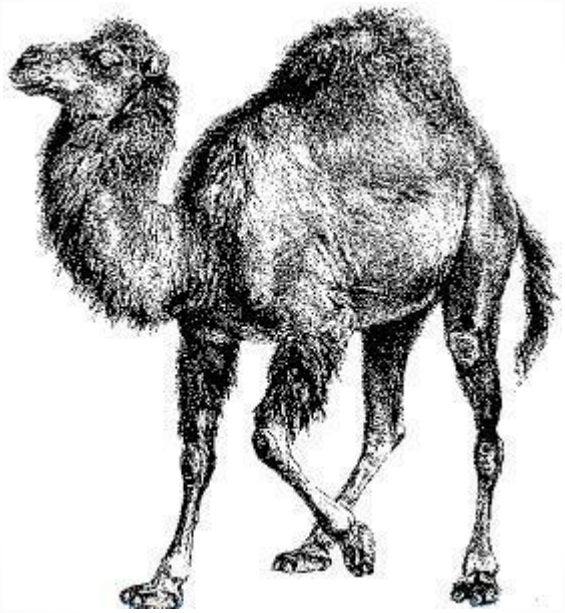
Докладчик:
Дмитрий Шаматрин. @justnoxx



Современная Web-Разработка на Perl. PSGI.

mod_perl, а надо ли?

Учи Perl!

