http://xkcd.com/224/

Perl

Baseless Myths & Startling Realities

by Tim Bunce, Sept 2009

Realities

I'm positive about Perl
Not negative about other languages

Pick any language well suited to the task

Good developers are always *most important*,
whatever language is used

Prefer ‘Good Developers’ over ‘Good Languages’

“For all program aspects investigated, **the performance variability that derives from differences among programmers** of the same language—as described by the bad-to-good ratios—**is on average as large or larger than the variability found among the different languages**.”

— An empirical comparison of C, C++, Java, Perl, Python, Rexx, and Tcl. IEEE Computer Journal October 2000

http://www.cis.udel.edu/~silber/470STUFF/article.pdf

“In the script group, the Perl subjects may be more capable than the others, because the Perl language appears more than others to attract especially capable people.” :)

Who am I?

Tim Bunce

Author of the Perl DBI module

Using Perl since 1991

Involved in the development of Perl 5

“Pumpkin” for 5.4.x maintenance releases

http://blog.timbunce.org

Perl 5.4.x 1997-1998

Living on the west coast of Ireland

~ Myths ~

http://www.bleaklow.com/blog/2003/08/new_perl_6_book_announced.html

~ Myths ~

Perl is dead

Perl is hard to read / test / maintain

Perl 6 is killing Perl 5

Another myth “Perl is slow” is squashed via Tim Bray:
http://www.tbray.org/ongoing/When/200x/2007/10/30/WF-Results

~ Myths ~

Perl is dead

Perl is hard to read / test / maintain

Perl 6 is killing Perl 5

Perl 5 isn't the new kid on the block

Perl is 22 years old

Perl 5 is 16 years old

A mature language with a mature culture

Contrast with "fire and motion" elsewhere:
How many times Microsoft has changed developer technologies in those years?

http://search.cpan.org/perldoc?perlhist Perl 1.0 1987-Dec-18, Perl 5.0 1993-Jul-31 thru 1994-Oct-17

http://perlbuzz.com/2007/12/perl-birthday-parties-across-north-america.html

Perl 0: embryo
Perl 1: infant
Perl 2: toddler
Perl 3: child
Perl 4: preteen
Perl 5: adolescent

You can guess where that's leading...

From "The State of the Onion 10" by Larry Wall, 2006
http://www.perl.com/pub/a/2006/09/21/onion.html?page=3

Buzz != Jobs

Perl5 hasn't been generating buzz recently

It's just getting on with the job

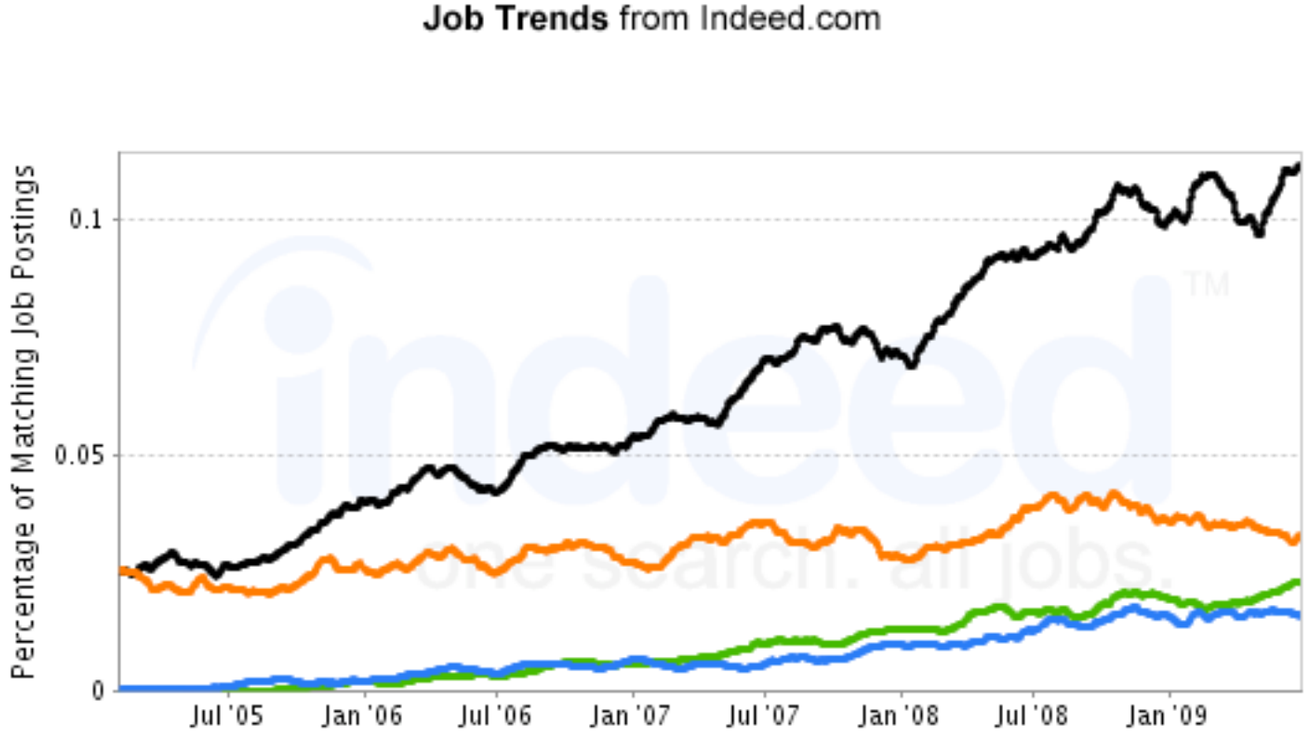
Lots of jobs

- just not all in web development

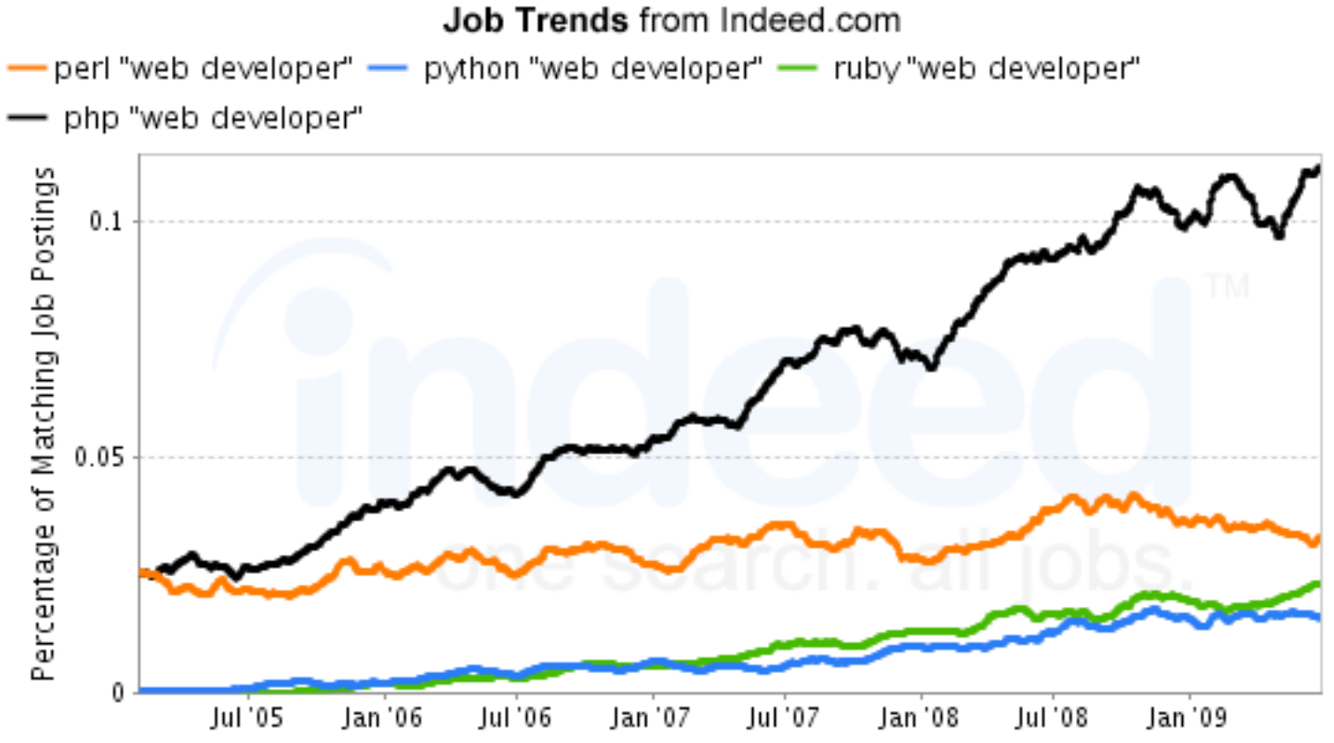
Web developers tend to have a narrow focus.

"At a recent finance technology conference in New York, the top 3 foundational technologies (by number of references) mentioned over the course of the conference by its speakers were: #3 - XML, #2 - SQL, #1 - Perl
There were no others mentioned." -- Richard Dice, president, The Perl Foundation,
in reference to O'Reilly's Money:Tech conference, New York, Feb 6 & 7, 2008.
Where "foundational technology" means a building block technology.

Guess the Languages



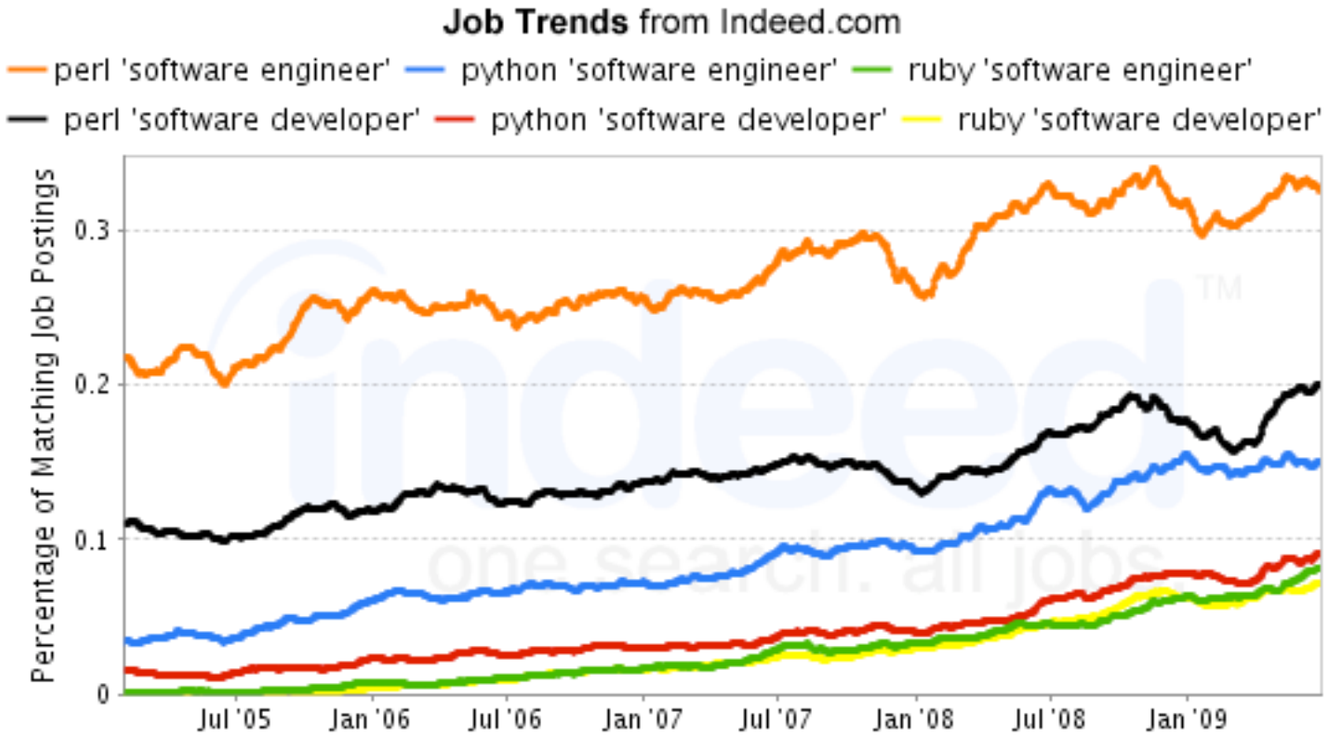
“web developer”



Yes, Perl is stagnant in *this* area,
but these are just “web developer” jobs

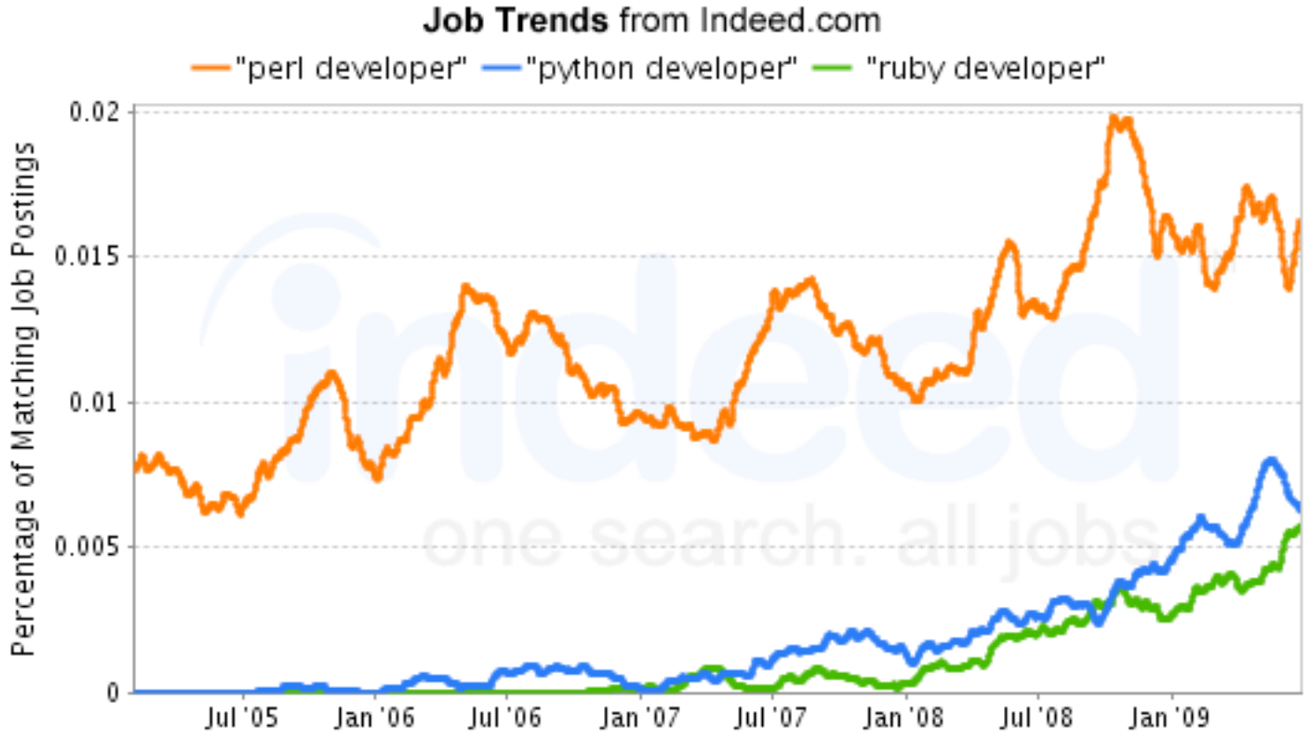
PHP - I’m not going to focus on PHP because it’s not a general purpose language.

