

Compiling My Own perl

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



It's not hard to come up with a situation where you want a different version of Perl than the system gives you. Perl 5.10 just came out and you're excited to use it, but you also need to keep everything you already have working. Or maybe you just got a CPAN Testers report that says that your module fails its tests on Perl 5.005. How can you test that before you upload a new version? You're on a shared-hosting service and you want to use a different Perl version, but you're stuck with Perl 5.8.0.

I realized, ratherly lately, that most people probably have never compiled their own perl binary. It's the disadvantage of Perl being so popular; it's preinstalled in many places, and on Windows I could use ActivePerl which requires no compilation. Although I still have the habit of downloading the latest stable Perl to compile and install it on any new system, most people don't do that. If it isn't preinstalled, someone does it for them.

When someone else compiles perl, they're usually not thinking about what you are going to do with Perl. They compile a vanilla perl that should work well with most tasks for most users. Some systems come with perl binaries that have features that most people don't want simply because the package manager turns on all of the features.

I might be able to benefit from optimizations by dumping features I don't want. For instance, do you have a threading perl? That might be nice, but if you don't want threading, you can tweak the performance a bit. Or, do you have a perl compiled with DEBUGGING? If you aren't hacking on the perl source code, you probably don't want DEBUGGING's performance penalty.

You can read about the compilation process in the README and INSTALL files in the top level of the perl source. I want to specifically apply this to compiling a new perl binary to see how it performs relative to what I already have, and how my applications work with new version of Perl, such as Perl 5.10. In each case, I want to try before I buy, and I want to do it as a normal user.

■ Getting the source -----

If you haven't compiled Perl before, you probably don't know where to get it. With package managers, someone has decided all of that information for you, and it might be downloading from it's own repository so it can apply patches and other customizations.

The perl source code is on CPAN with everything else, but it's in the /src directory instead of the /authors directory (where

the modules are). Start with the README file at <http://www.cpan.org/src/README.html>, which has links to most of the perl source distributions still on CPAN.

If you want to try a stable version of perl, look for an even number in the major release, such as 5.6, 5.8, or 5.10. The versions with an odd number are development versions that lead up to the next maintenance version. And, lately, the Perl 5 Porters have made an additional, explicit distinction about *testing* versions: these have a 0 in the point release, such as 5.8.0 and 5.10.0. Even though the testing versions are in the maintenance branch, they aren't for production use.

This jargon is different that what you may have noticed before. Formerly, the maintenance branch was the "stable" branch and there was a *stable.tar.gz*, and for the development branch there was a *devel.tar.gz*. The Perl 5 Porters no longer use these special links to the most current distributions in those branches. Now you have to look for the particular distribution that you want.

■ Configuring the source-----

Before I can compile perl, I have to configure the source for my system and preferences.

First, I run the Configure script, which prompts me for several questions it has about the installation. I can change the library search path, the installation directories, and many other things. Download the perl source for the version you want to try, unpack it, and run Configure. If I just want the defaults, I could skip all the questions and let Configure make its best guess:

```
prompt> ./Configure -des
```

That's not a lot of fun, and everyone should go through all the questions at least once to see what they are and what Configure is doing. In that case I run it without any options:

```
prompt> ./Configure
```

When I'm answering the questions my self, I have to watch out that I don't overwrite */usr/bin/perl*. If I already have one in place, Configure shouldn't try to overwrite it, but if I'm getting bored with the questions and am going quickly through them, I might give the wrong answer to this one by tapping on the y key too many times: