Printing PostScript Files from Microsoft Windows

Gene Michael Stover

created Tuesday, 9 November 2004
updated Tuesday, 9 November 2004

Contents

1 Introduction

Have you ever had a PostScript file on disk, a PostScript printer on the other side of the room, Microsloth Winders in between, & therefore no easy way to print that PostScript file? Sure, it's possible to print a PostScript file on Winders,

but it's not as easy as it should be. How easy should it be? Well, a PostScript printer speaks PostScript natively, so you should be able to send that PostScript file to that PostScript printer almost verbatim & see your output. But without special software, Winders doesn't let you do that.

Installing that special software[1] isn't too difficult, but what if you are at the public library, or Kinko's, where you don't have permission to install anything? What if you are at your mother's house, where no one has ever printed a PostScript file, & you are in a hurry? I mean that, sure, you can print a PostScript file from Microsloth Winders, but it's not as easy as it should be, & it should be almost as easy as copying a file because all you need to do is copy the PostScript file to your PostScript printer.

So I wrote a program to make it that easy. I kept it so simple, so small, that it's virtually guarranteed to work on whatever Winders system you have, & it's so quick to try that installation time & priviledges are not a problem. I hope.

2 License

The source code & executalbe file described in this article & in the accompanying files are licensed according to the terms of the Gnu General Public License ([1,)] (GPL).

The documentation (the document you are reading now) is copyrighted by Gene Michael Stover. The copyright notice is at the beginning of the document. The documentation is not licensed according to the GPL.

3 Obtaining

The executable file is http://lisp-p.org/psw/print-postscript.exe. You may download that file to your computer, & it'll be ready to use. I explain how to use it in Section 5.

The source code is in two files:

- http://lisp-p.org/psw/print-postscript.c, &
- http://lisp-p.org/psw/Makefile

To make use of the source code, you will need a C compiler, a `make`[2], & a knowledge of C programming.

4 Installation

If you want to compile it yourself, you will need to modify the Makefile. You'll want to ensure that the `COMPILE_c` & `LINK_c` macros use your compiler & linker.

[1] Gnu Ghostscript is an example of that special software.

[2] Microsloth's `nmake` will do.

You'll want to ensure that the paths to the include (`*.h`) files & other paths are appropriate for your system.

Then run `make` (or `nmake` if you are using Microsloth's version of make).

You'll have one result file, `print-postscript.exe`.

If you downloaded `print-postscript.exe` or built it yourself, copy it to a directory in your `path` or another directory from which you want to run it.

5 Usage

5.1 Synopsis

`print-postscript` --help
`print-postscript` -?
`print-postscript` -h
`print-postscript` /?
`print-postscript` /h
`print-postscript` /help
`print-postscript` [-f *pathname*] *printername*
`print-postscript` -l

5.2 Description

`print-postscript` is a command line program. In other words, to use it, you must first make a command line window.[3] You must run `print-postscript` by typing the program's name, followed by the appropriate command line options & arguments, into the command line window, then pressing Return[4].

The first six forms cause `print-postscript` to print a short usage message.

For the seventh for, `print-postscript` sends a PostScript file to a PostScript printer. The *pathname* argument, if present, is the name of the PostScript file; if it is absent, `print-postscript` reads the PostScript file from standard input. The *printername* argument is the name of the printer. It is critical to get the proper name of the printer or *print-postscript* won't work.

The eight form, with its `-l` option, exists to help determine the name of the printer.

5.3 Options

--help causes `print-postscript` to print a usage message

-? causes `print-postscript` to print a usage message

-h causes `print-postscript` to print a usage message

[3] A command line window is also called a "DOS box" or a "cmd window".

[4] On some keyboards, there is a Return key. On other keyboards, there is an Enter key. On still other keyboards, the equivalent key contains a bent arrow; from its start, it moves down, then turns to your left. It ends with the arrow head pointing left.

/? causes `print-postscript` to print a usage message

/h causes `print-postscript` to print a usage message

/help causes `print-postscript` to print a usage message

-f pathname specifies the name of the PostScript file to print. If it's absent, `print-postscript` reads the PostScript data from standard input.

-l causes `print-postscript` to print a list of local printers & recently used networked printers.