“software engineer”



Perl is mentioned in *many* more software engineer/developer jobs.

“*foo* developer”



Perl is the primary focus of more developer jobs.
Want a fun new job? Become a Perl developer!

The existence of “*foo* developer” job titles is a sign of maturity.

Why is Perl doing so well?

Massive Module Market

Vast library of free code on "CPAN"

Over 4,500 active 'authors' (making releases)

Over 18,700 distributions (72,000 modules)

A third of all CPAN distributions have been updated in the 12 months!

Libraries are more important than languages.

Large user community leads to large contributor community
(given good community tools - more on that later)

http://stats.cpantesters.org/statscpan.html
http://search.cpan.org/recent

Top Modules

Many gems, including...

DBI DBD::* DBIx::Class Rose::DB::Object

Catalyst Moose DateTime

Algorithm::* Statistics::* Thread::*

XML::* HTML::* WWW::* Parse::*

Net::* Email::* POE::* Locale::*

Test::* Devel::Cover Perl::Critic perltidy

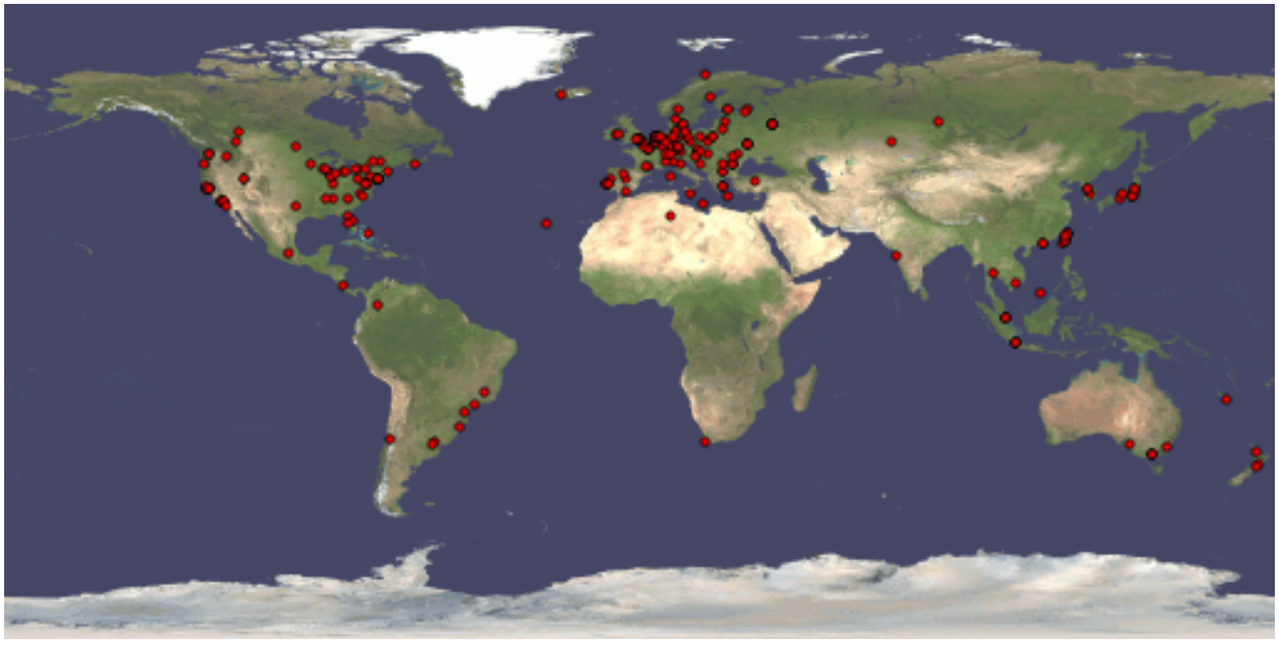
Quality varies on CPAN, naturally. (See Acme::* for some fun.)

http://www.serpentine.com/blog/2008/02/19/peruse-popular-perl-packages/

“Moose is pretty much the most exciting thing I’ve seen come out of the Perl 5 world in quite some time, and I’m really enjoying using it for my projects”
(My apologies if your favourite module isn’t included here.)

Comprehensive Perl Archive Network

377 mirrors in 56 regions (TLDs)



CPAN is a distributed replicated file archive that handles global distribution of perl modules.
CPAN is like a warehouse

Map image via http://cpansearch.perl.org/src/LBROCARD/Image-WorldMap-0.15/examples/cpan_mirrors/
http://mirrors.cpan.org/
Mirror status http://www.cs.uu.nl/stats/mirmon/cpan.html

Developer Services

Upload your perl module distribution
and you *automatically* get...

- global *distribution and archiving*
- *namespace* ownership and management
- a *bug tracking* queue at rt.cpan.org
- a *forum* at cpanforum.com
- *smoke testing* on many platforms

This is a mature environment with rich services.

I'll discuss smoke testing and quality a little later

search.cpan.org

The CPAN "shop window"

For each and every distribution:

- Browse well formatted inter-linked docs
- Links to forums, bug tracking, ratings, annotated documentation, dependency analysis, smoke test results, and more...

Over 600,000 unique visitors per month

Source for stats: google analytics

Home · Authors · Recent · News · Mirrors · FAQ · Feedback

in All CPAN Search

Stevan Little > Moose-0.38 permalink

Moose-0.38

This Release	Moose-0.38 [Download] [Browse] 15 Feb 2008
Other Releases	Moose-0.37 -- 14 Feb 2008 Goto
Links	[Discussion Forum] [View/Report Bugs (2)] [Dependencies] [Other Tools]
CPAN Testers	PASS (62) NA (1) [View Reports] [Perl/Platform Version Matrix]
Rating	★★★★★ (3 Reviews) [Rate this distribution]
License	Perl (Artistic and GPL)
Special Files	Build.PL MANIFEST Makefile.PL Changes META.yml README

Modules

Moose	A postmodern object system for Perl 5	0.38
Moose::Meta::Attribute	The Moose attribute metaclass	0.21

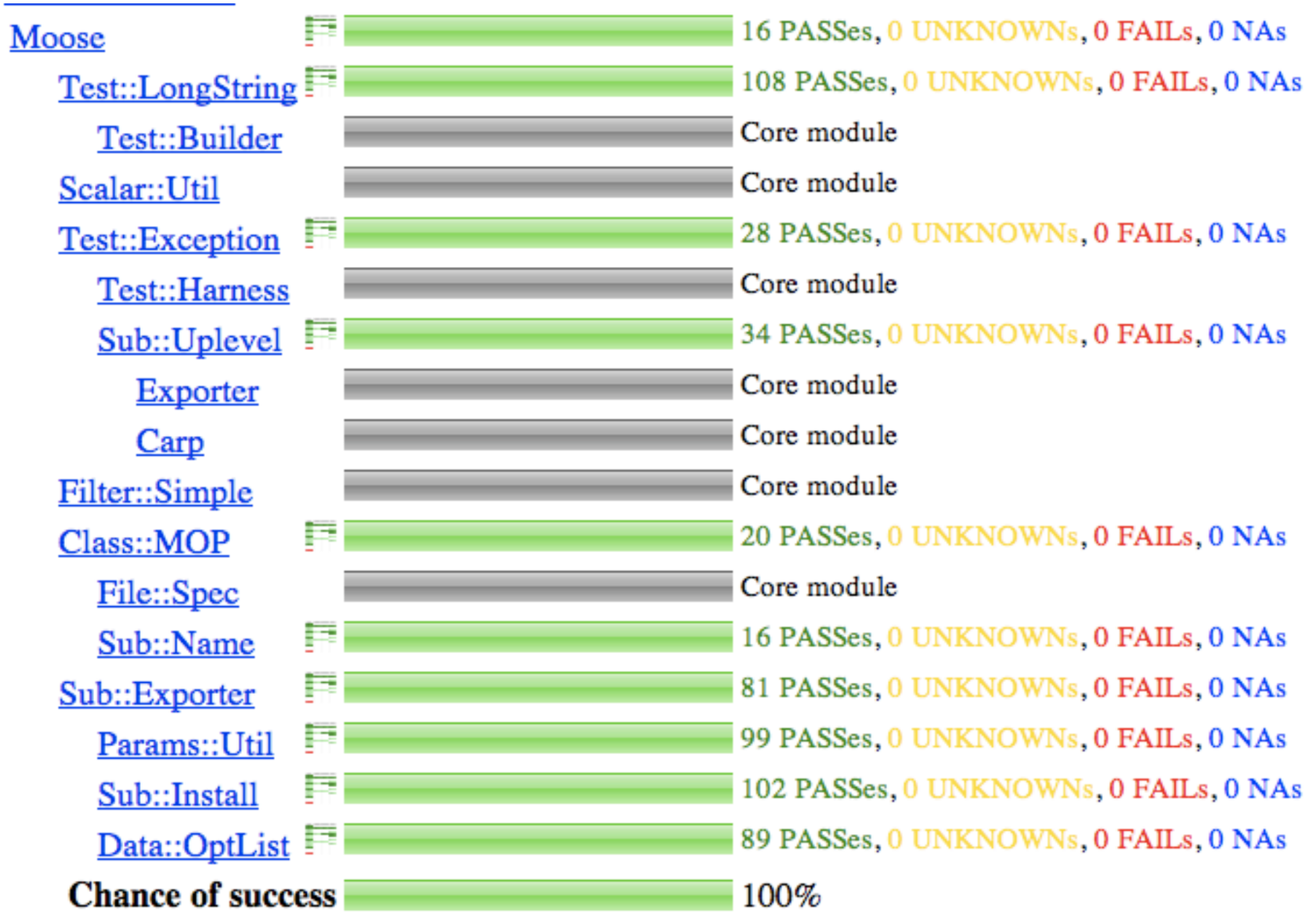
Documentation

Moose::Cookbook	How to cook a Moose
Moose::Cookbook::FAQ	Frequently asked questions about Moose

Example page for a distribution
Showing rich set of features
TOOLS Also has other tools, like grep'ing and diff'ing the distributions without downloading them.

