

# Què està passant a Perl 6

## Què està passant a Perl 6

La història de Perl, Rakudo i esdeveniments actuals

Tadeusz Sośnierz (autor original)  
Alex Muntada (traducció al català)

26 de juliol de 2010

# Per què parlem avui de Perl 6?

Larry Wall

«Perl 5 was my rewrite of Perl. I want Perl 6 to be the community's rewrite of Perl and of the community.»

# Per què parlem avui de Perl 6?

Larry Wall

«Perl 5 was my rewrite of Perl. I want Perl 6 to be the community's rewrite of Perl and of the community.»

- 19 juliol 2000 – anunci de Perl 6 – Fa exactament 10 anys!

# Per què parlem avui de Perl 6?

Larry Wall

«Perl 5 was my rewrite of Perl. I want Perl 6 to be the community's rewrite of Perl and of the community.»

- 19 juliol 2000 – anunci de Perl 6 – Fa exactament 10 anys!
- Perl 6 no és només una especificació

# Per què parlem avui de Perl 6?

Larry Wall

«Perl 5 was my rewrite of Perl. I want Perl 6 to be the community's rewrite of Perl and of the community.»

- 19 juliol 2000 – anunci de Perl 6 – Fa exactament 10 anys!
- Perl 6 no és només una especificació
- En 10 dies (29 juliol) sortirà Rakudo Star – una distribució de Perl 6 llesta per utilitzar

# Per què parlem avui de Perl 6?

Larry Wall

«Perl 5 was my rewrite of Perl. I want Perl 6 to be the community's rewrite of Perl and of the community.»

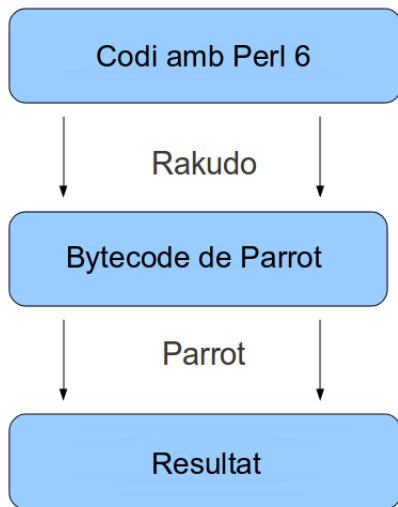
- 19 juliol 2000 – anunci de Perl 6 – Fa exactament 10 anys!
- Perl 6 no és només una especificació
- En 10 dies (29 juliol) sortirà Rakudo Star – una distribució de Perl 6 llesta per utilitzar
- Aquest dijous (22 juliol) sortirà Rakudo #31, en què es basarà Rakudo Star



» Ö «



# Rakudo i Parrot



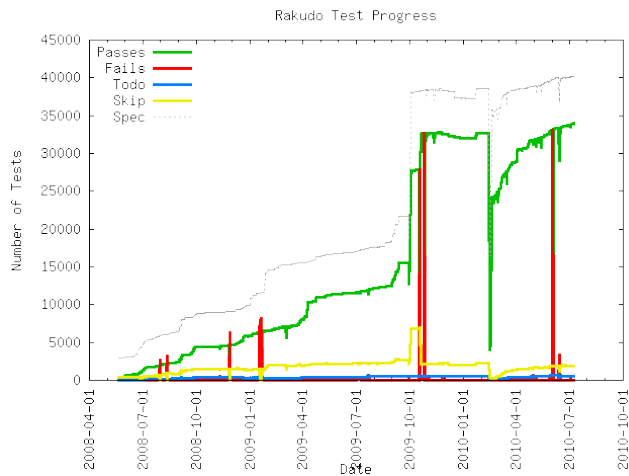


- Una màquina virtual per a llenguatges dinàmics
- Implementacions disponibles per a Perl 6, Python, Ruby i altres
- Surt regularment cada mes, demà (20.07) la versió 2.6.0



- (Jap.) «The Way Of The Camel», o «Paradise»
- Compilador de Perl 6 en Perl 6
- També publicat cada mes, dos dies després que surti Parrot
- Actualment compleix el 83% dels tests de les especificacions

# Rakudo – progrès



- «useful and usable»
- Serà una implementació completa, però segura d'utilitzar
- Per atraure l'atenció i encoratjar la portabilitat dels mòduls i la programació
- Data planificada – 29 juliol.

