Encapsulated PostScript File Format Summary

(https://simpleshare.io/go?site=pinboard&ga=UA-328425-1&url=https%3A%2F%2Fwww.fileformat.info%2Fformat%2Feps%2Fegff.htm&text=Encapsulated%20PostScript%20%3A%20Summary%
(https://simpleshare.io/go?site=hn&ga=UA-328425-1&url=https%3A%2F%2Fwww.fileformat.info%2Fformat%2Feps%2Fegff.htm&text=Encapsulated%20PostScript%20%3A%20Summary%
(https://simpleshare.io/go?site=twitter&ga=UA-328425-1&url=https%3A%2F%2Fwww.fileformat.info%2Fformat%2Feps%2Fegff.htm&text=Encapsulated%20PostScript%20%3A%20Summary%
(https://simpleshare.io/go?site=facebook&ga=UA-328425-1&url=https%3A%2F%2Fwww.fileformat.info%2Fformat%2Feps%2Fegff.htm&text=Encapsulated%20PostScript%20%3A%20Summary%
(https://simpleshare.io/go?site=reddit&ga=UA-328425-1&url=https%3A%2F%2Fwww.fileformat.info%2Fformat%2Feps%2Fegff.htm&text=Encapsulated%20PostScript%20%3A%20Summary%

Also Known As: EPS, EPSF, EPSI

Type	Page Description Language (PDL); used as metafile (see article)
Colors	Mono
Compression	See article
Maximum Image Size	NA
Multiple Images Per File	No
Numerical Format	NA
Originator	Adobe
Platform	Macintosh, MS-DOS, MS Windows, UNIX, others
Supporting Applications	Too numerous to list
See Also	None

Usage
Illustration and DTP applications, bitmap and vector data interchange.

Comments
A file format with wide support associated with the PostScript PDL. Although complex, internal language features are well-documented in Adobe's excellent PostScript publications and elsewhere. Many applications, however, write but do not read EPSF-format files, preferring to avoid supporting PostScript rendering to the screen.

Vendor specifications (spec/index.htm) are available for this format.

Code fragments (code/index.htm) are available for this format.

Sample images (sample/index.htm) are available for this format.

Data in an Encapsulated PostScript (EPSF) file is encoded in a subset of the PostScript Page Description Language (PDL) and then "encapsulated" in the EPS standard format for portable interchange between applications and platforms. An EPSF file is also a special PostScript file that may be included in a larger PostScript language document.

Contents:

EPSF files are commonly used to contain the graphics and image portions of a document. The main body of the document is defined in one or more PostScript files, but each piece of line art or photographic illustration embedded in the document is stored in a separate EPSF file. This scheme offers several advantages, including the ability to alter illustrations in a document without having to edit the document file itself.

EPSF also provides the ability to store image data in a 7-bit ASCII form, which is occasionally more convenient than the 8-bit binary format used by most bitmap formats.

Although we choose not to discuss PostScript itself in detail, because it is described so extensively elsewhere, we must look briefly at it in order to understand EPSF, which implements a subset of the language.

PostScript was created in 1985 by Adobe Systems and is most often described as a PDL. It is used mainly as a way to describe the layout of text, vector graphics, and bitmap images on a printed or displayed page. Text, color, black-and-white graphics, or photographic-quality images obtained from scanners or video sources can all be described using PostScript. PostScript, however, is a versatile general-purpose computer language, similar in some respects to Forth.

Partly because it is a language, PostScript is device-independent and provides portability and consistent rendering of images across a wide range of platforms. It also implements a de facto industry-standard imaging model for communicating graphics information between applications and hardware devices, such as between word processors and printers. In addition to general-purpose language features, however, PostScript includes commands used for drawing.