Хронология событий

18.12.1987. Perl 1. Начало.

Хронология событий

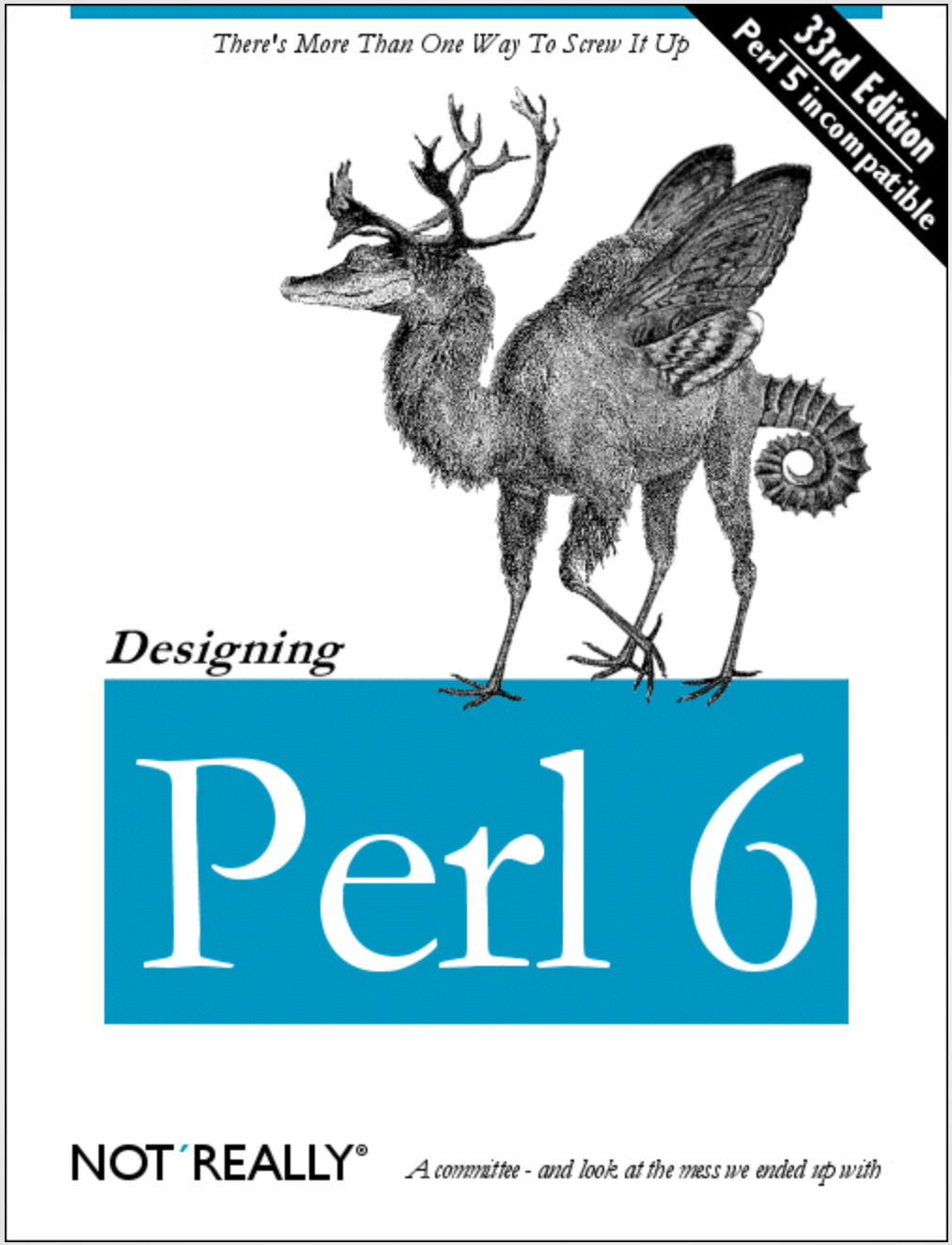
Perl 2. 1988

Perl 3. 1989

Perl 4. 1999

Perl 5.

17.10.1994


There's More Than One Way To Screw It Up
33rd Edition
Perl 5 incompatible
Designing
Perl 6
NOT REALLY®
A committee - and look at the mess we ended up with

```
echo "test... test... test..." | perl -e '$??s:;s:s;;$?::s;;=]=>%-{<-|}<&|`{;;y; -/:-@[-`{-};`-{/" -;;s;;$_;see'
```

```
`$=`;$_=\%!;($_)=/(.)/;$==++$|;($.,$/,$,,$\,$",$;,$^,$#,$~,$*,$:,@%)=(
$!=~/(.)(.
$_++;$_++;
;$,++;$^|=
```

```
#!/usr/bin/perl
print "Just another Perl hacker.\n";
```

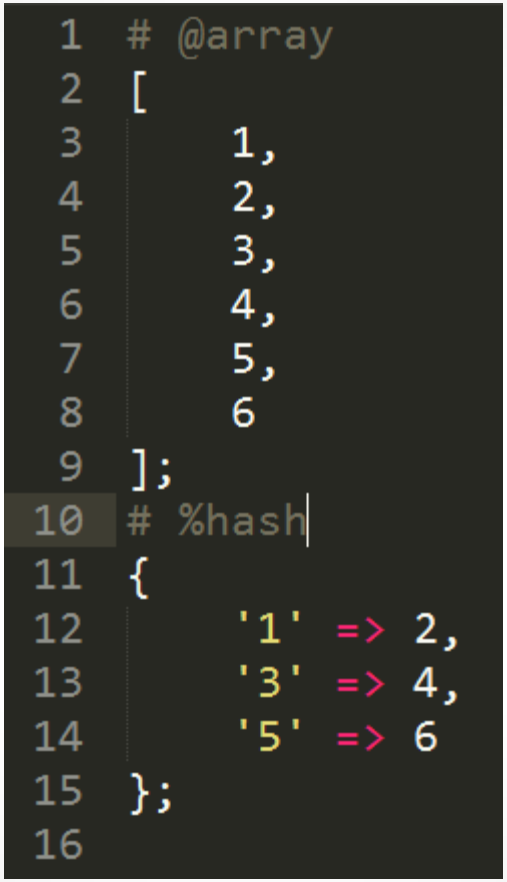
```
;$.++;$.++;
ᵕ,$#,);$,++
^$~$*.>&$=`
```

@rrays, $calars, %ashes

```
1  #!/usr/bin/perl
2  use strict;
3  my @var;
4  my %var;
5  my $var;
6  @var = 1..10;
7  %var = @var;
8  $var = \%var;
```

И еще один пример

```
1  #!/usr/bin/perl
2  use strict;
3  my @var;
4  my %var;
5  my $var;
6  @var = 1..10;
7  %var = @var;
8  $var = \%var;
```


1 # @array
2 [
3 1,
4 2,
5 3,
6 4,
7 5,
8 6
9];
10 # %hash
11 {
12 '1' => 2,
13 '3' => 4,
14 '5' => 6
15 };
16

```
1  #!/usr/bin/perl -w
2  use strict;
3  use Data::Dumper;
4  my @array = (1, 2, 3, 4, 5, 6);
5  my %hash = (1, 2, 3, 4, 5, 6);
```

Функции (subroutines).

```
1  sub my_sub {
2      my $param = shift;
3      print $param;
4  }
5  sub my_sub2 {
6      my ($param) = @_;
7  }
```

```
1  #!/usr/bin/perl -w
2  use strict;
3  $\ = "\n";
4  my @array = qw/1 2 3 4 5 6/;
5  foreach my $val (@array) {
6      print $val;
7  }
```