I juntament amb ell:

- Parrot
- Blizkost (en parlem tot seguit)
- Mòduls
  - Zavolaj (native call interface)
  - MiniDBI (subset DBI)
  - ...
- «The Perl 6 Book» (<http://github.com/perl6/book>)

# Blizkost

Permet la possibilitat d'utilitzar Perl 5 com un dels llenguatges a Parrot, que permet utilitzar mòduls de Perl 5 dins del codi en Perl 6

```
blikost/examples/cgi.pl
```

```
use v6;  
use CGI:from<perl5>;  
my $q = CGI.new;  
  
print $q.header, $q.start_html('Hello World'),  
      $q.h1('Hello World'), $q.end_html;
```

# Zavolaj

«Crida» una funció en C directament des de Perl 6

zavolaj/examples/sqlite3.p6 (fragments)

```
use NativeCall;
sub sqlite3_open( Str $filename, OpaquePointer $ppDB )
    returns Int
    is native('libsqlite3')
    { ... }

my OpaquePointer $db;
my $status = sqlite3_open("test.db", $db);
```

# Mòduls – proto, pls

- proto – instal·lador de mòduls de Perl 6
- projecte nou: pls (substitueix proto)
- <http://modules.perl6.org> – els mòduls base
  - Math::Model
  - MiniDBI
  - LWP::Simple
  - SVG
  - ufo
  - URI
  - XML, XML::Writer
  - Web
  - HTTP::Server::Simple, HTTP::Server::Simple::PSGI (no disponibles en proto)



# Altres implementacions

- PUGS (abandonat)
- YAPSI – Yet Another Perl Six Implementation  
(<http://github.com/masak/yapsi>)
- Niecza – compilador de Perl 6 en .NET/Mono  
(<http://github.com/sorear/niecza>)
- Bennu – compilador del Perl 6 per a LLVM  
(<http://github.com/ekiru/Bennu>)

Canvis, canvis, canvis

Què ha canviat al propi llenguatge?

# Què no agrada de Perl 5

- OOP en cru
- Accés als paràmetres de les funcions
- Capturar excepcions
- Una sintaxi no gaire maca de vegades (referències)

# Què ha canviat

- OOP – tot és un objecte, creant una nova sintaxi per a les classes (similar a la de MooseX::Declare)
- Expressions regulars, per una banda incompatibles amb les de Perl 5, però amb moltes més opcions
- Sintaxi – s'han simplificat algunes coses, però se n'han afegit moltes d'altres

Què ha canviat

Però això encara és Perl

# Quins mòduls ja no seran necessaris

- Moose
- Try::Tiny
- Data::Dumper – cada variable té un mètode .perl
- Devel::REPL – els REPL tenen un estàndard
- Getopt::\* – es poden detectar els paràmetres per a la funció MAIN (això en un moment)

# Comprovació de tipus

És possible assignar un tipus a una variable

```
> my Int $a = 5; $a = "foo"
```

Type check failed for assignment

```
> my Str $a = "foo"; $a = 5
```

Type check failed for assignment

# però...

```
> my Str $a = "foo"; $a = 5.Str; $a.perl.say  
"5"
```

```
> say ~[5.WHAT, 'string'.WHAT, (3/7).WHAT, /foo .*/.WHAT]
```

Int() Str() Rat() Regex()

# Tot és un objecte

```
> 'a string'.^methods.sort[40..45]  
bytes can capitalize ceiling chars chomp  
> "the big brown fox".split(' ').grep(/^b/).join(' and ')  
big and brown
```



# Prefixos de variables

A Perl 6 el prefix d'una variable (sigil) no canvia durant l'extracció d'un sol element