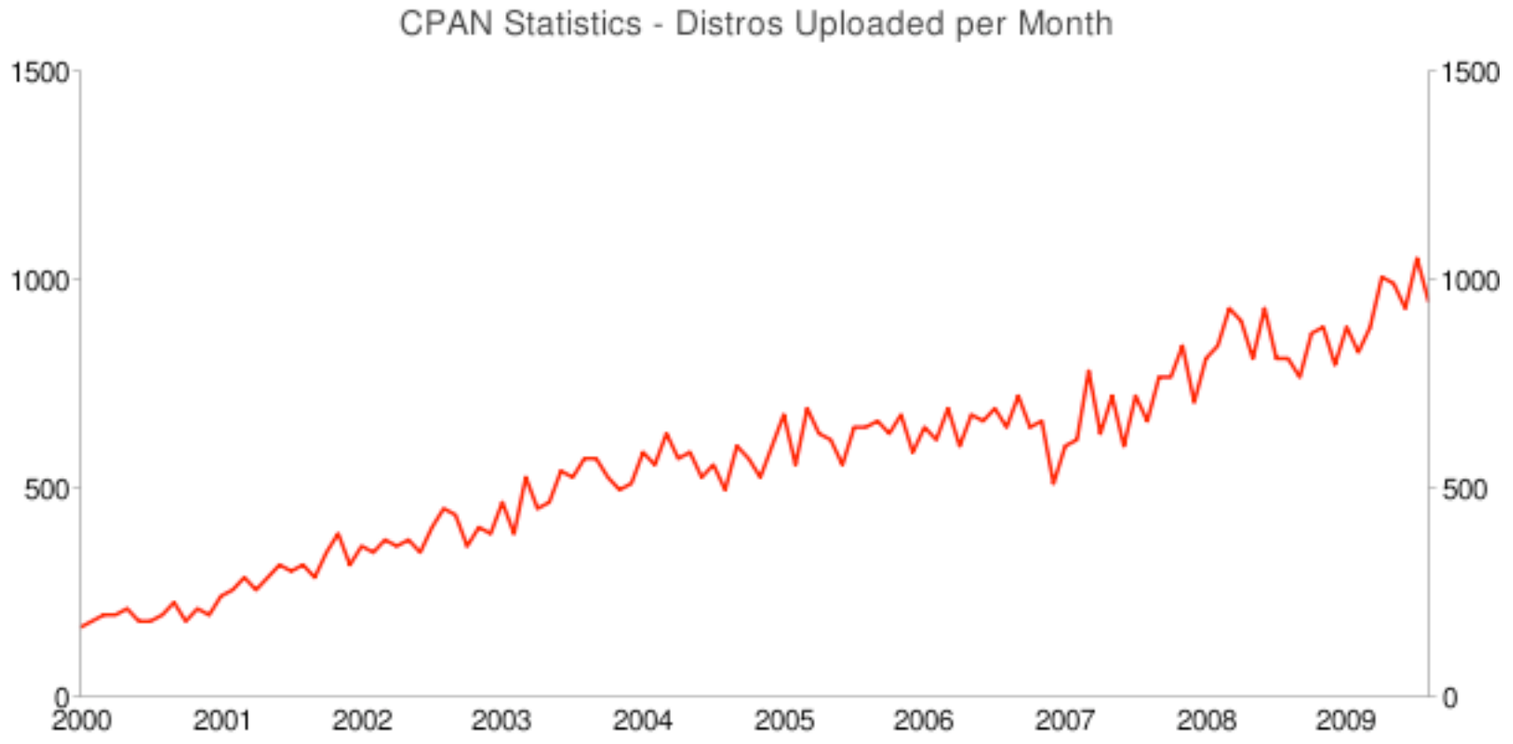
Dependency Analysis available for all Modules

http://deps.cpantesters.org/?module=Moose;perl=latest



Dependencies need not be hell!
Shows tree of dependencies. Identify risks.
Refine the view to match you particular operating system and perl version

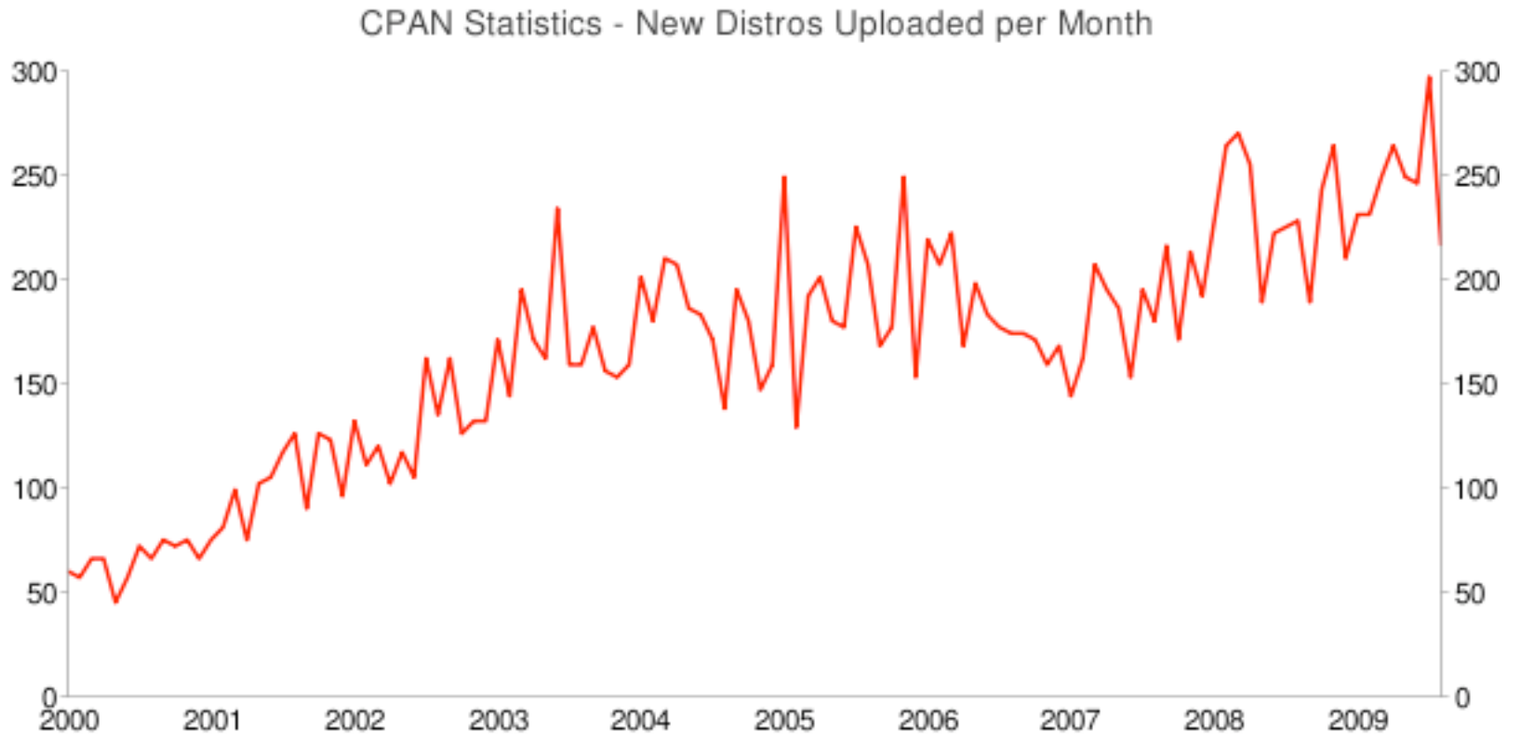
Rising Upload Rate



Steady growth in upload rate for over 10 years.

But could this be mostly just new versions of old code? Let's see...

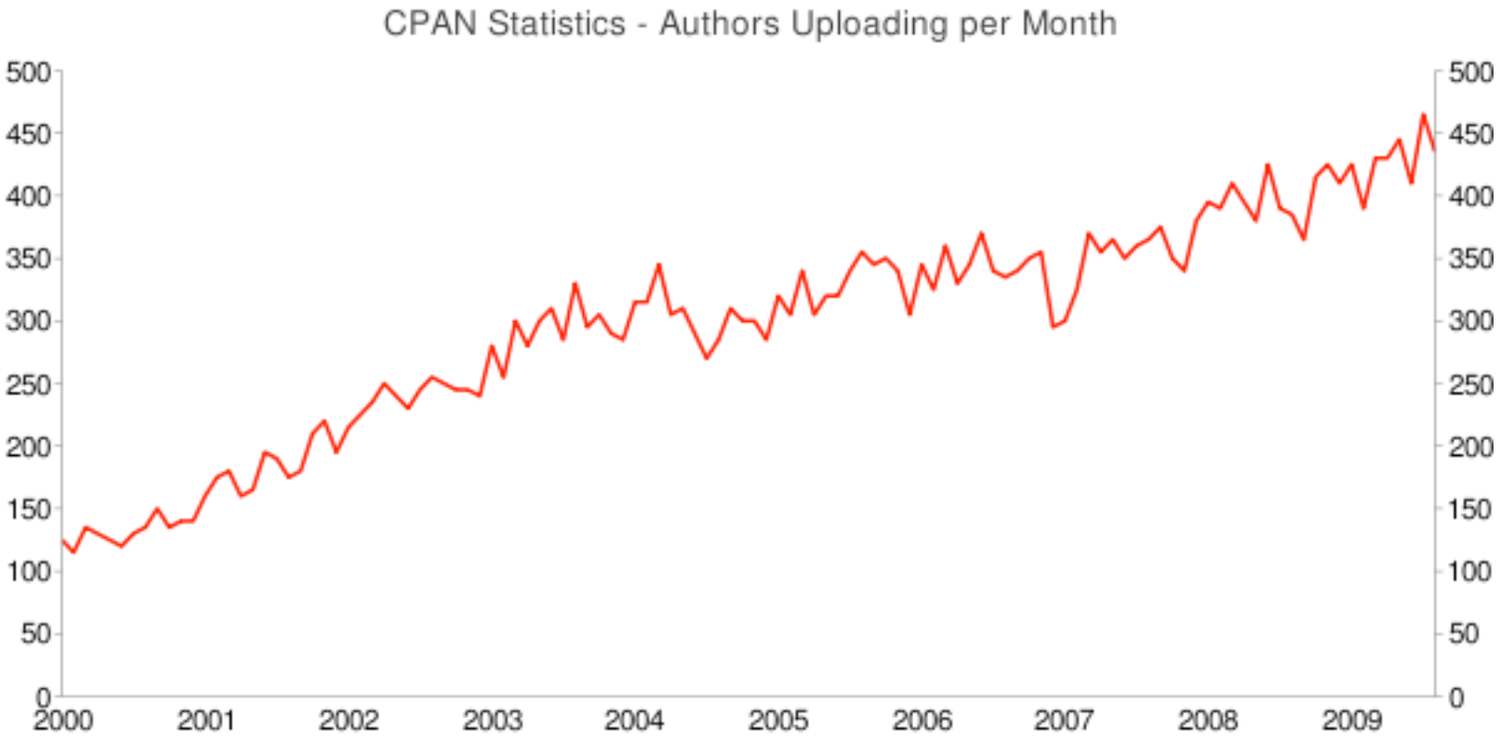
Expanding Scope



Every month CPAN is gaining a lot of NEW code offering NEW functionality.

But could this be coming from a static pool of authors? Let's see...

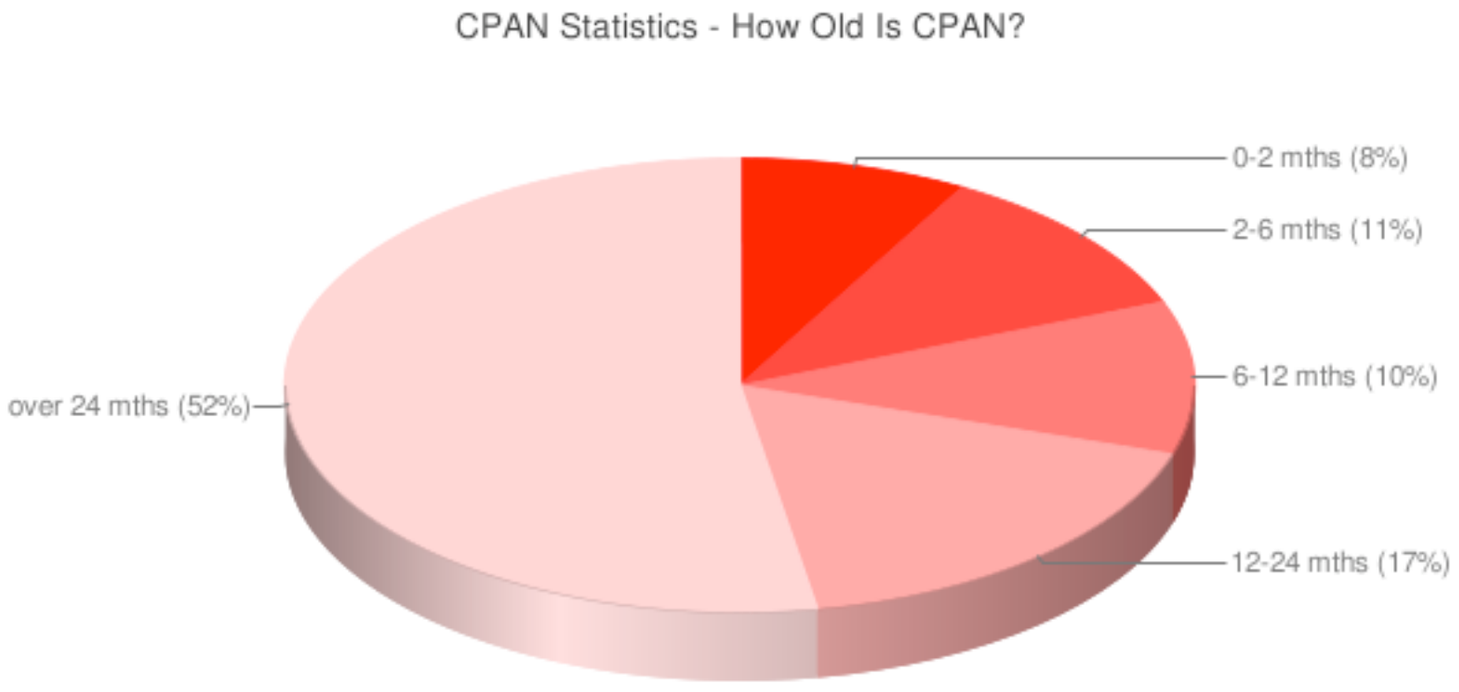
Growing Community of Contributors



Every month CPAN is gaining a lot of NEW contributors!

But is all this new code from new authors just a drop in an ocean of old code? Let's see...

How old is code on CPAN?



Tasty AND fresh!

It's the Community!

