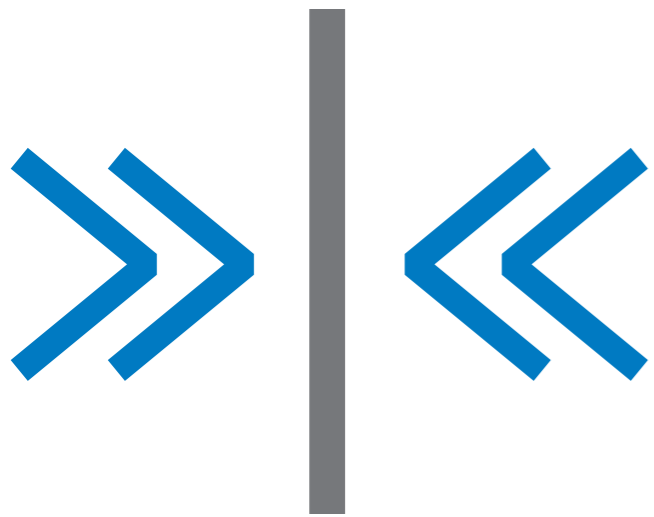


# Perl 6 Updates

Chia-liang Kao  
COSCUP 2008 .Taipei

高嘉良



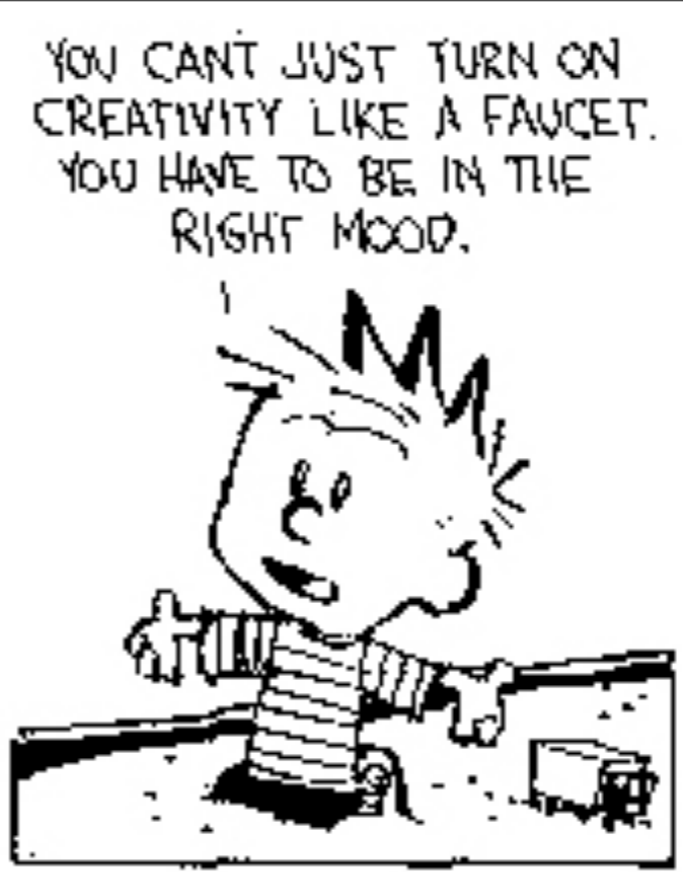
BEST  
PRACTICAL™



愛印網

# 8/23, COSCUP

- 忙著吃點心、聊天
- 只有四頁投影片
- PANIC!



Perl

在去去年過  
了  
二十歲生日

# 第一紀元

- Perl 1.0: 1987 冬
  - `@array`, `%hash`, `$scalar`, ``/regex/``
  - `sub name { ... }, ('list'), <FILE>`
- Perl 2.0: 1988 夏
  - `foreach`, `sort`, `do "file"`
  - `perl -w`, `local($var)`

# 第一紀元

- Perl 3.0: 1989 秋
  - GPL, pack/unpack
  - &sub, undef
  - package, "interpolated \$s @a", sockets
- Perl 4.0: 1991 春
  - "Programming Perl"
  - Artistic License
  - caller, qx//, 0 + @array

# 第二紀元

- Perl 5.0: 1994 冬
  - OO, POD, `my($var), $Package::var`
  - `use Module, (name => "value")`
  - `tie($scalar)`
- Perl 5.1: 1995 春
  - `sub { ... }`
  - `$SIG{__WARN__}, $SIG{__DIE__}`

# 第二紀元

- Perl 5.2: 1996 春
  - `sub name ($$;$) { ... }`
  - `use overload`
- Perl 5.4: 1997 春
  - `$code->(@args), use 5.004;`
  - `UNIVERSAL::isa()`, `UNIVERSAL::can()`

# 第三紀元

- Perl 5.5: 1998 夏
  - `B::*`, `threads`
  - `tie(@array)`
- Perl 5.6: 2000 春
  - `use utf8`
  - `use warnings`
  - `open $fh`