5.4 Arguments

printername is the true, full name of your printer. This isn't always obvious, so I give hints for determining it in Section 5.5.

5.5 Determining the Printer's Name

Determining the printer's true, full name isn't always easy.

You can use the `-l` command line option to make `print-postscript` print a list of printers.

For locally connected printers, such as a printer connected to the LPT or USB ports, the `-l` should always work.

For networked printers, you might need to print a short document to the printer before trying "`print-postscript -l`". This is because the flags that `print-postscript` uses to get a list of printers seems to list local printers & *recently used* remote printers. I haven't figured out what flags print all remotely connected printers, not just recently used ones.

I apologize for how `print-postscript.exe` is so picky about the printer's name, but if you keep at it & get the right printer name, it should work fine. (Well, it has for me so far.)

5.6 Examples

Let's say that `print-postscript.exe` is installed in the `C:\bin` directory, the file I want to print is `C:\My documents\report.ps`, & the printer is `zippo` shared from `zardos`.

To print the aforementioned `report.ps`, I would create a command line window. In it, I could type "`C:\bin\print-postscript C:\My documents\report.ps \\zardos\zippo`".

I could type a shorter command if I take advantage of the current directory, like this:

```
C:\> cd "My documents"

C:\My documents> C:\bin\print-postscript -f report.ps \zardos\zippo
```

If `print-postscript.exe` is in the same directory as `report.ps`, my job is even easier. After I make the command line window, I could do this:

```
C:\> cd "My documents"

C:\My documents> .\print-postscript -f report.ps \\zardos\zippo
```

6 How It Works (for Programmers)

The source code is in http://lisp-p.org/psw/print-postscript.c.

If you look at it, one of the first things you'll notice is that I make use of `static` functions. Most people hate that. Sorry, but I've been programming C since 1985, & I prefer to use `static` functions where `static` functions do.

Then you will notice that the names of my `static` functions begin with `S_`. I make most function names begin with the name of the module, followed by an underbar. For static functions, I use `S` as a pseudo-module name. Again, most people hate it, but I've been doing this since 1985, & this is the style I prefer.

Yet another convention I use is that most functions return zero on success, non-zero on exit. This is the convention of the unix API.

The program's entry point is, of course, `main`. It uses `S_CommandLine` to parse the command line, then `S_Run` to do everything else.

`S_CommandLine` is as uninteresting as `main`.

`S_Run` opens the PostScript file for input. We use the Standard C I/O functions on the PostScript file because they are convenient, but we must use Winders-specific functions for the printer. We send the PostScript data to the printer as "raw" data, so we must open the printer with `OpenPrinter`, tell Winders that we're printing a document with `StartDocPrinter`, & then ensure that the printer is in PostScript mode. All that printer preparation happens in `S_StartPrinter`.

After that, it's just a matter of copying the PostScript data from the input file to the printer. That happens in `S_Loop`.

When we're done with the copying operation, we tell Winders that we're done printing with `EndDocPrinter`. Then we close the printer handle.

7 Enhancements

I guess `print-postscript` could check a printer's capabilities. If the printer understood PostScript, then `print-postscript` could tell Windows that it's a PostScript job we'll be printing. That might allow us to print PostScript files to printers which do not understand PostScript in their firmware but for which there is a PostScript driver installed.

If the printer didn't understand PostScript, we'd want to send the PostScript data to the printer as "raw", which is what we do now. That's so that we can print a PostScript file to a PostScript printer with a minimum of fuss, which is the main duty of `print-postscript`.

The printer name could be an optional argument. If the printer name is present, `print-postscript` could work as it does now. If the printer name is absent, `print-postscript` could present a menu of printers & allow the user to select one. The menu could be a text-only list of numbered items.

A Change Log

2004 Nov 24 Fixed bug in the program. With the bug, print-postscript never read the PostScript file from standard input. Now it does unless you use the "-f" command line option.

B Other File Formats

- This document is available in multi-file HTML format at http://lisp-p.org/psw/.
- This document is available in Pointless Document Format (PDF) at http://lisp-p.org/psw/psw.pdf.

References

[1] Free Software Foundation. General public license. world wide web. http://www.gnu.org/licenses/licenses.html#GPL.