A PostScript output device typically contains an interpreter designed to execute PostScript programs. An application sends a stream of PostScript commands (or copies a file to) the device, which then renders the image by interpreting the commands. In the case of printers, typesetters, imagesetters, and film recorders, their main function is to interpret a stream of PostScript language code and render the encoded graphical data onto a permanent medium, such as paper or photographic film. PostScript is capable of handling monochrome, gray-scale, or color images at resolutions ranging from 75 to over 3000 DPI.

PostScript files are written in 7-bit ASCII and can be created using a conventional text editor. Although PostScript can be written by hand, the bulk of the PostScript code produced today comes from applications, which include illustration packages, word processors, and desktop publishing programs. Files produced in this manner are often quite large; it is not unusual to see files in the range of several megabytes.

The PostScript specification is constantly evolving, and two fairly recent developments include Display PostScript and PostScript Level 2. Display PostScript is used for on-screen imaging and is binary-encoded. It is used to drive window-oriented PostScript interpreters for the display of text, graphics, and images. PostScript Level 2 adds additional features to the PostScript Level 1 language, including data compression (including JPEG), device-independent color, improved halftoning and color separation, facsimile compatibility, forms and patterns caching, improved printer support, automatic stroke adjustment, step and repeat capability, and higher operational speeds. PostScript Level 2 code is completely compatible with PostScript Level 1 devices.

File Organization

As we mentioned at the beginning of this section, the EPS format is designed to encapsulate PostScript language code in a portable manner. To accomplish this, an EPS file normally contains nothing but 7-bit ASCII characters, except for the Display EPS format discussed later. An example of a small EPS file is shown in the section called "EPS Files."

File Details

This section contains information about, and examples of, EPS, EPSI, and EPSF files.

EPS Files

An EPS file contains a PostScript header, which is a series of program comments, and may appear as:

```
%!PS-Adobe-3.0 EPSF-3.0
%%Title: Figure 1-1, Page 34
%%Creator: The Image Encapsulator
%%CreationDate: 12/03/91 13:48:04
%%BoundingBox:126 259 486 534
%%EndComments
```

Any line beginning with a percent sign (%) in a PostScript file is a comment. Normally, comments are ignored by PostScript interpreters, but comments in the header have special meanings. Encapsulated PostScript files contain two comments that identify the EPS format. The EPS identification comment appears as the first line of the file:

```
%!PS-Adobe-3.0 EPSF-3.0
```

The version number following the "PS-Adobe-" string indicates the level of conformance to the PostScript Document Structuring Conventions. This number will typically be either 2.0 or 3.0. The version number following the "EPSF-" indicates the level of conformance to the Encapsulated PostScript Files Specification and is typically 1.2, 2.0, or 3.0.

The next EPS-specific comment line is:

```
%%BoundingBox:
```

followed by four numeric values, which describe the size and the position of the image on the printed page. The origin point is at the lower-left corner of the page, so a 640x480 image starting at coordinates 0,0 would contain the following comment:

```
%%BoundingBox: 0 0 640 480
```

Both the %%PS-Adobe- and the %%BoundingBox: lines must appear in every EPS file. Ordinary PostScript files may formally be changed into EPS files by adding these two lines to the PostScript header. This, however, is a kludge and does not always work, especially if certain operators are present in the PostScript code (such as initgraphics, initmatrix, initclip, setpageparams, framedevice, copypage, erasepage, and so forth) which are not part of the EPSF subset.

The other comment lines in the header, %%Title:, %%Creator:, and %%CreationDate: are used to identify the name, creator, and creation date of the EPS file. The header is always terminated by the %%EndComments comment. Other comments may also appear in the header, such as %%IncludeFile, %%IncludeFont, and %%Page.