Christy John: “This is by far the most important thing I have noticed. Perl people are humble. ... Most of them are well versed in many languages. Actually all of them would tell you to learn other languages too and use one which suits you. I think the confidence is from the fact that once you start coding in Perl you are unlikely to move into others” -- http://thejoysofcomputing.wordpress.com/2009/09/02/why-i-love-perl-already/

redspike: People are “the reason that I started to look at Perl in the first place and why I have come back to it several times ... The people were fantastic, informative, erudite, witty, enthusiastic, genuinely interested in what you were doing or trying to do and had a love of the bizarre. ... I started hanging out on IRC ... I felt I belonged and that I was valued.” -- http://use.perl.org/~redspike/journal/39576

Brian “INGY” Ingerson: “But in the final analysis, it's the Perl community that keeps me around. These people are my weird looseknit family.” -- http://osdir.com/Article1534.phtml

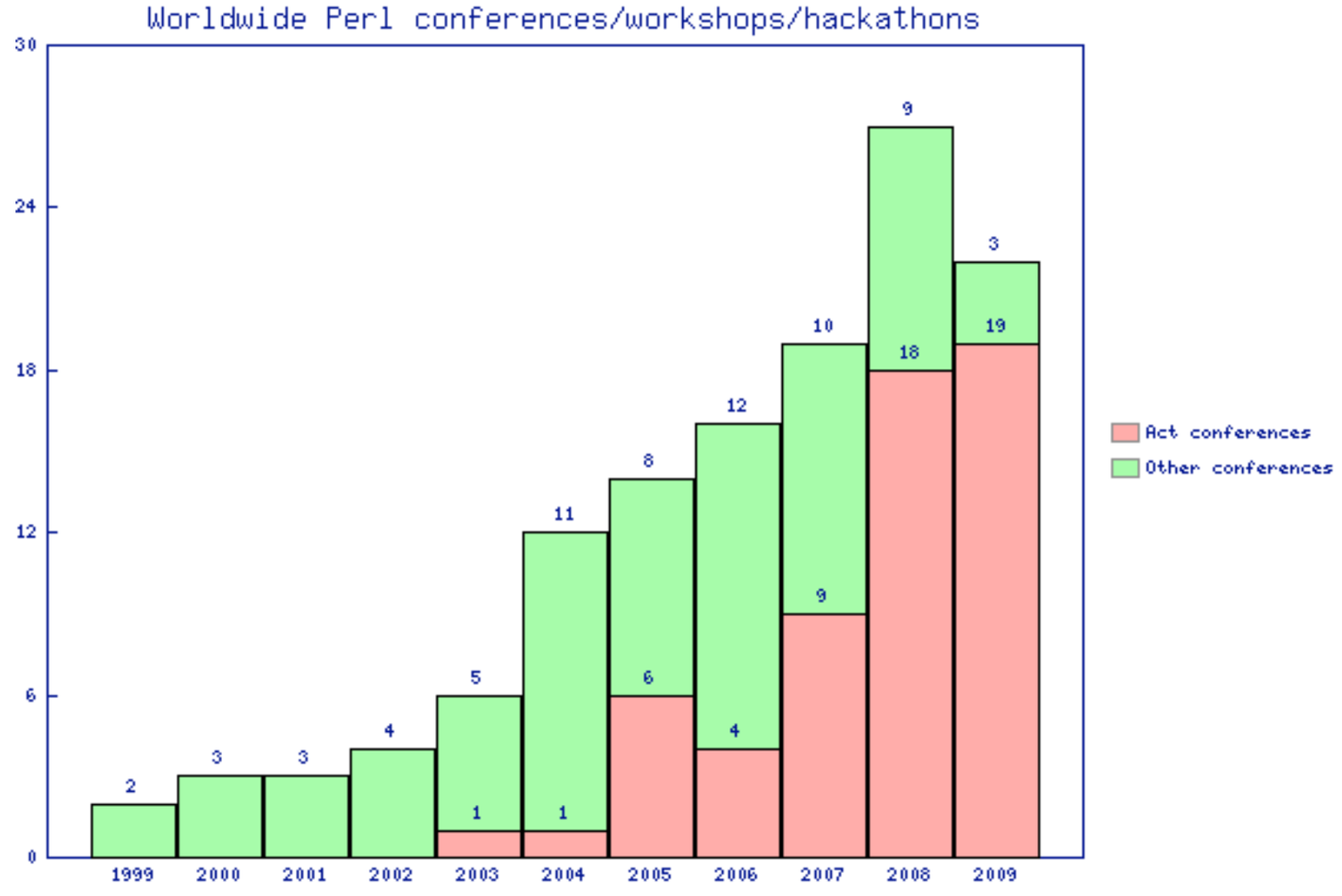
Perl Mongers Everywhere!

http://www.pm.org/groups/map.html

Perl Monger User Groups
Not all of these are active and some active groups aren't on the map.

Perl Mongers

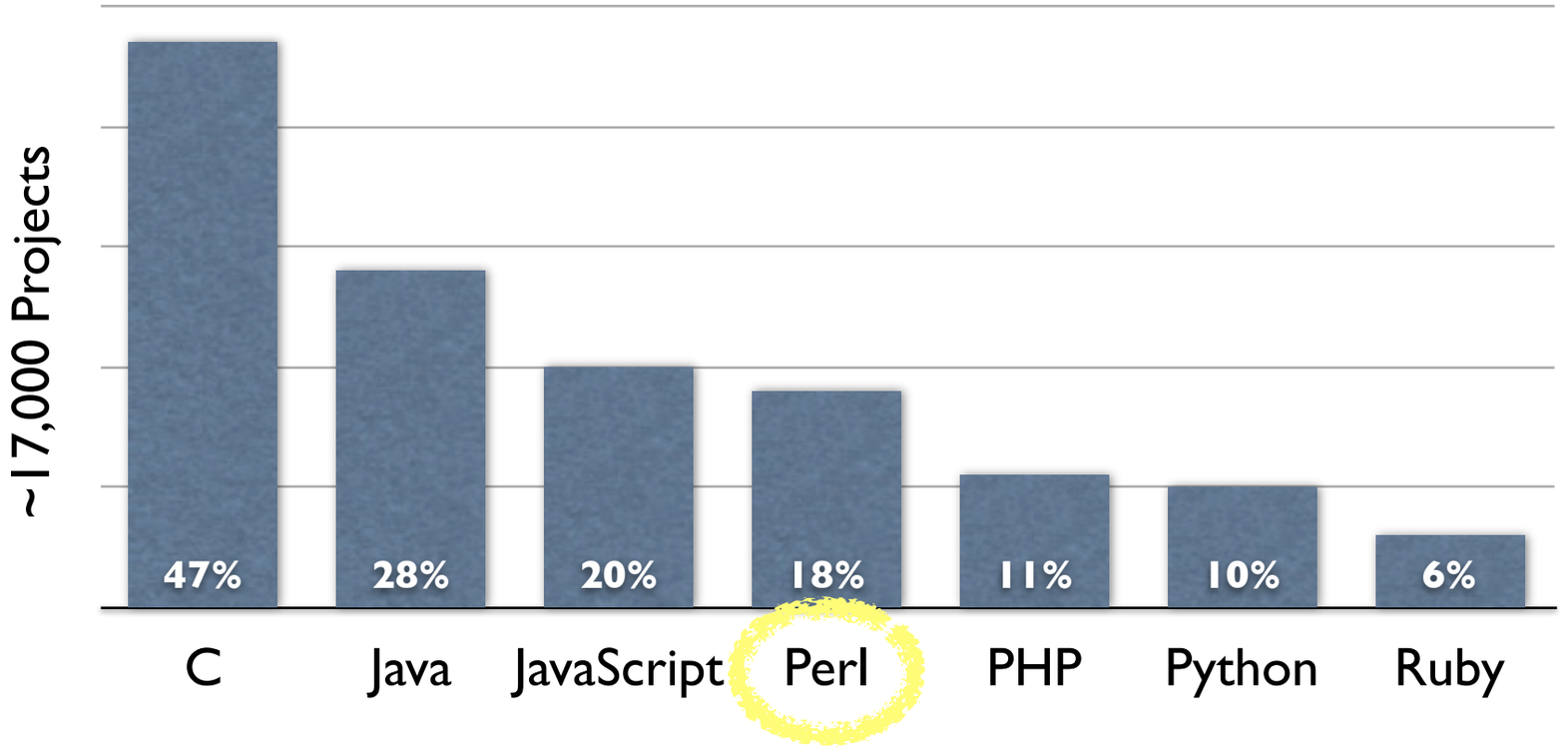
Grass Roots Community & Conferences



Act: “A Conference Toolkit” is a multilingual, template-driven, multi-conference web site that can manage the users, talks, schedule and payment for your conference

http://act.mongueurs.net/act-conferences.png
http://use.perl.org/article.pl?sid=08/04/29/0512200
http://www.yapceurope.org/events/conferences.yml
From: "Philippe Bruhat (BooK)": book@cpan.org

Languages chosen for new OS projects in 2008



Source http://www.blackducksoftware.com/news/news/2009-01-21

~ Myths ~

Perl is hard to read / test / maintain

Perl 6 is killing Perl 5

~ Myths ~

Perl is dead

Perl is hard to read / test / maintain

Perl 6 is killing Perl 5

“True greatness is measured by how much freedom you give to others, not by how much you can coerce others to do what you want.”

—Larry Wall

I greatly value the freedoms perl give me
Freedoms in the language, the community, the technology, the culture.
With freedom comes responsibility
You *can* write poorly in any language.
You *can* write beautiful code in Perl.
http://jeremy.zawodny.com/blog/archives/009873.html#comment-39486

Guidelines and Tools

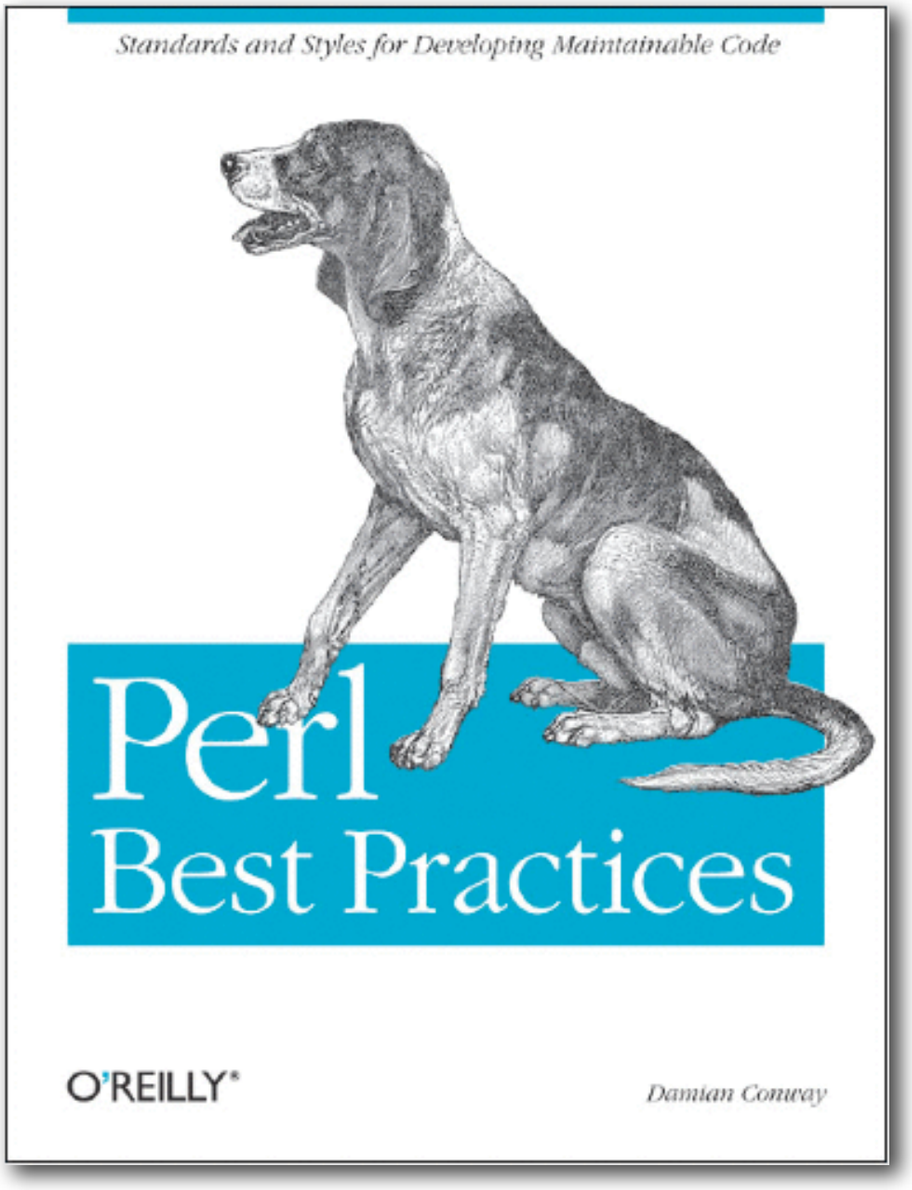
Perl Best Practices

Perl::Tidy

Perl::Critic

Test::*

Devel::Cover



Already a growing culture of quality and testing when the book came out in July 2005
256 guidelines

More than one way to do it, but...
Agreeing on one set of sane guidelines
is more important than the exact details of the guidelines.

PBP makes it easy for a team to agree a baseline policy “We’ll just follow PBP guidelines”
and just discuss local variations.

Perl::Tidy

Perl code beautifier

Works beautifully - can be trusted

Supports *many* options for personal styles

Perl Best Practices recommended options

Normalise the coding style of existing code.
Very simple and effective way to add clarity to a code base.

Now you've got pretty code,
but is it good code?