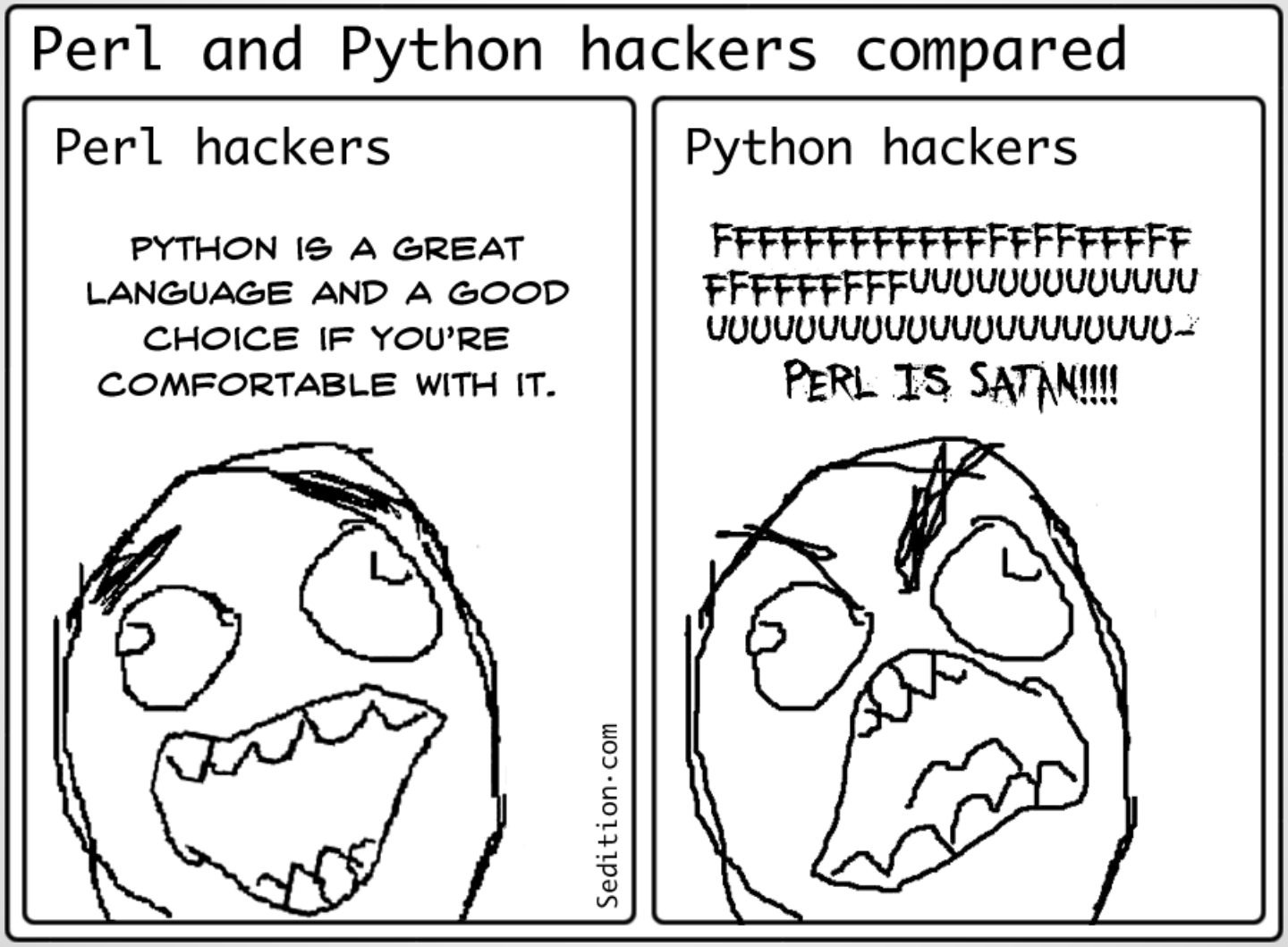
Ссылки

```perl
#!/usr/bin/perl -w
use strict;
my %hash = (
    1   =>  2,
    3   =>  4,
    5   =>  6
);
print $hash{3}; # 4
# or my $hash = \%hash;
my $hash = {
    1   =>  2,
    3   =>  4,
    5   =>  6
};
print $hash->{3}; # 4

my @array = (1, 2, 3, 4, 5, 6);
print $array[1]; # 2
# or my $array = \@array
my $array [1, 2, 3, 4, 5, 6];
print $array->[1]; # 2
```


