

Programming with CPAN.pm

by brian d foy
brian.d.foy@gmail.com



Anyone can easily make their own tools to deal with the Comprehensive Perl Archive Network (CPAN). Instead of relying on the tools that make other people's life easier, I can make a tool that does exactly what I need to do. Not every tool has to install modules, either.

I'm the author of the `cpan` script that comes with CPAN.pm (and hence with `perl` too), and examples in this article are also features of `cpan`. However, if my tool doesn't do the job you need, I'm not going to feel bad if you create your own. And, even though I use CPAN.pm, it's just as easy to create tools with CPANPLUS.

■ A note on terms -----

In talking about CPAN and its ecosystem has a problem: everything is called something that is pronounced alike. For this article, CPAN in all capital letters and no special formatting refers to the archive network itself. I'll always say CPAN.pm with the code font to refer to the module, and when I want to talk about my script that comes with CPAN.pm, I'll use the lowercase `cpan` in the code font. If you're reading this aloud to your kids at bedtime, make up special voices for each (a feature I want for my `Pod::Speakit::MacSpeech` module)..

Also note, that certain filesystems have made odd choices. I'm not going to single out the case-insensitive *but case preserving* Mac OS X filesystem because I am sure there are others, but it's something I have to remember when I use `perldoc`. When I tell `perldoc` to find something, it stops at the first thing it finds. For modules, it allows an extra `.pm` or `.pod` at the end of my term. However, in this case, `cpan` can match different things. Try these variations of `perldoc` to see what you get:

```
% perldoc cpan
% perldoc CPAN
% perldoc cPaN
% perldoc CPAN.pm
```

On my Mac, the first three bring up the documentation for my `cpan` script, although the various other BSDs I use handle the case as most people would expect. The filesystem is mostly to blame for this. I could choose to use UFS on my Mac, but then I'd have to do work. Just remember that you might have to tack on the extra `.pm` to look at the actual CPAN module.

■ The old way-----

The old CPAN.pm way involved starting a shell and issuing interactive commands. From the command line, I could include the CPAN.pm module with the `-M` switch:

```
% perl -MCPAN -e shell
```

Once I ran that, I had a prompt where I would have to tell CPAN.pm to do something.

```
cpan> install Business::ISBN Tie::Cycle
```

■ The new way-----

Although the interactive prompt is adequate, it's just a little bit too much work for me. I don't like all of that typing on the command line just to start it, so I created my own bash alias for it and put it in my `.bash_profile`:

```
alias cpan 'perl -MCPAN -e shell'
```

Now I just needed to type five characters (don't forget to count that newline that everyone ignores!) to start the shell:

```
% cpan
cpan[1]>
```

But then, I needed to type a lot more to install a module. The shell expects a command first then the arguments that go with that command:

```
% cpan
cpan[1]> install Set::CrossProduct
```

I got rid of my bash alias and created a Perl program instead. With arguments, it installed the name modules, and without arguments, it started the shell.

```
% cpan HTML::SimpleLinkExtor
% cpan
cpan[1]>
```

From there, I started to add other features to make things more convenient for me, Andreas included it in CPAN.pm so other people started to use it, and other people have contributed features they found convenient. Although my initial features