Encapsulated PostScript (EPS) Example

The following shows an example of an EPS file:

```
%!PS-Adobe-2.0 EPSF-1.2
%%Creator: O'Reilly 2.1
%%CreationDate: 12/12/91 14:12:40
%%BoundingBox:126 142 486 651
%%EndComments
/ld {load def} bind def
/s /stroke ld /f /fill ld /m /moveto ld /l /lineto ld /c /curveto ld /rgb
{255 div 3 1 roll 255 div 3 1 roll 255 div 3 1 roll setrgbcolor} def
126 142 translate
360.0000 508.8000 scale
/picstr 19 string def
152 212 1[152 0 0 -212 0 212]{currentfile picstr readhexstring pop}image
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFF8000EE614FFFFFFFFFFFFFFFFFFFFFFFFF
FFFF00000000000000000000000000FFFFFFFF
FFFF000000000000000000000000000000FFFF
FFFF000000000000000000000000000800FFFF
FFFF80000000435C7FFFFFFF600001FFFFFFFF
FFFF0000000000000FFFFFFF700047FFFFFFFF
FFFF000000000000000000000000000001FFFF
Thousands of lines deleted to save space in this book.
FFFF000000000000000000000000000000FFFF
FFFF80000000001DF7E61000000001FFFFFFFF
FFFF00000000391FFFFFFFFFE70003FFFFFFFF
FFFF00000000000000000000000001FFFFFFFF
FFFF000000000000000000000000000000FFFF
FFFF80000000000000000000000000837DFFFF
FFFF8000000067DFFCF0000000001FFFFFFFFF
FFFF00000000000000000000000000FFFFFFFF
FFFF000000000000000000000000000001FFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
showpage
^D
```

Looking at the EPS example, we can see the comments header at the start of the file. Following the header there is a short block of PostScript code, which does the actual drawing, scaling, cropping, rotating, and so on of the image. This block is sometimes all that needs to be changed in order to alter the appearance of the image.

Following the PostScript code block is bitmap data, which in an EPS file is called a graphics screen representation. This consists of hexadecimal digits. Each byte of image data contains eight pixels of information and is represented by two bytes of hexadecimal data in the bitmap. Image line widths are always a multiple of eight bits and are padded when necessary. Only black-and-white, 1-bit per pixel images are currently supported by EPS.

At the end of the EPS file is the showpage operator, which normally indicates a PostScript output device on which to print or display the completed image or page. In EPS files embedded in other documents, however, showpage is not really needed. Sometimes an EPS file fails to display or import properly because the PostScript interpreter reading the file does not expect to encounter a showpage and becomes confused. You can solve the problem either by disabling the showpage operator in the interpreter or by removing the showpage operator from the EPS files.

If you look at a PostScript file in a editor, you may find that the very last character in the file is a **CTRL-D** (ASCII 04h) character. This control code has a special meaning to a PostScript device; it is an End-Of-Job marker and signals that the PostScript code stream has ended. When a PostScript device reads this character, it may perform an end-of-file terminate-and-reset operation in preparation for the next PostScript data stream. The presence of this character can sometimes be a source of problems to applications that are not expecting or equipped to handle a non-printable ASCII character in the data stream. On the other hand, problems can occur in PostScript output devices if a file does not have this character at the end of the code stream. And if spurious **CTRL-D** characters appear in the data stream, not much will come out of your printer at all.

EPS files (as opposed to normal PostScript) are generally created exclusively by code generators and not by hand. Each line is a maximum of 255 characters wide and is terminated with a carriage return (ASCII 0Dh) on the Macintosh, a linefeed (ASCII 0Ah) under UNIX, and a newline (ASCII 0Dh/0Ah) under MS-DOS. A PostScript interpreter should be able to recognize files using any of these line-termination conventions.

EPS files may also be in preview format. In this format, an actual image file is appended to the end of the file. This provides a quick method of viewing the image contents of the EPS file without having to actually translate the PostScript code. This is handy for applications that cannot handle PostScript interpretation, but which can display bitmap graphics and wish to import EPS files. Previews are typically scaled down (but not cropped), lower-resolution versions of the image. EPS previews are similar to postage stamp images found in the Truevision TGA and Lumena bitmap formats.

