NEWS & INFO

A perl6 binary

http://use.perl.org/article.pl?sid=07/12/30/0912211

chromatic posted to use.Perl the instructions to build a command named `perl6` by using a `pbc_to_c` tool which converts Parrot byte code to an executable. Don't get too excited, though: it's really just a shortcut for running the same Perl 6 programs that you could previously, but without seeing the parroty bits on the command line. Don't discount that either; perception is a big deal to many people, so even if Perl 6 isn't closer, it feels closer.

Chocolate Perl for 2008

http://use.perl.org/~Alias/journal/35209

Vanilla Perl was the first project to get Perl and CPAN module installation to work on Windows just like it does on unix. That gave way to Strawberry Perl (*strawberryperl.com*), the single file installer that gives you everything that you need. That's great it your coming from the unix world and want to work the same way on Windows. The next step, according to Adam Kennedy, is to create a distribution that works like other Windows software, including a graphical interface to some sort of CPAN client.

A faster Moose

http://use.perl.org/~Stevan/journal/35102

Moose, a new object system for Perl 5 built on top of the existing object system for Perl 5, got a big performance benefit by using some clever XS code and internal caching to cut it's load time in half. This has been one of the biggest factor limiting its use in vanilla CGI environments. Moose gets another speed-up from Perl 5.10 enhancements that change how Perl looks up and finds methods, which is a continuing effort in dynamic language that can change method definitions or add or subtract methods at runtime. Previously, any change invalidated the method lookup cache and Perl would have to start over. With Perl 5.10's `mro` (method resolution order) pragma Moose can control that a bit so a package change only affects that package cache.

Besides the speed-ups, the versions of Moose from 0.34 on include a way to exclude methods from a role (handy when two roles provide the same method name), an easier interface to extend metaclasses, and `MooseX::Compile`, a way to pre-compile Moose classes to a *.pmc* (compiled Perl module) so you don't have to compile it every time you want to use it.

Solving Y2038

http://use.perl.org/article.pl?sid=08/02/07/197204

Perl currently relies in many places on the definition of `time_t`, the *libc* `struct` that describes how big the system time might be. On a 32-bit machine, that's probably 32 bits, On a 64-bit machine, that's probably 64 bits. Or maybe it's not. Either way, it shouldn't matter to Perl or any of its date processing modules.

To get around this, Michael Schwern submitted a patch using the C function localtime_r from Paul Sheer (*2038bug.com*). The `localtime_r` function uses its own 64-bit `gmtime` and does a bit of magic to figure out current time zones for the years after 2038 (as if that will even matter 30 years from now). Don't think that this

UPCOMING CALENDAR

See *The Perl Review* Community Calendar at *http://www.theperlreview.com/community_calendar*

APRIL 5-7
Oslo QA Hackathon
Oslo, Norway
http://perl-qa.hexten.net/wiki/index.php/OsloQAWorkshop2008

MAY 15-16
YAPC::Asia
Tokyo, Japan
http://conferences.yapcasia.org/ya2008/

JUNE 6-7
Portuguese Perl Workshop
Braga, Portugal
http://workshop.perl.pt/ptpw2008/

JUNE 16-18
YAPC::NA
Chicago, IL
http://www.yapc.org/

JULY 13-15
Open Source Convention
Portland, OR
http://conferences.oreilly.com

AUGUST 13-15
YAPC::EU
Copenhagen, Denmark
http://yapceurope2008.org/

is a problem just yet? In 2004, we passed the half way mark from 1970 to 2038. Adding two near-current times to take an average has a sign overflow in 32 bits. If Perl doesn't rely on the system time `struct` definition, Perl doesn't have to care. With this patch and a fix-up to some modules that might rely on the underlying system definitions, you shouldn't have to care either. Schedule that lunch in 2039 with confidence!

Perl 5.10 is out, on to Perl 5.12

http://www.xray.mpe.mpg.de/mailing-lists/perl5-porters/2008-01/thrd5.html

We were already talking about Perl 5.10 and already had articles in *The Perl Review* before it was officially released on December 18, Perl's birthday 20th birthday. Jon Allen updated *perldoc.perl.org* to show the Perl 5.10 documentation by default, although you can still get to the Perl 5.8.8 documentation (*http://perldoc.perl.org/5.8.8/*). Not long after the release, people on the *perl5porters* mailing list were working on a wish list for Perl 5.12. Heady from the inclusion of Perl 6 ideas in Perl 5.10, it looks as 5.12 may be even closer to Perl 6 if it adds named subroutine parameters, junctions, and methods distinguishable from normal subroutines. Other suggestions included a fully Dtrace-able perl and automatic `bignum` support.