PERL INTERVIEW
HOW WOULD YOU WRITE THIS REGEX
... USING LESS THAN 12 CHARACTERS
toonlet.com/creator/dalmaer
10 feb 08, 12:19 AM

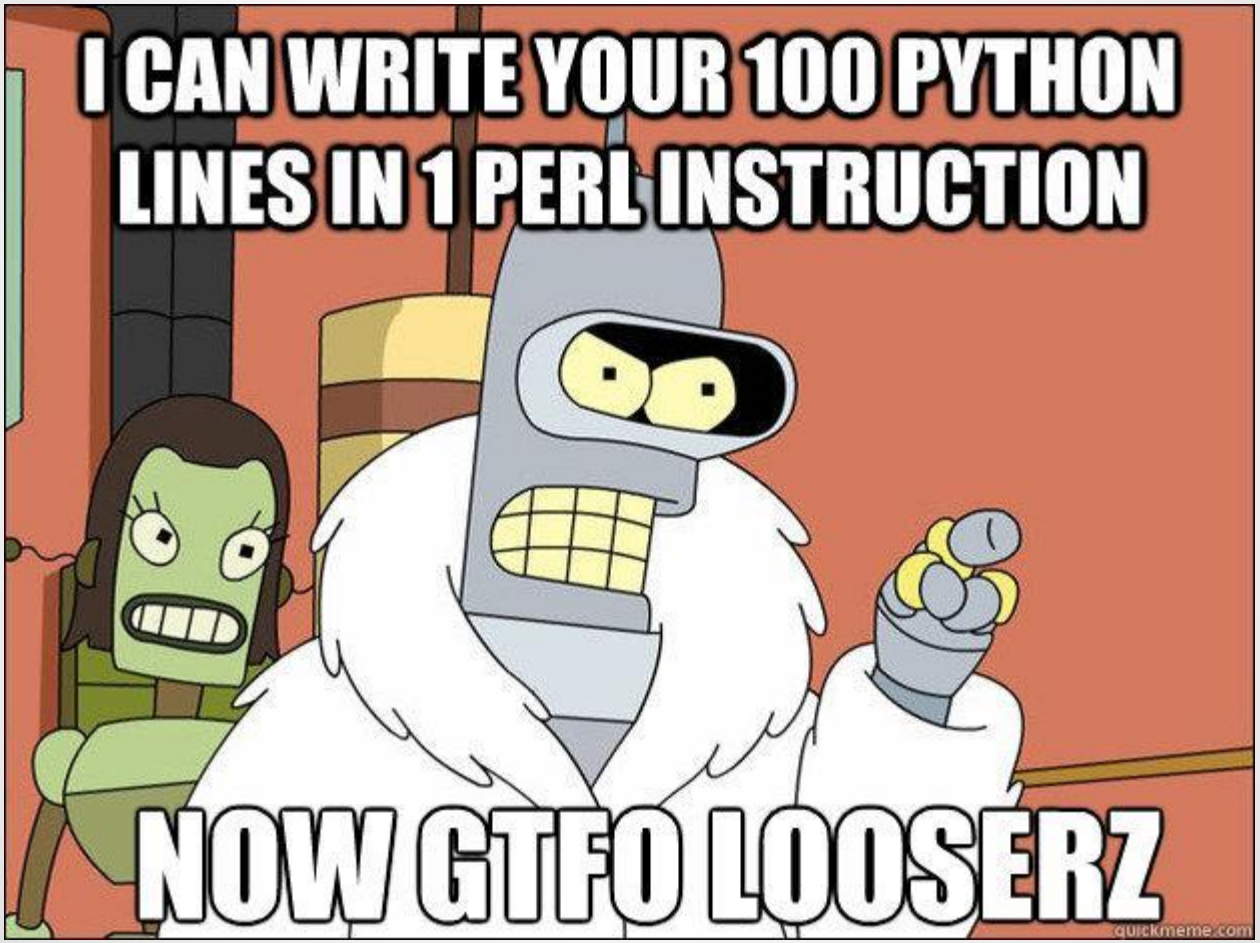
У Perl есть проблема.



Это действительно проблема.

А вот ее следствие

Ho


I CAN WRITE YOUR 100 PYTHON LINES IN 1 PERL INSTRUCTION
NOW GTFO LOOSERZ
quickmeme.com

Python

Python,
Я хочу сыграть с тобой в игру!