Four file formats may be used as EPS preview images: TIFF, Microsoft Windows Metafile (WMF), Macintosh PICT, and EPSI (Encapsulated PostScript Interchange format). In the Macintosh environment, an EPS file is stored only in the file data fork. A PICT preview is stored in the resource fork of the EPS file and will have a resource number of 256. PostScript Level 2 supports JPEG-compressed images as well. TIFF and WMF files are appended to EPS files in their entirety. Because MS-DOS files do not have a resource fork or similar mechanism, a binary header must be prepended to the EPS file containing information about the appended image file.

EPS Preview Header

The EPS preview header is 32 bytes in length and has the following format:

```
typedef struct EPSHeader
{
  BYTE Id[4];                   /* Magic Number (always C5D0D3C6h) */
  DWORD PostScriptOffset;       /* Offset of PostScript code */
  DWORD PostScriptLength;       /* Size of PostScrip code */
  DWORD WMFOffset;              /* Offset of WIndows Metafile */
  DWORD WMFSize;                /* Size of Windows Metafile */
  DWORD TIFOffset;              /* Offset of TIFF file */
  DWORD TIFSize;                /* Size of TIFF file */
  DWORD CheckSum;               /* Checksum of previous header fields */
} EPSHEADER;
```

Id, in the first four bytes of the header, contains an identification value. To detect whether an MS-DOS EPS file has a preview section, read the first four bytes of the file. If they are the values C5h D0h D3h C6h, the EPS file has a preview section. Otherwise, the first two bytes of an EPS file will always be 25h 21h (%!).

PostScriptOffset and PostScriptLength point to the start of the PostScript language section of the EPS file. The PostScript code begins immediately after the header, so its offset is always 32.

WMFOffset and WMFSize point to the location of the WMF data if a WMF file is appended for preview; otherwise, these fields are zero.

The same is true for TIFOffset and TIFSize. Because either a TIFF or a WMF file (but not both) can be appended, at least one, and possibly both, sets of the fields will always be zero. If the checksum field is set to zero, ignore it. Offsets are always measured from the beginning of the file.

The three preview formats detailed are only somewhat portable and are fairly device dependent; not all environments can make use of the TIFF and WMF image file formats and fewer still of PICT. For one thing, the addition of 8-bit binary data to the EPS file prevents the file from being transmitted via a 7-bit data path without special encoding. The EPSI format, described in the next section, however, is designed as a completely device-independent method of image data interchange.

EPSI Files

EPSI bitmap data is the same for all systems. Its device independence makes its use desirable in certain situations where it is inconvenient to store preview data in 8-bit TIFF, WMF, or PICT format. EPSI image data is encoded entirely in printable 7-bit ASCII characters and requires no uncompression or decoding.

EPSI is similar to EPS bitmap data except that each line of the image begins with a comment % token. In fact, nearly every line in the EPSI preview is a comment; this is to keep a PostScript interpreter from reading the EPSI data as if it were part of the EPS data stored in the file.

Typically, an application will support one or more of the device-dependent preview formats. An application should also support the EPSI format for the export of EPS data to environments that cannot use one of the other preview formats.

The following shows an example of an EPSI file:

```
%%Title: EPSI Sample Image
%%Creator: James D. Murray
%%CreationDate: 12/03/91 13:56:24
%%Pages: 0
%%BoundingBox: 0 0 80 24
%%EndComments
%%BeginPreview: 80 24 1 24
% C0FFFFFFFFFFFFFFFFFF
% 0007F1E18F80FFFFFFFF
% 0007F1C000000001FFFF
% 001FFFE78C01FFFFFFFF
% E7FFFFFFFFFFFFFFFFFF
% 000FFFFF81FFFFFFFFFF
% 0007F38000000001FFFF
% 000FFFCE00000001FFFF
% E1FFFFFFFFC7FFFFFFFF
% 00FFFFFFFFEFFFFFFFFF
% 0003FFCF038008EFFFFF
% 0003FFCF00000000FFFF
% 007FFFFFFFCFFFFFFFFF
% 40FFFFFFFFFFFFFFFFFF
% 0003FFFFFFFBFFFFFFFF
% 00003FFFFFE00001FFFF
% 0001FFFFFFF00031FFFF
% 00FFFFFFFFFFFFFFFFFF
% 0003FFFFFFFFFFFFFFFF
% 00000E000FF80073FFFF
% 00000E000FF80001FFFF
% 0003FFCFFFFFFFFFFFFF
% 0007FFFFFFFFFFFFFFFF
% 000003800071FFFFFFFF
%%EndPreview
%%EndProlog
%%Page: "one: 1
4 4 moveto 72 0 rlineto 0 16 rlineto -72 0 rlineto closepath
8 setlinewidth stroke
%%Trailer
```