# 第三紀元

- Perl 5.8: 2002 秋
  - Encode, PerlIO, iThreads
  - Test::More
- Perl 5.10: 2007 冬
  - 引入 Perl 6 語法
    - `use features qw(switch say);`
    - `$a // $b; $c ||= $d`
    - `s/(?<letter>.)\k<letter>/${letter}/g`
  - CPANPLUS
  - lexical pragma

# Perl 6: 以簡馭繁

```
say "Hello, World!" if $x == any(1..3);  
sub unfinished { ... }
```

# Perl 6: 自由

- 給程式員最大的自由
- 要靠習俗, 不設定律
- 保持語言的彈性
- 隨時反應環境的需求
  - 可變式剖析器
  - 共存的方言架構
  - 多樣文件格式
  - 使用者自定的 "核心功能"
- 因應二十年以上的長程需求

# Perl 6: 變數符號

- 符號成為變數名稱的一部份

```
$scalar, @array, %hash  
@array[3]          # 不再是 $array[3]  
%hash{'key'}      # 也可以寫成 %hash<key>
```

- 因此陣列和雜湊也可以當物件用了!

```
%hash.keys  
@array.sort
```

# Perl 6: 點號

- 常用的小箭頭縮短成點號了

```
$obj.method()
```

```
$a_ref.[1]
```

```
$h_ref.<key>
```

```
$c_ref.()
```

```
# 也可以省略成 $ref[1]
```

```
# $h_ref<key>, $h_ref{'key'}
```

```
# $c_ref()
```

# Perl 6: 具名參數

- 原來的寫法依然適用

```
sub sum {  
    my $sum;  
    $sum += $_ for @_  
    return $sum;  
}
```

- 新增各種具名參數

```
sub clean ($text, $method) { ... } # 傳址  
sub by_value ($text is copy) { ... } # 傳值  
sub some_opt ($req, ?$opt = $req) { ... } # 選用  
sub modify ($text is rw) { ... } # 可讀寫  
sub typed (Int $num, Str $txt) { ... } # 具型別
```

# Perl 6: 超維算符

- 雙向超維

```
(1, 1, 2, 3, 5) »+« (1, 2, 3, 5, 8); # (2, 3, 5, 8, 13)
```

- 單向超維

```
@objects ».run();
```

- 自動昇維

```
('a'..'c') »x« 3; # ('aaa', 'bbb', 'ccc')
```

# Perl 6: 類別

- 類別宣告

```
class Tree { method nodes { ... } }
```

- 繼承

```
class Leaf is Tree {  
    has Tree $.val;  
    method nodes { .val }  
}
```

# Perl 6: 文法

- 文法/規則 ::= 物件/方法
- 給規則一個命名空間

```
grammar URI {  
    rule reserved    { <[;/?:@&=+$/, \[ \ ]> };  
    rule mark        { <[-_!.~*' ()]> };  
    rule unreserved  { rule { <[A-Za-z0-9]+<mark>> };  
    rule scheme      { <[a-zA-Z]><[a-zA-Z0-9.+ -]>* };  
    rule uri         { <+<reserved>+<unreserved>+[ "% " ]> };  
}
```

# Perl 6: 文法

- Perl 6 本身也是一個文法!

```
grammar Perl6 {  
    rule statement { ... }  
    rule identifier { ... }  
}
```

# Multiple Implementations

- Pugs - Haskell implementation
- v6 - Compiles to Perl5
  - on CPAN today! use v6;
- Rakudo - On Parrot
- SMOP - C runtime
- Multiple approaches to bootstrap - to write Perl6 in Perl6

