



pmtools

Mark Leighton Fisher

mark-fisher@mindspring.com

Introduction

The impetus that motivated Tom Christiansen to create `pmtools` was the idea of using a suite of small command-line programs to help manage Perl's extremely flexible module path system. In his February 1999 announcement he said:

This is pmtools -- a suite of small programs to help manage modules. The names are totally preliminary, and in fact, so is the code. We follow the "keep it small" notion of many tiny tools each doing one thing well, eschewing giant megatools with millions of options.

Nowadays, many shells provide at least a linear list of directories to search for program files. However, library components, whether object libraries, DLLs, modules, or what-have-you, are limited to a few, fixed locations which ordinary developers (and much less ordinary users) cannot modify. Meanwhile, Perl's `@INC` module path mechanism:

- Provides a list of paths to search for module and POD files much like a shell's PATH
- Is specific to the Perl binary, so I can have several Perls installed, each with their own list of module paths
- Divides Perl modules into CORE, system-, site-, and vendor-specific categories
- Divides Perl modules into pure Perl and binary-specific categories
- Is extensible via use lib, `$Config{siteperl}`, `sitecustomize.pl`, `PERL5LIB`, and `PERLLIB`
- Is ultimately extensible since I can put code references in `@INC`, thereby allowing me to change `@INC` processing however I like

Along with this flexibility, you also have greater complexity. Finding the path to a module to view its source—without a tool such as `perldoc` to help the process—requires an extraordinarily long command line (over 240 characters on my system).

You say, "There has to be a better way!" And there is with `pmtools`.

History

Tom Christiansen released the original `pmtools` in February of 1999. Although the code was supposed to be preliminary, Tom wrote `pmtools` so well that few changes were even suggested at the time. Since then, Tom has mostly retired from Perl, including maintenance of his perl.com websites for `pmtools`, `language.perl.com` and `mox.perl.com`.

I went to look for a copy of `pmtools` a couple of months ago while working at the Regenstrief Institute to find out that the only `pmtools` sources on the Net were in Linux packages.

Since my desktop is Cygwin on Windows XP, I wanted a simple tarball version of `pmtools`. Fortunately, I remembered that I had loaded a copy of the tarball version of `pmtools` on my home PC a couple of years back. Rather than just loading a personal copy of `pmtools` at work, I thought it would be better to make Tom's original `pmtools` into a version available on CPAN so that everyone with a command line and Perl could make use of them.

Why command-line tools?

Why do I need a suite of command-line tools for Perl module maintenance? Because module maintenance is mostly a linguistic activity, rather than a spatial or visual activity. When performing module maintenance, command outputs are either one name or a short