Perl::Critic

Static Code Analysis for Perl

Includes over 120 policies

Most based on Perl Best Practices

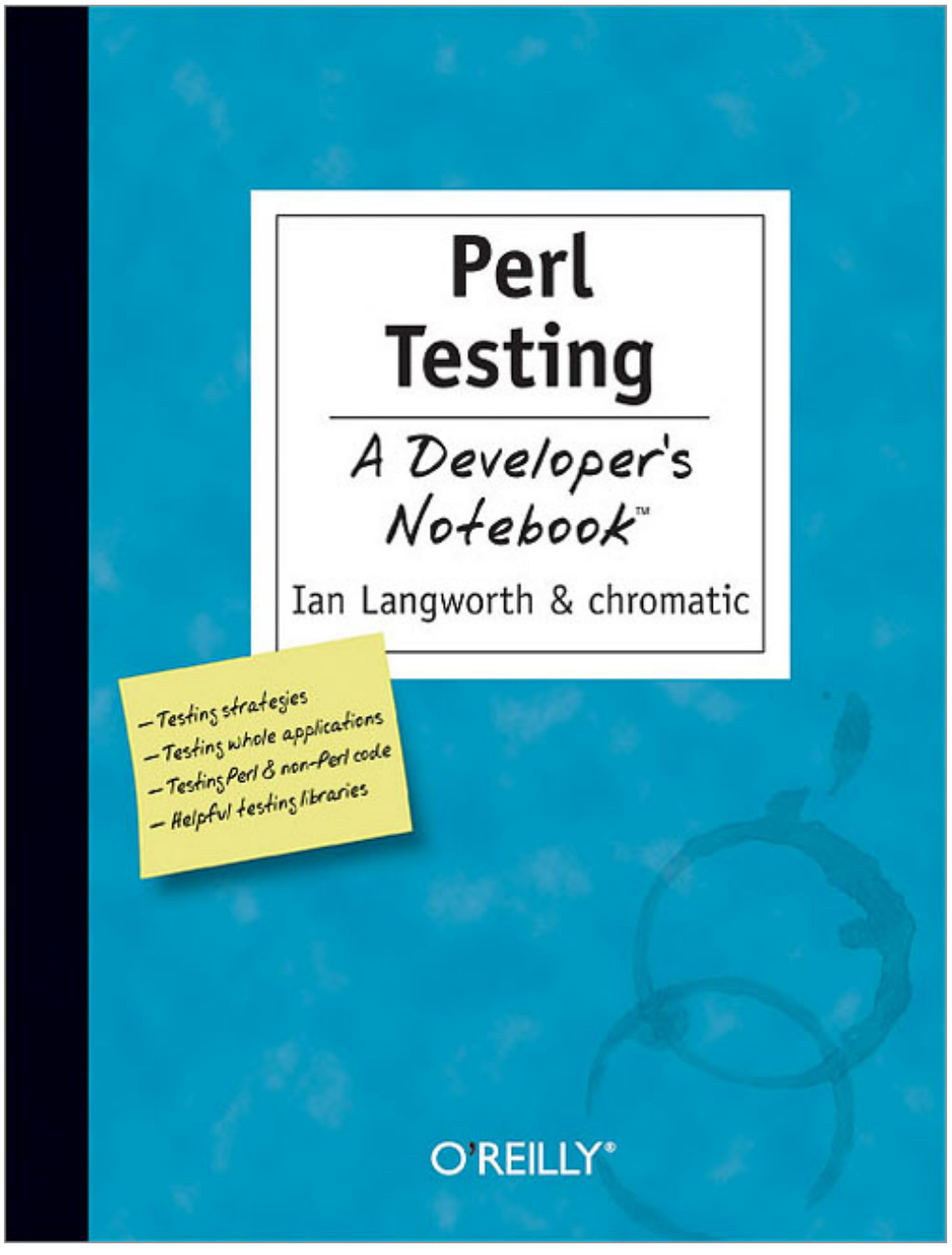
Grouped into levels and themes

Configurable and extensible
for local policies and styles

An extensible framework for creating and applying coding standards to Perl source code

Perltidy address the layout of code.
Perlcritic addresses the semantics.

Now you've got pretty code,
that follows best practices,
but does it work?



Published July 2005

http://www.oreilly.com/catalog/perltestingadn/

Test::*

Perl culture takes testing seriously

Excellent mature innovative tools for testing

Test Anything Protocol - producers/consumers
http://en.wikipedia.org/wiki/Test_Anything_Protocol

Test harness can run tests in parallel

Test::* modules make it easy to write tests

Over 450 Test::* distributions on CPAN!

Also many ‘mock’ modules for mocking objects and other functionality to ease testing
Test::Class provides xUnit styles tests. The test modules work together.
http://search.cpan.org/search?m=dist&q=Test%3A%3A&s=1&n=100

Now...
- you’ve got pretty code,
- that follows best practices,
- has tests and the tests pass,
- but how much code is actually exercised by the tests?

COVERAGE (next)

Devel::Cover

Coverage Analysis for Perl

Tells you what code has been executed

Statement, branch, condition, subroutine, pod

Produces drill-down reports in HTML

Includes documentation coverage analysis.

Devel::Cover Reports

http://pjcj.sytes.net/cover/latest/

File	stmt	bran	cond	sub	pod	total
AcePerl-1.91	72.48	52.74	43.18	74.66	56.36	64.15
Algorithm-Annotate-0.10	100.00	50.00	n/a	100.00	0.00	87.80
Algorithm-C3-0.06	100.00	100.00	100.00	100.00	100.00	100.00
Algorithm-Dependency-1.102	75.13	41.60	31.43	85.00	92.11	67.14
Algorithm-Diff-1.1902	87.47	78.68	77.66	80.77	100.00	83.85

file	stmt	bran	cond	sub	pod	time	total
blib/lib/Algorithm/Dependency.pm	95.2	83.3	n/a	93.3	100.0	34.4	92.7
blib/lib/Algorithm/Dependency/Item.pm	100.0	50.0	33.3	100.0	100.0	8.2	86.8
blib/lib/Algorithm/Dependency/Ordered.pm	100.0	75.0	n/a	100.0	100.0	9.3	93.7
blib/lib/Algorithm/Dependency/Source.pm	86.3	55.0	46.7	90.9	100.0	22.2	75.5
blib/lib/Algorithm/Dependency/Source/File.pm	100.0	50.0	n/a	100.0	100.0	7.8	85.2
blib/lib/Algorithm/Dependency/Source/HoA.pm	100.0	50.0	n/a	100.0	100.0	0.5	90.5
blib/lib/Algorithm/Dependency/Weight.pm	97.7	44.4	n/a	100.0	100.0	13.6	85.9
inc/Class/Inspector.pm	41.1	19.6	0.0	43.5	83.3	2.1	35.6
inc/Test/ClassAPI.pm	77.2	40.5	28.6	100.0	50.0	1.9	68.4
Total	75.1	41.6	31.4	85.0	92.1	100.0	67.1

blib/lib/Algorithm/Dependency/Weight.pm			
Criterion	Covered	Total	%
statement	42	43	97.7
branch	8	18	44.4
condition			n/a
subroutine	12	12	100.0
pod	5	5	100.0
total	67	78	85.9

line	stmt	bran	cond	sub	pod	time	code
1							`package Algorithm::Dependency::Weight;`
223							`sub weight_all {`
224	1			1	1	11	`my $self  = shift;`
225	1					13	`my @items = $self->source->items;`
226	1	50				15	`defined $items[0] or return undef;`
227	1					10	`$self->weight_hash( map { $_->id } @items );`
	6					67	

line	l	!l&&r	!l&&!r	condition
140	1037	0	0	`$$self{'loaded'} or $self->load`
161	11	3	0	`$$self{'loaded'} or $self->load`
182	4	1	0	`$$self{'loaded'} or $self->load`

Your Power Tools

Perl Best Practices

Perl::Tidy

Perl::Critic

Test::*

Devel::Cover

~ Myths ~

Perl is dead

Perl is hard to read / test / maintain

Perl 6 is killing Perl 5

~ Myths ~

Perl is dead

Perl is hard to read / test / maintain

Perl 6 is killing Perl 5

Perl 6 *saved* Perl 5!

"Perl 5 had already started dying, because people were starting to see it as a dead-end language.

It seemed odd at the time, but when we announced Perl 6, Perl 5 suddenly took on a new life."

—Larry Wall, 2002

http://www.perlfoundation.org/perl6/index.cgi?state_of_the_onion

Perl 6 *saved* Perl 5!

In 2000 development of perl was struggling

The Perl 6 RFC process "vented spleen"

Perl 5 development has gone smoothly since

Much refactoring driven by Perl5-on-Parrot

Many new features inspired by Perl 6 work

Back around 2000...

- Perl5 was getting very hard to maintain
- Bickering on the mailing lists
- Few volunteers for any real work
- Lack of direction

Using Perl 5 as one of the backends for Perl 6.

Perl 5.10

Perl 5.10.0 was released in December 2007
Five years after 5.8.0, two years after 5.8.8

Refactored internals

many fixes, more speed, less memory

Smart matching, named captures, state variables, defined-or, field hashes, pluggable regex engines, say, trie-based non-recursive pattern matching, and more...

http://www.slideshare.net/rjbs/perl-510-for-people-who-arent-totally-insane

Perl 5.10.1

Perl 5.10.1 was released in August 2009

- Many bug fixes, and optimized internals

Source code moved to git

- Reorganized development process
- 5.10.x now strictly maintenance only
- Moving to more rapid release cycle

A Culture of Testing

Another bonus from Perl 6:

> *Strong test suites for Perl 5 code are needed to ensure backwards compatibility*

Perl Test Suite

2002: Perl 5.8.0 had **26,725** core tests
+41,666 more for bundled libraries etc.

2007: Perl 5.10.0 has **78,883** core tests
+109,427 more for bundled libraries etc.

2009: Perl 5.10.1 has **92,697** core tests
+142,101 more for bundled libraries etc.

For perspective: in Jan 2008 Ruby has ~1,400 core tests plus ~14,000 for bundled libraries etc.
(Ruby now has http://rubyspec.org project but I don't know how many tests/assertions they have.)

See comments in http://www.oreillynet.com/onlamp/blog/2007/05/trust_but_verify.html
and http://reddit.com/info/1uzda/comments/

Stats from "make test" and "cd ./t && perl TEST -core"

Module Test Suites

CPAN Testers Network (CPANTS):

Automated smoke testing of all CPAN uploads

Runs the test suite included in distribution

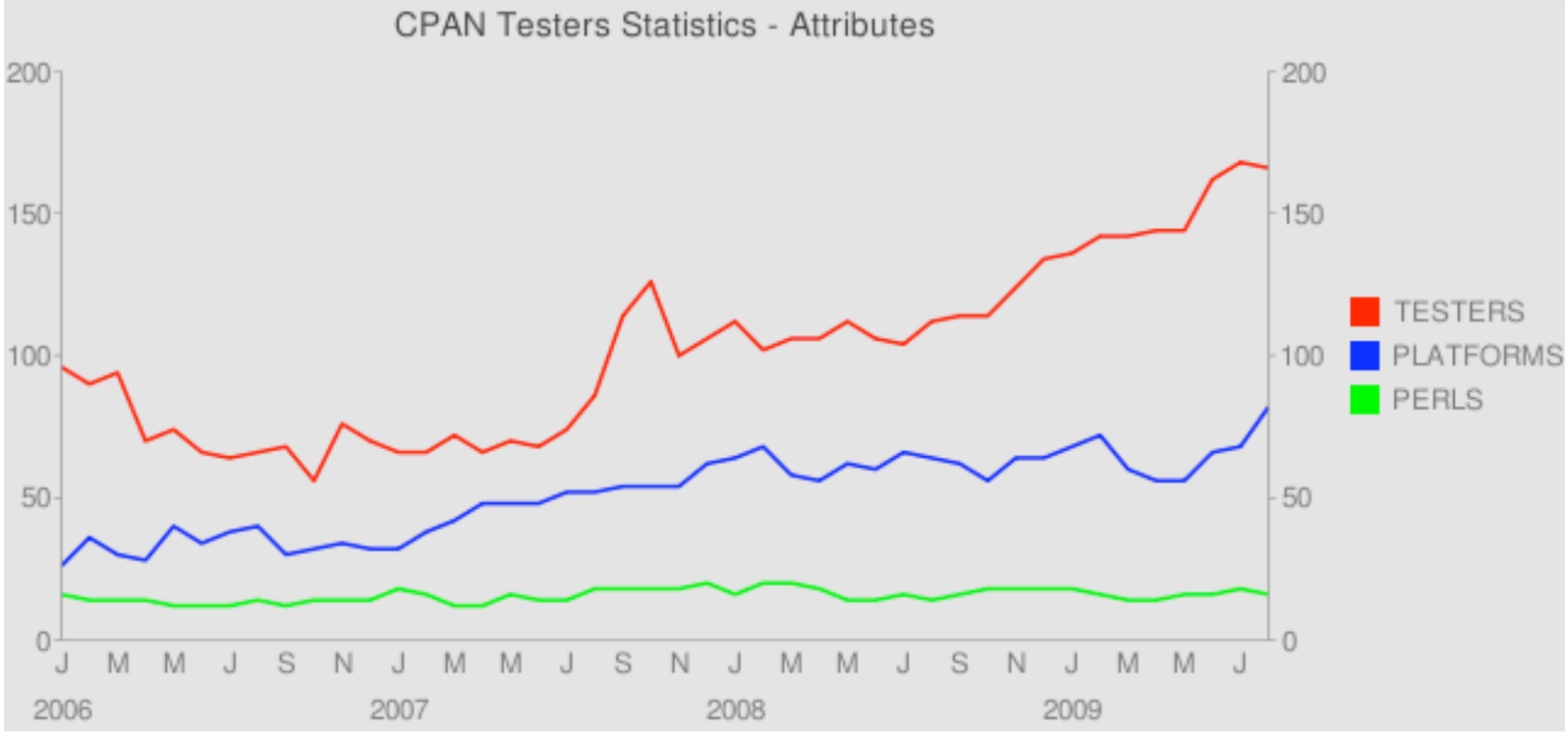
~ 60 different platforms
~ 20 different perl versions

Immediate feedback for developers

Some by completely automated robots in virtual machines.