Поиграем в сравнение

Сейчас мы сравним Perl с Python в контексте разработки под Web.

Python

Для тестирования Python мы воспользуемся Apache2 со следующими настройками:

```
ScriptAlias /cgi-bin/ /var/www/cgi-bin/
<Directory "/var/www/cgi-bin">
    AllowOverride None
    Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
    Order allow,deny
    Allow from all
    AddHandler cgi-script .py
    AddHandler default-handler .html .htm
</Directory>
```

Python

А код у нас будет выглядеть так:

```
#!/usr/bin/python
import cgi
import cgitb
print "Content-Type: text/html; charset=UTF-8"
print ""
print '''
hello world!
'''

```

Perl

В качестве сервера мы будем использовать Feersum, а код у нас будет выглядеть так:

```
1  my $app = sub {
2      return [200, [], ["\nHello world\n\n"]];
3  };
```

Проведем же замеры!

В первую очередь воспользуемся time + curl.

Результат:

	real	user	sys
Python	0m0.356s	0m0.000s	0m0.056s
Perl	0m0.117	0m0.000s	0m0.056s

Тестовая машина - Intel Atom 1.2 ГГц, 1 ГБ ОЗУ

Следующий наш ход - ab

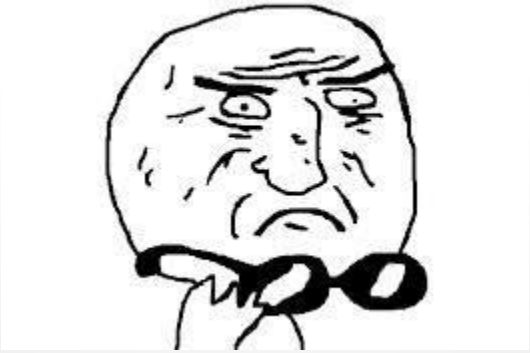
Проверка будет производится следующей командой:

ab -n 1000 url

Результаты

	Python	Perl
Document Length	15	15
Time	167.172 seconds	2.813 seconds
RPS	5.98	355.50 rps
TPR	167.172 ms	2.813 ms

Perl быстрее в 59.448160535 раз!




PERL
THE ORIGINAL REVOLUTIONARY LANGUAGE

Понравился бенчмарк?

Вот именно так и выглядят большинство сравнений Perl и Python.

Perl. Современная web-разработка.

Фреймворки?

1. Dancer
2. Mojolicious
3. Amon2
4. **Plack**

Шаблонизаторы?

1. Template::Toolkit
2. HTML::Template
3. HTML::Template::Compiled
4. Mason
5. **Text::Xslate**

Асинхронность?

1. **AnyEvent**
2. Coro
3. POE
4. EV

PSGI Серверы?

1. **Feersum**
2. Twiggy
3. Starman
4. Corona
5. Starlet
6. Twiggy::TLS
7. Monoceros
8. Twiggy::Prefork

PSGI?

PSGI или **Perl Web Server Gateway Interface** - это спецификация, предназначенная для отделения среды веб-сервера от кода веб-фреймворка. PSGI не является программным интерфейсом (API) для веб-приложений.

PSGI сервер - это программа на Perl, предоставляющая среду для запуска в ней PSGI приложения. Его часто называют PSGI Application Container, так как он похож на Java Servlet container, который представляет собой Java-процесс, предоставляющий среду для сервлетов Java.

PSGI приложение

Пример простейшего PSGI приложения мы видели раньше. Вот оно:

```
1  my $app = sub {
2      return [200, [], ["\nHello world\n\n"]];
3  };
```

Plack & Starman

```perl
#!/usr/bin/perl
use strict;
use Plack;
use Plack::Request;
use Plack::Builder;
my $app = sub {
    my $env = shift;
    my $req = Plack::Request->new($env);
    my $res = $req->new_response(200);

    my $params = $req->parameters();
    my $uploads = $req->uploads();
    $res->header('Content-type' => 'text/html; charset=utf-8');
    $res->body('Hello, I am Plack!');
    return $res->finalize();
};

my $main_app = builder {
    mount "/" => builder {$app};
};
```

Я бы с радостью рассказал очень много всего приятного про Plack, но, увы, время доклада ограничено.

Более подробно про Plack можно почитать на http://pragmaticperl.com

Если у вас появятся вопросы, задавайте их в любое время.

Спасибо за внимание, надеюсь, было интересно.