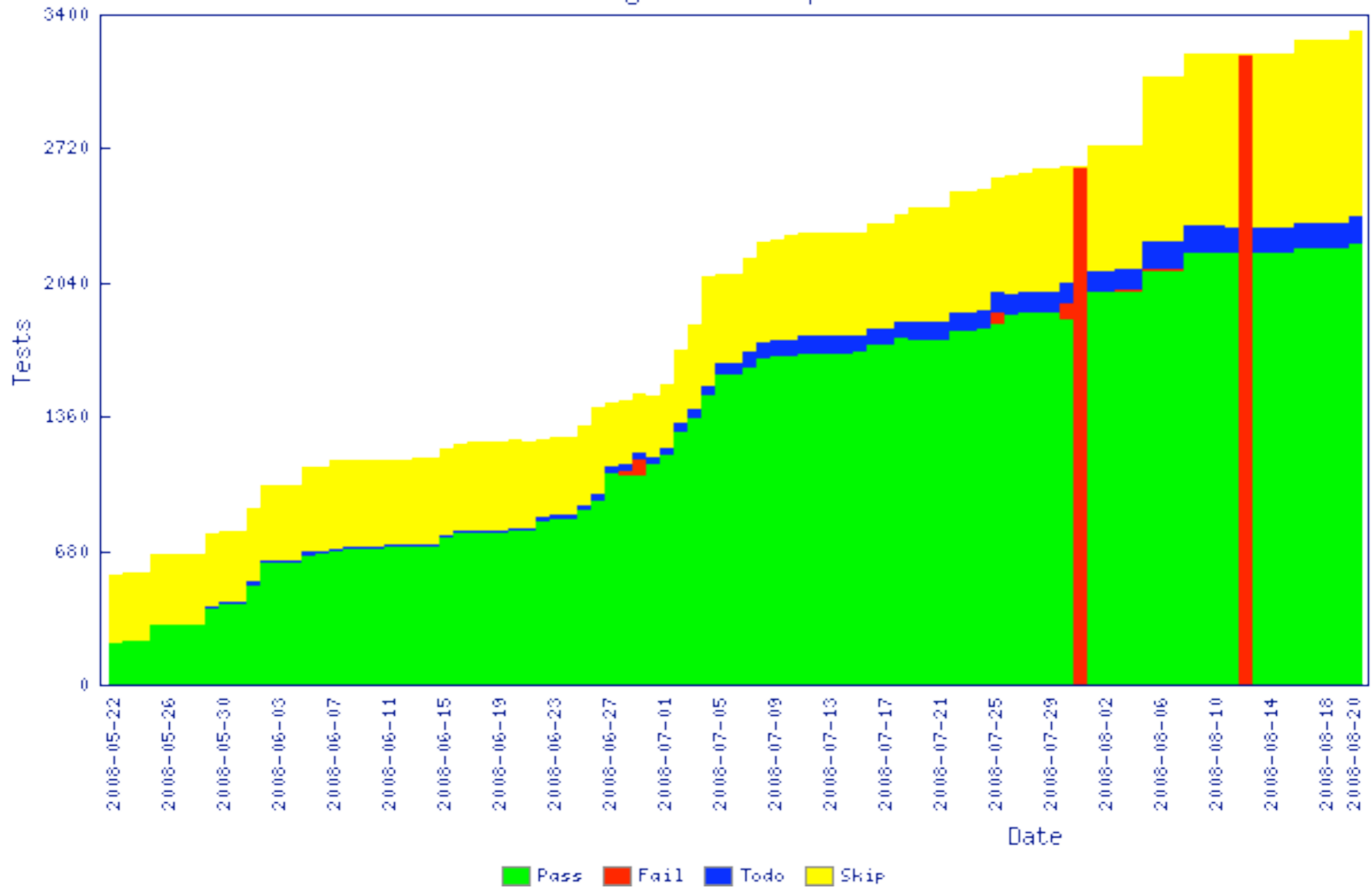
# STD.pm

- The Perl6 Grammar of Perl6
- parses itself!
  - `gimme5` compiles `STD.pm` to `STD.pmc`
  - `try5` tries to use the rules provided by `STD.pmc` to parse Perl6 expressions
- now parses all the tests too!

# Tests

- 20,000 unit tests
- 5000 or so spectests reviewed and categorised by the parrot team

# Passing Rakudo Spectests



# Perl6 對 Perl5 的影響

- Perl 5.10 的新功能

- say

- switch

```
given ($variable) {  
    when (1) {} # $variable == 1  
  
    when (@edge_cases) {} # if $variable in  
@edge_cases {}  
  
    when ($_ > 10e6) { } # if $variable is over  
a million  
}
```

# Perl 5.10 的新功能

- named captured in regex

```
s/(?<letter>.)\k<letter>/${letter}/g
```

Named



Backtracking



Captured



# Perl6 對 Perl5 的影響

- Moose - 後現代物件導向系統

```
package Point;
use Moose; # 自動打開 strict and warnings

has 'x' => (is => 'rw', isa => 'Int');
has 'y' => (is => 'rw', isa => 'Int');

sub clear {
    my $self = shift;
    $self->x(0);
    $self->y(0);
}
```

# Moose

- 繼承

```
package Point3D;
use Moose;

extends 'Point';

has 'z' => (is => 'rw', isa => 'Int');

after 'clear' => sub {
    my $self = shift;
    $self->z(0);
};
```

# Moose - Meta Objects

- 完整的 meta-object

```
my $metaclass = Moose::Meta::Class>create(  
  'New::Class', roles => [...] );
```

```
my $metaclass = Moose::Meta::Class->create_anon_class(  
  superclasses => ['Foo'],  
  roles        => [qw/Some Roles Go Here/],  
  cache       => 1,  
);
```

# Perl6 對 Perl5 的影響

- Perl6::Declare, or prototype::signatures

```
use Perl6::Declare;
```

```
sub6 fnord(Int $n, :$foo = 42, $raah?) {  
  
};
```

```
fnord(p(foo => 99, raah => 123), 100);
```

# When?

- By Christmas!
  - which?
  - “but once perl6 is released, everyday will be like christmas” - Audrey Tang
- But use the new features inspired by Perl6 in Perl5 today!

# 謝謝！

- #perl6 on irc.freenode.net