and should be the valid PostScript names (without a slash) for the fonts. An application that imports an EPS file should be responsible for satisfying these font needs, or at least updating its own DocumentFonts list to reflect any new fonts.

%%DocumentNeededFonts: font1 font2 ....
The DocumentNeededFonts comment lists all fonts that are to be included at specific points within the EPS file as a result of the %%IncludeFont comment. These fonts must also be listed in the Document Fonts comment, but an application may or may not preload these at the beginning of the job. The responsibility should be taken, however, to make sure the fonts requested will be available.

Within the body of the PostScript file, an application or spooler should be prepared to handle the following comment:

%%IncludeFont: fontname
The IncludeFont comment signals to an application that the specified font is to be loaded at that point in the imported PostScript code. An application should load the specified font regardless of whether the same font has been loaded already by other preceding IncludeFont comments, since the font may be embedded within a PostScript save and restore construct. However, if the font is determined to be available prior to the entire included EPS file (for instance, it may be in ROM on the printer or might have been downloaded prior to the whole job) the IncludeFont comment may be ignored.

A font that is wholly contained, defined, and used within the EPS file (a downloaded font) should be noted in the DocumentFonts comment but not the DocumentNeededFonts comment. The font should follow conventions listed in the PostScript Document Structuring Conventions in order to retain full compatibility with print spoolers. In particular, it should be embedded with the %%BeginFont and %%EndFont comments.

File Inclusion Comments

%%IncludeFile: filename
This comment, which can occur only in the body of an EPS file, allows a separate file to be inserted at any point within the EPS file. The file might not be searched for or inserted until printing actually occurs, so user care is required to ensure its availability. If it is used, the %%DocumentFiles comment should be used as well. See the Structuring Conventions for more information.

"Well-Behaved" Rules

An application should encapsulate the imported EPS PostScript code in a save / restore construct, which will allow all printer VM (memory) to be recovered and all graphics state restored. Since the code in the imported EPS file will be embedded within the PostScript that an application will generate for the current page, it is necessary that it obey the following rules, in order to keep from disrupting the enclosing document:

Operators to Avoid
The following PostScript operators should not be included in a PostScript file for import; the result of executing any of these is not guaranteed (see the Structuring Conventions for more on this):

grestoreall	initgraphics
initmatrix	initclip

erasepage	copypage
banddevice	framedevice
nulldevice	renderbands
setpageparams	note
and especially not exitserver	

showpage:
The showpage operator is permitted in EPS files primarily because it will be present in so many PostScript files. It is reasonable for an EPS file to include a reference to showpage if needed (although it is not necessary if the file is truly imported into another document), it is the including applications responsibility to disable showpage if needed. The recommended method to accomplish this is as follows:

Temporarily Disabling Showpage:

```
/--save0-- save def      % save state
/showpage { } def
   include the EPS file here, which may
   execute showpage with no effect
--save0-- restore        % restore state
```

This method will only disable the showpage operator during the execution of the EPS file, and will restore the original semantics of showpage afterward. It is the responsibility of the EPS file itself to avoid the operators listed above which might cause unexpected behavior when imported.

PostScript Stacks:

All PostScript stacks (including the dictionary stack) should be left in the state that they were in before the imported PostScript code was executed.

Dictionaries:

No global strings should be changed.

It is recommended that the imported PostScript EPS file create its own dictionary instead of writing into whatever the current dictionary might be. Make sure that this dictionary is removed from the dictionary stack when through (using the PostScript end operator) to avoid the possibility of an invalidrestore error.

If a special dictionary is required in order for the imported PostScript code to execute properly, then it should be included as part of the PostScript file. However, it should be enclosed in comments as specified in the Structuring Conventions. No dictionary should be assumed to be present in the printer, and fonts should be reencoded as needed by the EPS file itself.

File Types and File Naming

Apple Macintosh files
The Macintosh file type for application-created PostScript files is EPSF. Files of type TEXT will also be allowed, so that users can create EPS files with standard

editors, although the Structuring Conventions must still be strictly followed. A file of type EPSF should contain a PICT resource in the resource fork of the file containing a screen representation of the PostScript code.

MS-DOS files
The recommended file extension is .EPS. Other file extensions will also be allowed, but it will be assumed that these files are text-only files with no screen metafile included in them.

Screen Representation

The EPS PostScript file will usually have a graphic screen representation so that it can be manipulated and displayed on a workstation's screen prior to printing. The user may position, scale, crop or rotate this screen representation, and the composing software should keep track of these manipulations and reflect them in the PostScript that is ultimately sent to the PostScript printing device.

Macintosh: PICT Representation
A QuickDraw representation of the PostScript file can be created and stored as a PICT in the resource fork of the file. It should be given resource number 256. If the PICT exists, an application may use it for screen display. If the picframe is transformed to PostScript coordinates, it should agree with the %%BoundingBox comment. Given the size limitations on PICT images, this may not always agree for large illustrations. If there is a discrepancy, the %%BoundingBox should be taken as the "truth".

MS-DOS: MetaFile or TIFF Representation
Either a Microsoft Windows MetaFile or a TIFF (Tag Image File Format) section can be included as the screen representation of an EPS file. The file format for EPS files is:

Header:	Bytes	Description
	0-3	Must be hex C5D0D3C6 (byte 0=C5)
	4-7	Byte position in file for start of PostScript code section.
	8-11	Byte length of PostScript section
	12-15	Byte position in file for start of Metafile screen representation.
	16-19	Byte length of Metafile section (PSize)
	20-23	Byte position of TIFF representation
	24-27	Byte length of TIFF section
	28-29	Checksum of header (XOR of bytes 0-27) NOTE: if FFFF then it is to be ignored.
Body:	Bytes	Description
	0-PSize-1	Metafile contents (according to Microsoft specification for Windows)

Note: It is assumed that either the MetaFile or the TIFF position and length fields are zero; that is, only one or the other of these two forms are included in the EPS file.

The MetaFile should follow the guidelines set forth by the Windows specification. In particular, it should not set the viewport or mapping mode, and it should set the window origin and extent. The application should scale the picture to fit within the %%BoundingBox comment specified in the PostScript file.