Perl 5	Perl 6
<pre>my @arr = (1, 2, 3) my \$elem = \$arr[1]</pre>	<pre>my @arr = (1, 2, 3) my \$elem = @arr[1]</pre>
<pre>my %hash = (foo =&gt; 'bar') my \$elem = \$hash{'foo'}</pre>	<pre>my %hash = (foo =&gt; 'bar') my \$elem = %hash{'foo'}</pre>

# Tot és una referència

No hi ha distinció entre variables i referències cap a elles, d'aquesta manera no hi ha problemes de dereferenciació.

```
my $a = [ { foo => [1, 2, 3] } ];
```

Perl 5

```
say $a->[0]->{foo}->[1] # 2
```

Perl 6

```
say $a[0]<foo>[1] # 2
```

# Funcions i paràmetres

```
sub foo (Str $what, Int $times = 1) {  
    say $what x $times  
}
```

```
foo "hello", 5; # hellohellohellohellohello
```

```
foo "hello";    # hello
```

```
foo;            # Not enough positional parameters passed;  
                # got 0 but expected between 1 and 2
```

# La funció MAIN

```
sub MAIN($v?, :$arg!, :$arg2 = 'predeterminat') {  
    if ($v) { # verbose  
        say "Inici"  
    }  
    say "arg: $arg, arg2: $arg2"  
}
```

```
$ perl6 main.pl
```

Usage:

```
./main.pl [-arg2=value-of-arg2] -arg=value-of-arg [v]
```

```
$ perl6 main.pl -v -arg=5
```

Inici

arg: 5, arg2: predeterminat

```
$ perl6 main.pl -arg=5 -arg2=foo
```

arg: 5, arg2: foo

# Junctions

```
> say 'ok' if 5 == any(3, 5, 7) # albo 5 == 3 | 5 | 7
ok
> say 'ok' if 5 == none(4, 6, 8)
ok
> my Junction $x = 3 | 5; say 'ok' if $x == 5
ok
> my @scores = 32, 41, 73, 99, 52;
> say 'ok' if all(@scores) > 30;
ok
```

# Laziness

```
> my $even = (2, 4 ... *); $even[^10].perl.say  
(2, 4, 6, 8, 10, 12, 14, 16, 18, 20)  
> my $sq = gather for 0..Inf { take $_ * $_ }; $sq[5].say  
25
```

# Try-CATCH

```
try {  
  die "Oh noes!";  
  CATCH {  
    say "Something went wrong: $_";  
  }  
}
```

# OOP

## Perl 5 amb Moose

```
package Point;
use Moose;

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

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

package Point3D;
use Moose;
extends 'Point';

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

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

## Perl 6

```
class Point {

    has Int $.x is rw;
    has Int $.y is rw;

    method clear {
        $.x = $.y = 0;
    }

}

class Point3D is Point {

    has Int $.z is rw;

    method clear {
        nextsame;
        $.z = 0;
    }

}
```



# Gramàtiques

```
grammar URI {
  token TOP {
    <schema> ':'//
    [<hostname> | <ip> ]
    [ ':' <port> ]?
    '/' <path>?
  }
  token byte {
    (\d**{1..3}) <?{ $0 < 256 }>
  }
  token ip {
    <byte> [\ . <byte> ] ** 3
  }
  token schema {
    \w+
  }
  token hostname {
    (\w+) ( \ . \w+ )*
  }
  token port {
    \d+
  }
  token path {
    <[ a..z A..Z 0..9 _\-.!~*'():@&=+$/,/ ]>+
  }
}

my $match = URI.parse('http://perl6.org/documentation/');
say $match<hostname>; # perl6.org
say $match<path>;    # documentation
```

# Què podem fer ara

Què podem fer mentre esperem a Rakudo Star?

- Escriure codi per detectar errors
- Fer soroll positivament sobre Perl 6 :)

# Enllaços

- <http://perl6.org/>
- <http://parrot.org/>
- <http://rakudo.org/>
- <http://perlgeek.de/en/article/5-to-6> – introducció a Perl 6 per a programadors de Perl 5
- <http://perl6advent.wordpress.com/> – Perl 6 Advent Calendar, una sèrie d'articles interessants amb notícies sobre Perl 6
- El canal `#perl6` a `irc.freenode.net`