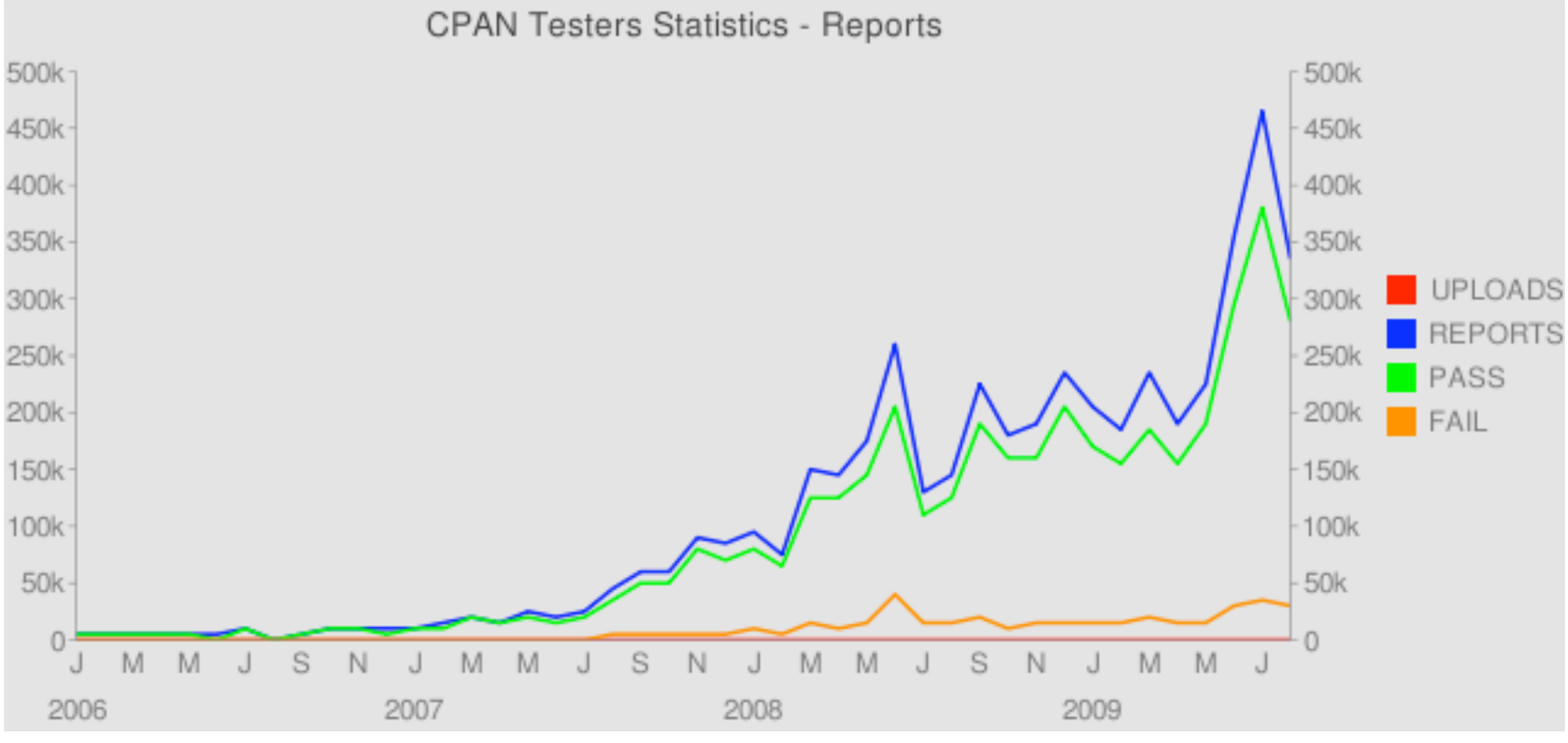
Some by users who submit reports as they download and test via the installer.

Version & Platform Coverage



http://birmingham.pm.org/talks/barbie/stats-of-cpan/slide431.html

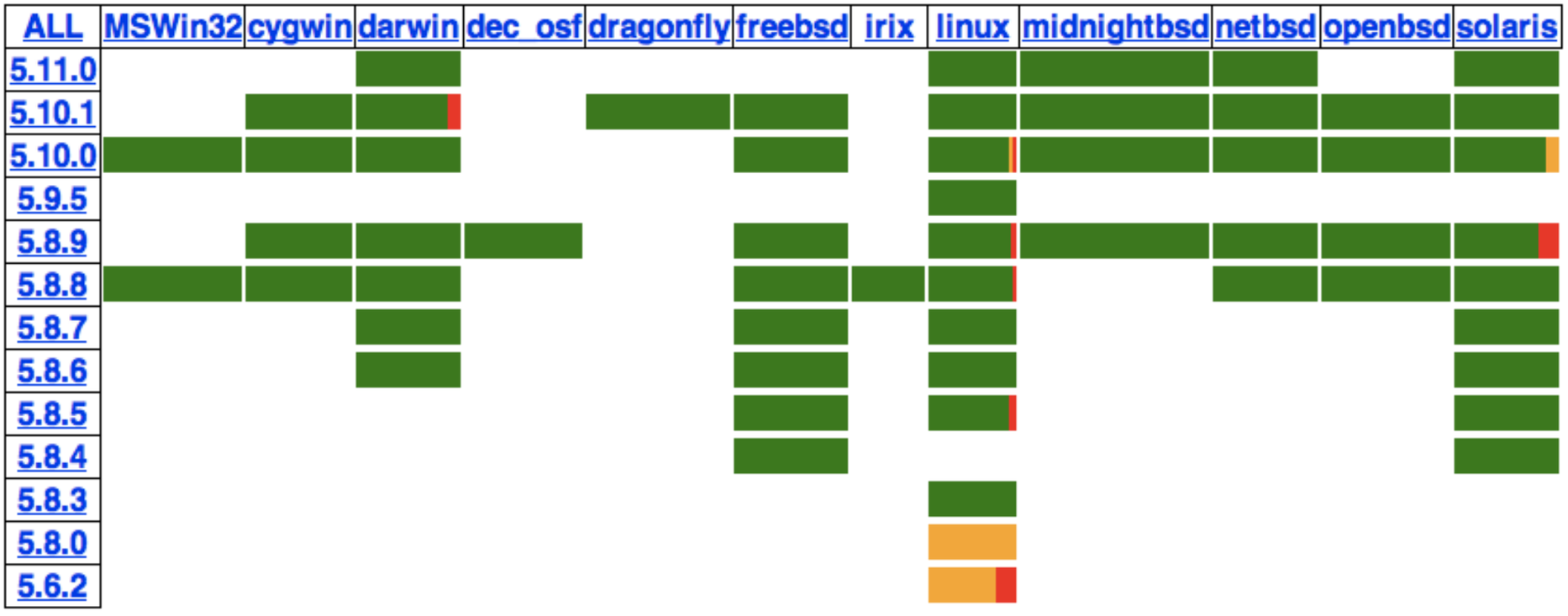
Over 300,000 Tests/Month



http://birmingham.pm.org/talks/barbie/stats-of-cpan/slide430.html

Platform and Version Result Matrix available for all distributions

Example results for DBI 1.609:



http://matrix.cpantesters.org/?dist=DBI+1.609

Milestones in the Perl Renaissance

2001: Lexical file-handles, Test::Simple
2002: Module::Build, Test::Builder,
2003: PAR, Perl 5.8.1
2004: Perl 6 Apocalypse 12 (Roles), CPANTS
2005: PPI, Perl::Critic
2006: CPAN Testers, Moose, Strawberry Perl
2007: Devel::Declare, local::lib
2008: Padre, Enlightened Perl Organization
2009: Iron Man Blogging Challenge

http://www.modernperlbooks.com/mt/2009/07/milestones-in-the-perl-renaissance.html

All this happened after the Perl 6 project was announced.
Modern Perl, 21st Century Perl, is very different to 20th Century Perl.

~ Myths ~

Perl is dead

Perl is hard to read / test / maintain

Perl 6 is killing Perl 5

~ Myths ~

Perl is dead

Perl is hard to read / test / maintain

Perl 6 is killing Perl 5

Community Resources

"Awesome community.
Perl people tend to be laid-back and friendly."

perlbuzz.com - Headlines and articles

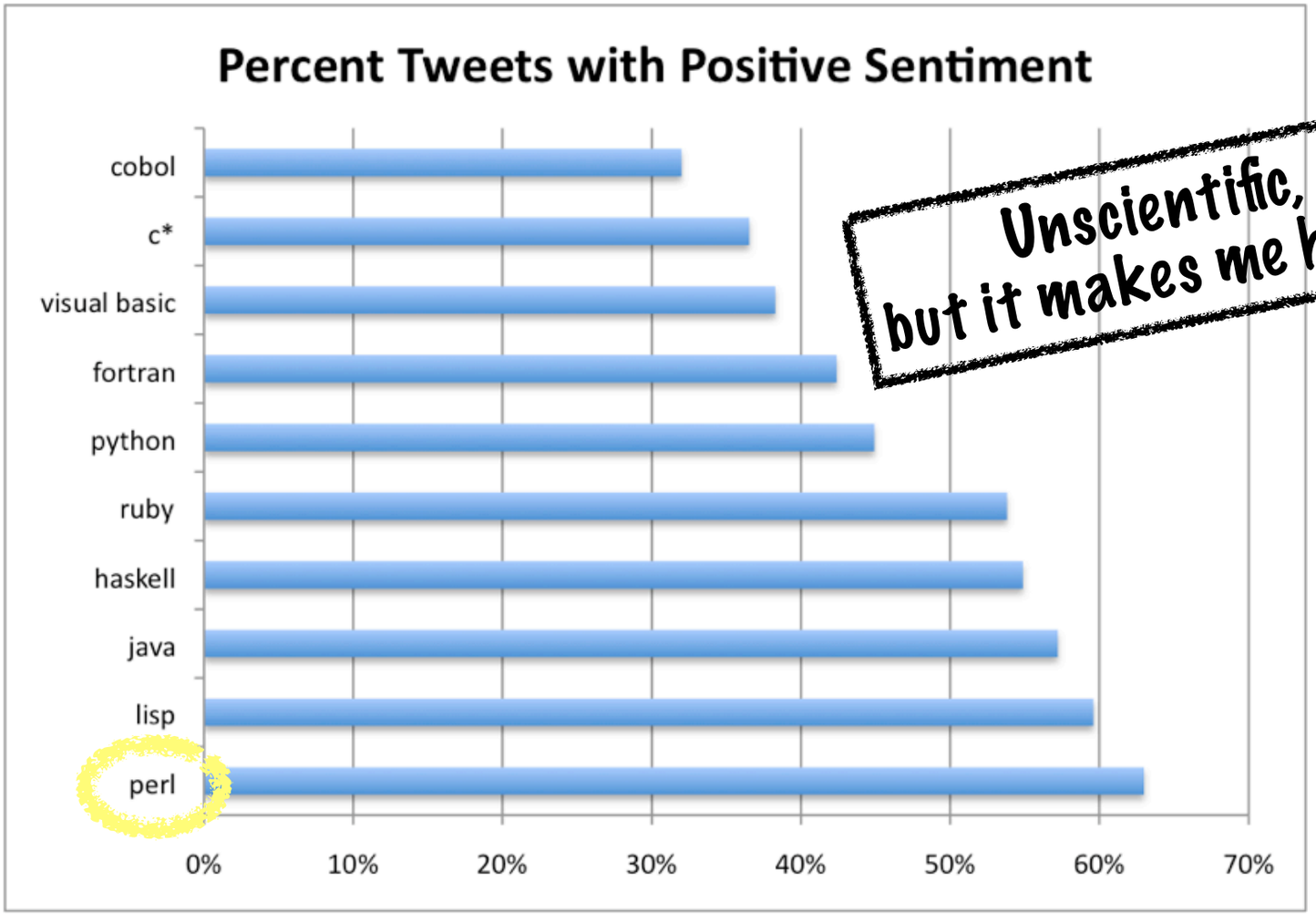
perlmonks.org - Meditations and wisdom

use.perl.org - News and blogs

irc.perl.org - Real-time help and chat

Quote taken from comment on http://blog.timbunce.org/2008/02/14/perceptions-of-perl-views-from-the-edge/

Perl Makes You Happy!



Source http://blog.doloreslabs.com/2009/05/the-programming-language-with-the-happiest-users/

Based on 150 tweets (per language?) on a particular day.

Perl 6

I'm a Second System.
What's your problem?

Meet Camelia, the Perl 6 logo

Perl 6 Myths

“Perl 6 is not Perl”

“It'll never be finished”

“There's no code written in Perl 6”

“Perl 6 is not Perl”

What makes Perl Perl?

Learn it once, use it many times. Learn as you go. Many acceptable levels of competence. Multiple ways to say the same thing. No shame in borrowing. Indeterminate dimensionality. Local ambiguity is okay. Punctuation by prosody and inflection. Disambiguation by number, case and word order. Topicalization. Discourse structure. Pronominalization. No theoretical axes to grind. Style not enforced except by peer pressure. Cooperative design. ``Inevitable'' Divergence.

Natural Language Principles in Perl

http://www.wall.org/~larry/natural.html

These principles are the essence of Perl, and haven't changed.