Perl 5.8.9 before Christmas?

http://www.xray.mpe.mpg.de/mailing-lists/perl5-porters/2008-01/msg00783.html

Nicholas Clark, the maintainer for the Perl 5.8 branch, mentioned that he'd like to release Perl 5.8.9 this year. He produced a snapshot of his current work so the smoke testers could go to work on it. If you think any of this is an easy task, read Nicholas's outline from last year about what a maintainer has to do before he can make a release (*http://www.nntp.perl.org/group/perl.perl5.porters/2007/03/msg122190.html*). It's not the code that is hard, but not breaking the entire world six hours after a new stable release.

A Dtrace-able perl

http://search.cpan.org/dist/Devel-DTrace

Dtrace (*http://docs.sun.com/app/docs/doc/817-6223*) is a dynamic tracing utility from Sun designed to let you peer into your compiled programs (so, the `perl` binary, not your Perl scripts) and the system to see what's happening. Programs written with Dtrace in mind can provide probes that Dtrace can query or watch. Working from earlier patches by Alan Burlison and Richard Dawe (*http://blogs.sun.com/alanbur/entry/dtrace_and_perl*), Andy Armstrong created `Devel::DTrace` so you can watch subroutine entries and exits. To make it work, `Devel::DTrace` creates a custom perl binary named `dtperl` that provides the right hooks by modifying some of the runops. Run your program with `dtperl` instead of perl, then use DTrace to watch the action. It looks as if a DTrace-able perl might run slightly faster, too, according to Andy's measurements (*http://www.xray.mpe.mpg.de/mailing-lists/perl5-porters/2008-01/msg00302.html*) The module is still in its infant stages of development, and Andy suggested that some of this show up in Perl 5.12.

Perl5porters summaries are back

http://use.perl.org/search.pl?tid=12

In late December and after a year of neglect, David Landgren restarted the summaries of the perl5porters mailing list, posting them to *use.perl.org*. Each week, David goes through all of the list traffic and writes a short summary of each conversation so you can get the main point and the decision without the long discussions.

Perl in Microsoft's Winter Scripting Games

http://www.microsoft.com/technet/scriptcenter/funzone/games/default.mspx

By the time you read this, Microsoft's 2008 Winter Scripting Games is already over. From February 15 to March 3, Microsoft invited the public to compete in either beginner or advanced divisions to solve 10 programming problems using Visual Basic, Powershell, or, making its first appearance in the Games, Perl. The judges tested the Perl solutions under ActiveState's ActivePerl, and ActiveState is sponsoring some of the prizes, including some Perl Dev Kits. ActiveState employee Jan Dubois is providing commentary and his own solutions (*http://www.microsoft.com/technet/scriptcenter/funzone/games/games08/experts.mspx*). By press time, the judges had not announced winners or awarded the prizes.

Mark Jason Dominus's "Welcome to my ~/bin"

http://blog.plover.com/prog/perl/mybin.html

In 2004, Mark Jason Dominus presented a talk about the contents of his personal bin directory, including the various small command-line scripts and utilities that he created to get his work done. He's put the entire talk online. Many of the tools start with a how he over-engineered the problem or re-invented the wheel, which might be more interesting than the other bits of the code. He's released this content for free.

More Perl in the social web

Last issue we reported the Perl infusion into LinkedIn. Get a handful of people started on a service and they tell two friends, each of whom tell two friends, and so on and so on. This time around it's Ohloh (*http://www.ohloh.net/*), which tracks activity in open source projects through their source control repositories. Not only does it analyze a single repository, but it can look at the numbers across all repositories to see who is using what and how they are using it. Ohloh can figure out the usual metrics, such as language that you are using, the number of comment lines, and the number of contributors, and they can do that across all projects it tracks. See their language comparison (*http://www.ohloh.net/languages*) for aggregate numbers across all projects. Note that it only tracks projects submitted to it, and doesn't fully support git repositories yet.

Also this month, there is a bit of a Perl move into SlideShare, which allows people to upload their presentations so other people can see them. People can group presentations by events or interest. There is a Perl group, currently with 85 talks, as well as events for YAPC::NA 2006 and 2007, and for Nordic Perl Workshops 2006 and 2007. There are also groups for Windy City and Shibuya Perl Mongers. Anyone can view the presentations, but to download them people need to have an account. SlideShare turns the presentations into Flash widgets that people can embed in their web page. Look for them on Perlcast (*http://www.perlcast.com*).