EPSF Files

A question that is frequently asked is "What is the difference between an EPS file and an EPSF file?" The answer is that there is no difference; they are the same format. The actual designation, EPSF, is often shortened to EPS, which is also the file extension used for EPSF files on operating systems such as MS-DOS and UNIX.

EPSF files come in two flavors. The first is a plain EPSF file that contains only PostScript code. The second is a Display or Preview EPSF file that has a TIFF, WMF, PICT, or EPSI image file appended to it. Under MS-DOS, Encapsulated PostScript files have the extension .EPS, and Encapsulated PostScript Interchange files have the extension .EPI. On the Macintosh, all PostScript files have the file type EPSF. Also on the Macintosh, a file type of TEXT is allowed for PostScript files created in an editor. Such files should have the extension .EPSF or .EPSI. All other systems should use the filename extensions .EPSF and .EPSI.

For Further Information

PostScript was created and is maintained by Adobe Systems Inc. For specific information about PostScript, contact:

Adobe Systems Inc.
Attn: Adobe Systems Developer Support
1585 Charleston Rd.
P.O. Box 7900
Mountain View, CA 94039-7900
Voice: 415-961-4400
Voice: 800-344-8335
FAX: 415-961-3769
WWW: *http://www.adobe.com/*
FTP: *ftp://ftp.adobe.com/*

Adobe Systems distributes and supports a PostScript Language Software Development Kit to help software developers create applications that use PostScript. The kit includes technical information on PostScript Level 1 and Level 2 compatibility, optimal use of output devices, font software, and file format specifications. Also included are sample fonts, source code, and many PostScript utilities.

There are also numerous books written on PostScript. The fundamental reference set consists of the blue, green, and red books available in bookstores or directly from Adobe Systems or Addison-Wesley:

Adobe Systems, *PostScript Language Tutorial and Cookbook*, Addison-Wesley, Reading, MA, 1985.
Adobe Systems, *PostScript Language Program Design*, Addison-Wesley, Reading, MA, 1988.
Adobe Systems, *PostScript Language Reference Manual*, Second Edition, Addison-Wesley, Reading, MA, 1990.

Other excellent and readily available books about PostScript include:

Braswell, Frank Merritt, *Inside PostScript*, Peachpit Press, Atlanta, GA, 1989.
Glover, Gary, *Running PostScript from MS-DOS*, Wincrest Books, 1989. Reid, Glenn C., *Thinking in PostScript*. Addison-Wesley, Reading, MA, 1990.
Roth, Stephen F., *Real World PostScript*, Addison-Wesley, Reading, MA, 1988.
Smith, Ross, *Learning PostScript, A Visual Approach*, Peachpit Press, 1990.

A very helpful PostScript resource also exists in the form of a gentleman named Don Lancaster. Mr. Lancaster is the author of more than two dozen books on PostScript, laser printers, and desktop publishing. His articles can be read regularly in *Computer Shopper* magazine. His company, Synergetics, offers many PostScript products and information as well as a free technical support hotline for PostScript users. Contact:

Synergetics
Attn: Don Lancaster
Box 809-PCT
Thatcher, AZ 85552
Voice: 602-428-4073

More Resources (internal.htm)