“Perl 6 [...] is clearer, more direct, more expressive, and without many of the old false leads and rough edges in Perl 5.

[...] a programmer looking at a Perl 6 program will instantly recognize that it is "Perl"”

—Larry Wall

```
my @suits  = < ♣ ♢ ♡ ♠ >;
my @ranks  = 2..10, < J Q K A >;

# concatenate each rank with each suit
my @deck = @ranks X~ @suits;

# create hash of card to points value
my %points = @deck Z (   (2..10, 10, 10, 10, 11)
                      X+ (0,0,0,0) );

# grab five cards from the deck
my @hand = @deck.pick(5);

# display my hand
say ~@hand;

# tell me how many points it's worth
say [+] %points{@hand};
```

A very simple example showing some very basic features.
Things that were fiddly in Perl 5 are trivial to express in Perl 6.

Perl 6 will run your Perl 5 code

Will have a choice of runtime compilation or once-off source code translation.

“It’ll never be finished”

What does finished mean?
Is Perl 5 finished?

"*If we'd done Perl 6 on a schedule, you'd have it by now. And it would be crap.*"

—Larry Wall

"*do it right*" and "*it's ready when it's ready*"

"*Truly radical and far-reaching improvements over the past few years.*"

It's given the design team the freedom to look deeply into issues.

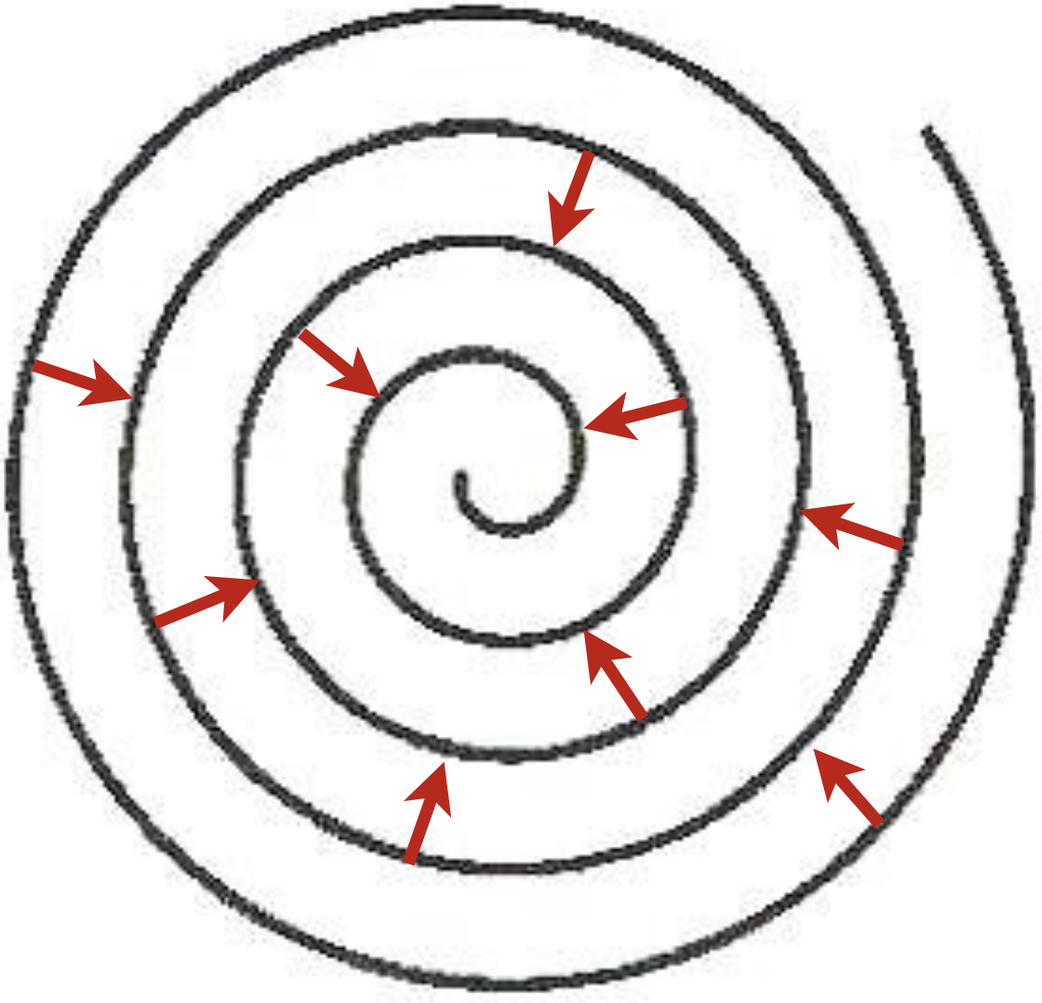
To re-balance the design and grammar as it evolved, in ways that wouldn't be possible after a release.

When will it be done?

"*We're not doing the Waterfall [model of development] we're doing the* ***Whirlpool****, where the strange attractor whirls around* ***with feedback at many levels*** *but eventually converges on something in the middle.*"

—Larry Wall

in 'What criteria mark the closure of perl6 specification'



“feedback at many levels”

“*In other words, a whirlpool sucks, but the trick is to position your whirlpool over your intended destination, and you'll eventually get there, though perhaps a bit dizzier than you'd like.*"

-- Larry Wall, in 'What criteria mark the closure of perl6 specification'

Multiple Implementations

Perl 6 compilers:

Pugs - initial experimentation in Haskell

KindaPerl6 - perl5 - aiming to self-host

Mildew - built on *SMOP* runtime, C non-VM

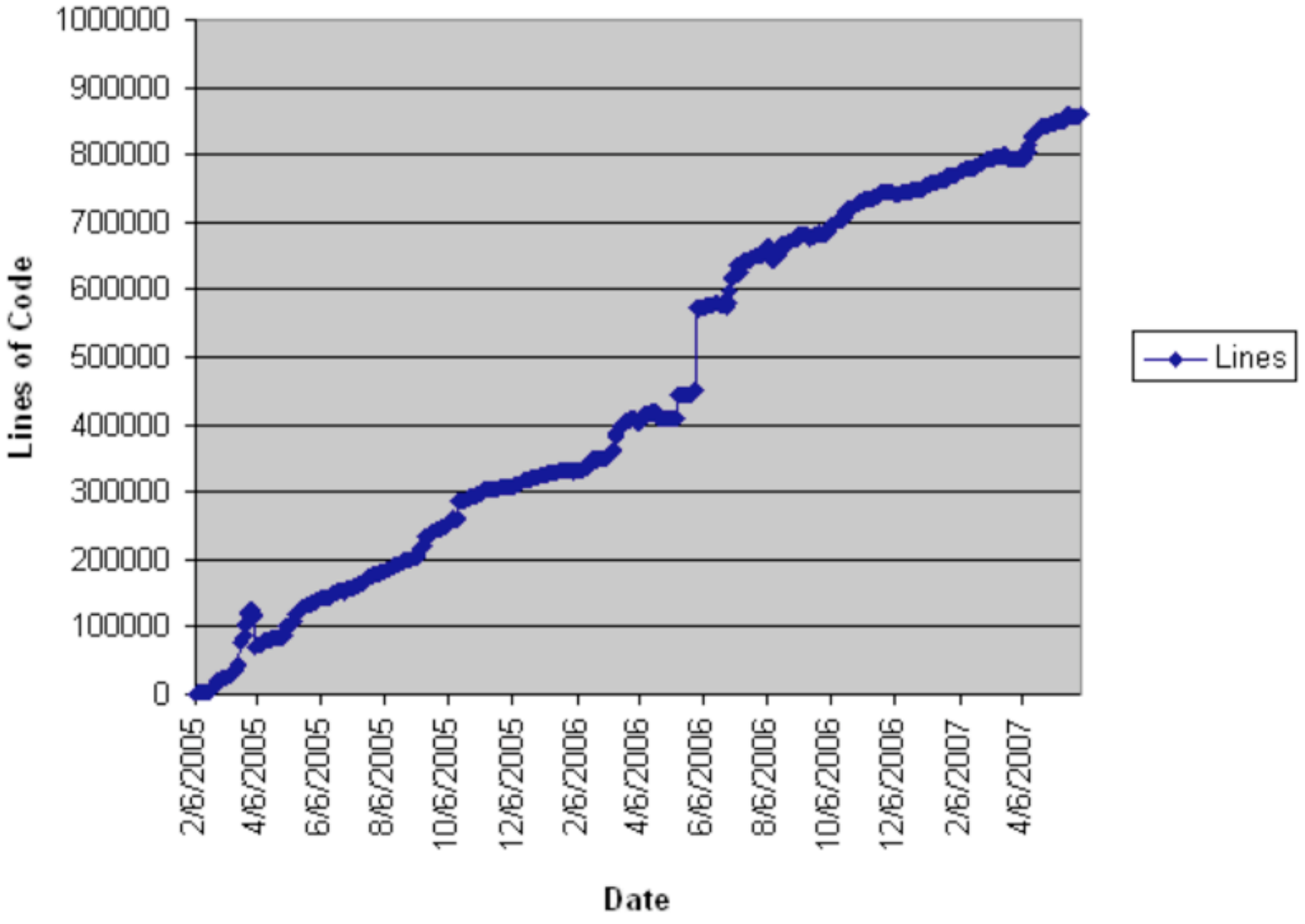
Elf - perl6 compiler written in perl6

Rakudo - built on Parrot Compiler Toolchain

All sharing a *common test suite*

Mildew & SMOP: http://www.perlfoundation.org/perl6/index.cgi?smop
Rakudo: http://rakudo.org

Pugs - 850,000 Lines of Code in 2 Years

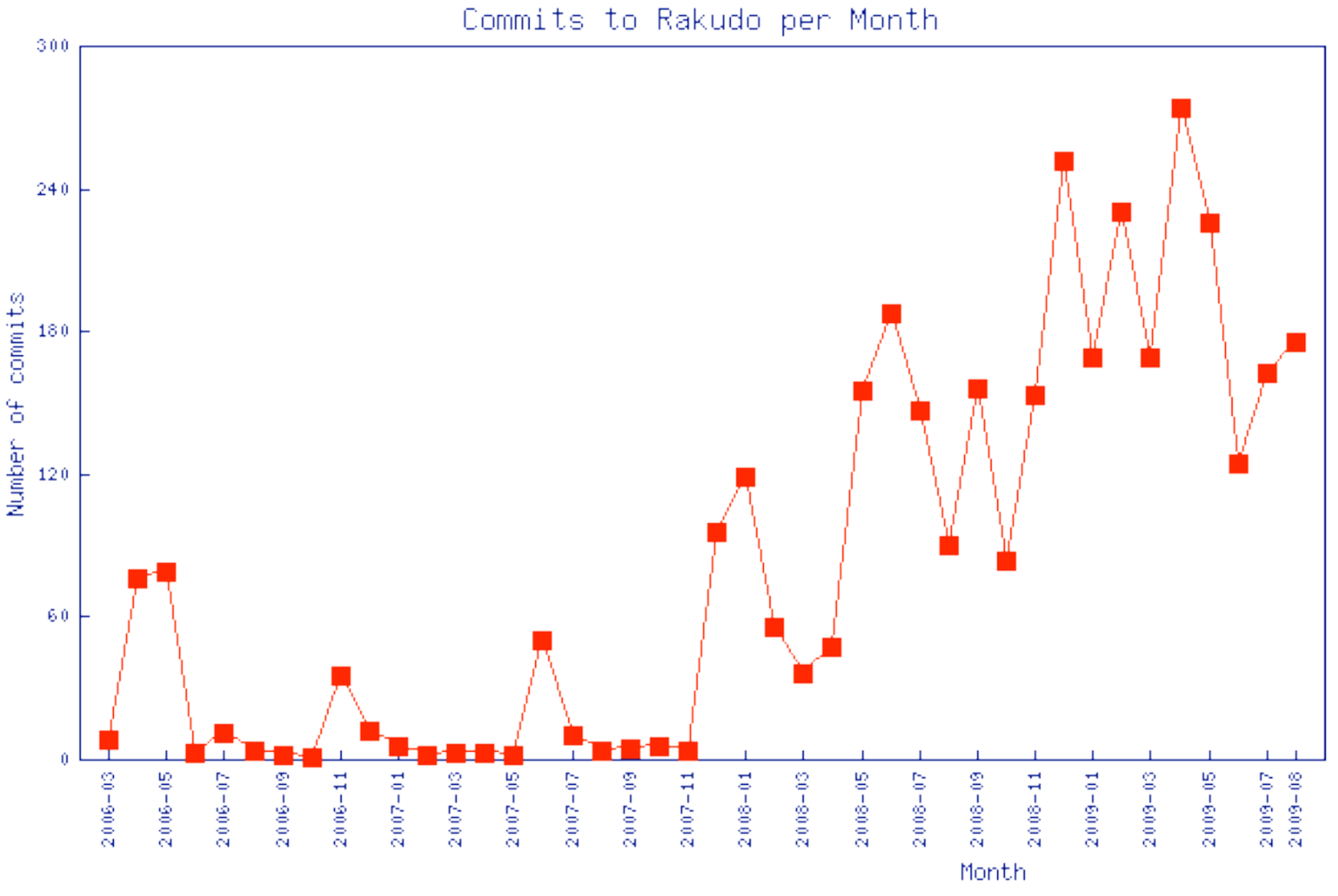


An extraordinary amount of work.

Valuable feedback into the ‘whirlpool’ of design and evolution.

Test suite is the official Perl 6 test suite

Rakudo Development Takes Off after Pugs



The next spin of the whirlpool
http://moritz.faui2k3.org/tmp/commits.png
generated by tools/commit-stats.pl in the Rakudo repository.

http://www.sgeier.net/fractals/indexe.php

“There's no Perl 6 code”

Perl 6 Projects

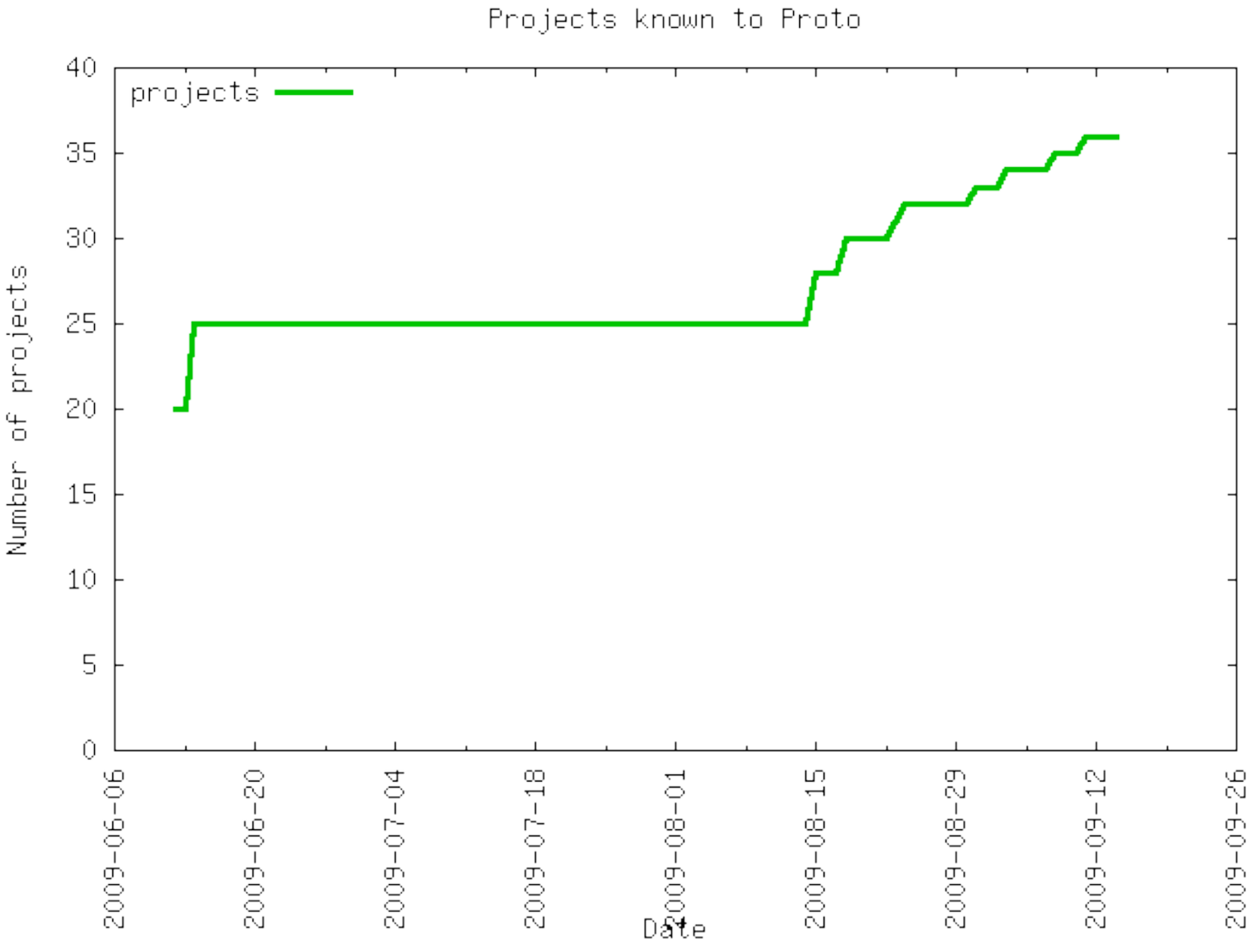
Perl 6 is starting to get jobs done:

- HTTP::Daemon - web server
- mod_perl6 - apache module
- Perl6::SQLite - database
- SVG::Plot - generate charts in SVG
- Web - web application framework
- November - a wiki

See http://perl6.org for more

Also http://github.com/perl6/perl6-examples/

Implemented in Perl 6



proto - a lightweight installer for perl6 projects http://use.perl.org/~masak/journal/38876
http://github.com/masak/proto/blob/master/projects.list

Coded in Perl 6

>35,000 lines of tests

>10,000 lines of examples

>10,000 lines in the Rakudo compiler

>1,600 lines in the November wiki

(figures from mid-2008)

(line counts don't mean much for Perl 6)

Rakudo

Perl 6 on Parrot

"Rakudo" short for "rakuda-do"

Japanese for "Way of the Camel"

"rakudo" also means "paradise"

The most advanced Perl 6 implementation

An advanced virtual machine designed for dynamic languages

Register based with Continuation Passing control flow

Already supports over 50 languages about 20 of which are non-toy

Python, Ruby, PHP, Lua, Lisp, TCL, ...

Many innovative features.
Will use LLVM for JIT.

Parrot Compiler Toolkit

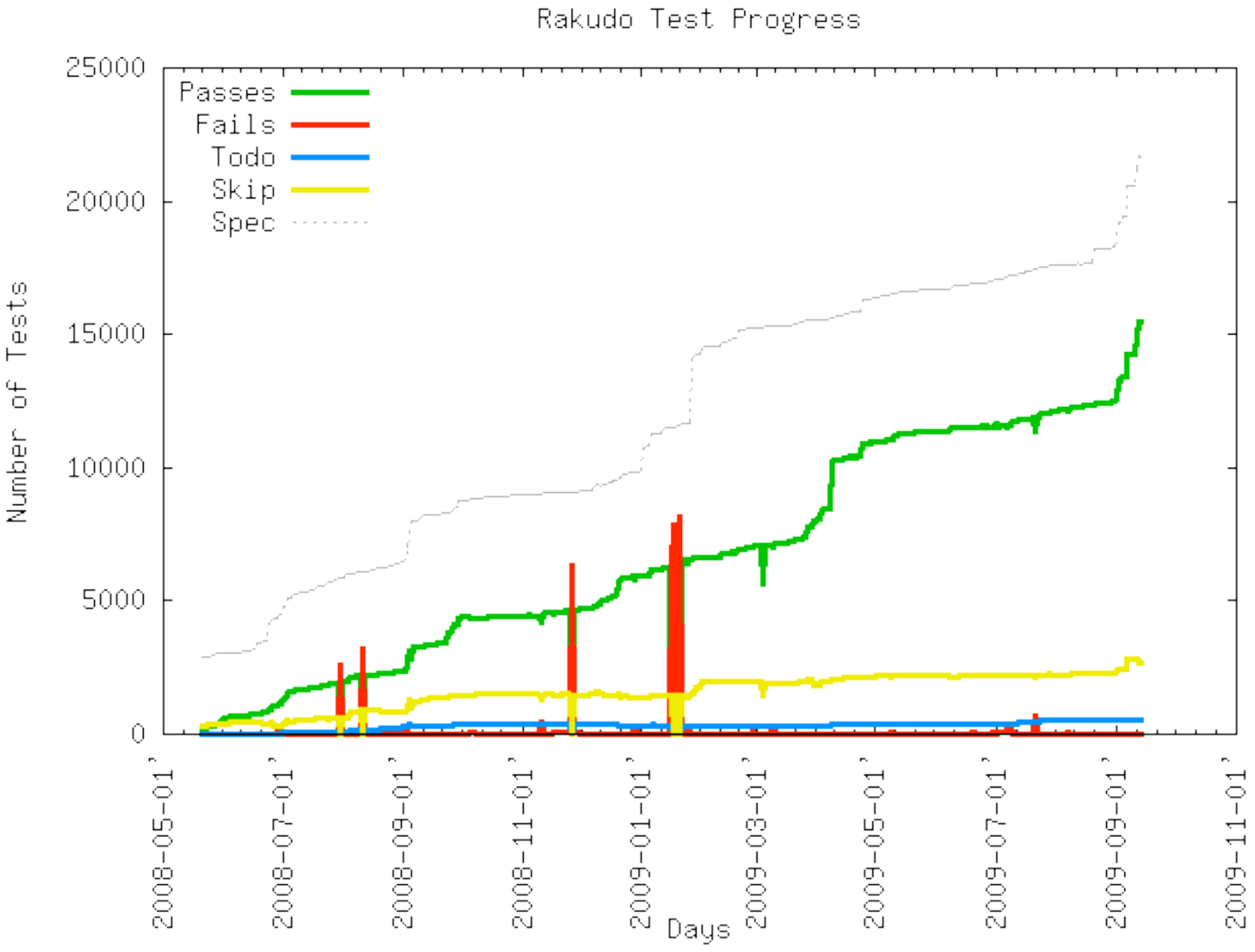
Write a compiler in an afternoon!

"Parrot is really quite wonderful. [...] Parrot lets you implement your own languages using Perl 6 rules for the grammar and Perl 6 for the compiler."

- Simon Cozens

http://blog.simon-cozens.org/post/view/1323

Rakudo Progress



Over 15,000 tests pass. Great progress in a little over a year.
Note that the number of tests keeps increasing.
http://rakudo.de/ and http://rakudo.org/status

“Rakudo Star”

Not “finished”

Not “Perl 6.0.0”

but a “useful” and “usable” release of Perl 6

in “Spring 2010”

http://use.perl.org/~pmichaud/journal/39411
A star is rising!

In Summary...

Perl

has a massive library of reusable code

has a culture of best practice and testing

has a happy welcoming growing community

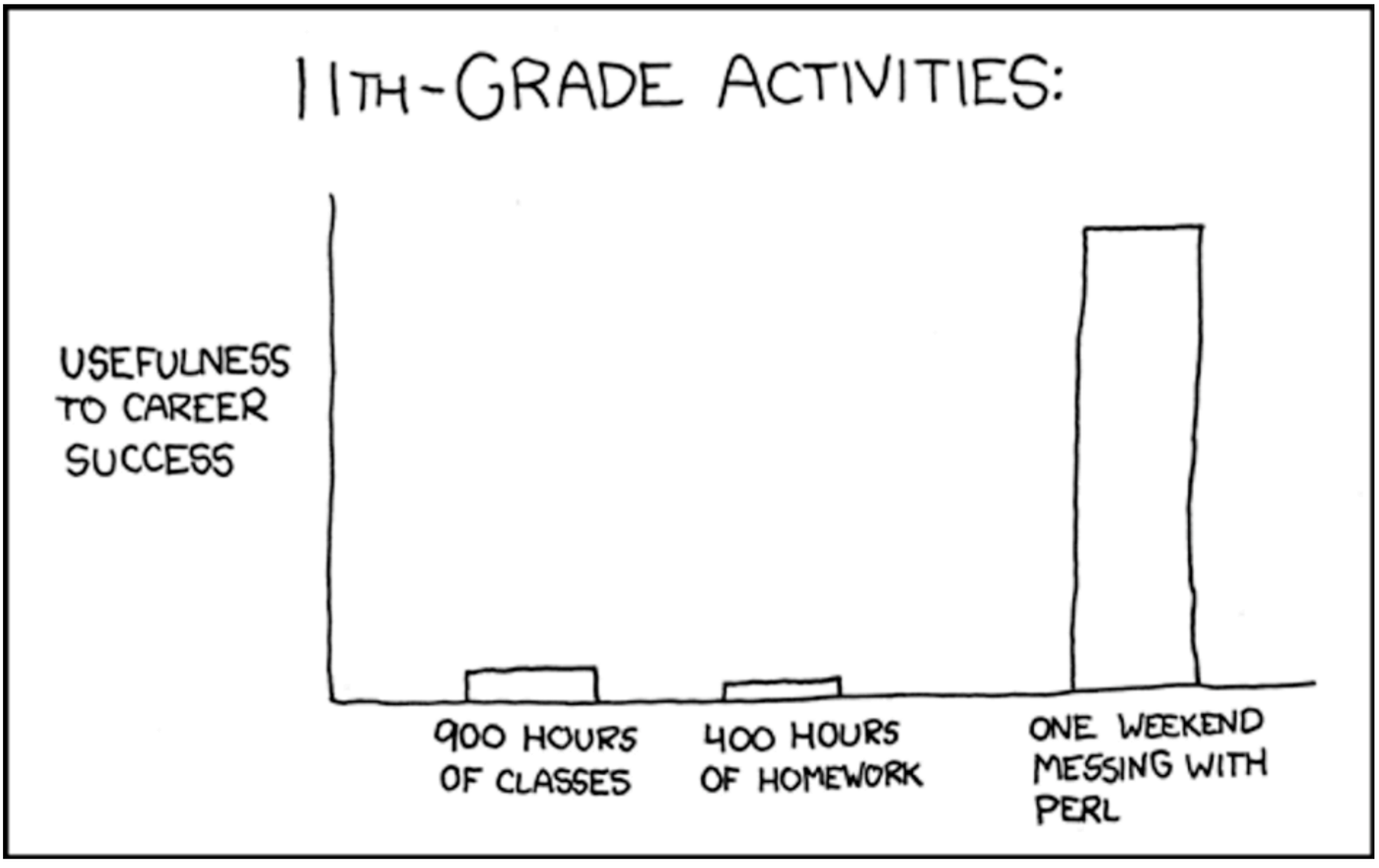
has a great future in Perl 5 and Perl 6

is a great language for *getting your job done*

for the last 20 years, and the next 20!

Any questions?

http://perlmonks.org
http://search.cpan.org
http://rakudo.org/how-to-help
http://blog.timbunce.org



http://xkcd.com/519/