

Perl 6



fis17.0

SPB - Sociedade Perl do Brasil
<http://www.perl.org.br>

Flávio S. Glock
fglock@gmail.com



Introdução

- Novos conceitos
- Linguagem
- Compilador
- Máquinas Virtuais



Introdução

- qual problema Perl 6 resolve?
- 12 anos de Perl5
- strict by default



migração

```
use v6;  
use perl5::DBI;  
  
use v6-pugs;
```



"jargão" - Perl 5

references, anonymous arrays,
anonymous hashes, symbolic
references, closures,
string eval, threads,
exceptions, modules, OO, ties,

Perl 6



unicode, lazy evaluation, auto
quoting,



"jargão" - Perl 6

coroutines, lazy context,
state, continuation, trait,
multimethods, junctions,
currying, property, grammar,
hyper operator, hypothetical
binding, autoboxing, mixins,

Perl 6



enums, introspection, capture,
rule, macro, coercion,



variáveis

`$scalar`

`@array`

`%hash`

- o símbolo faz parte do nome da variável



escalares

```
$string = "texto";
```

```
$int = 40;
```

```
$float = 3.14;
```

```
$pair =
```

```
    'a' => 1;
```

Perl 6



```
'a' => 1;
```

```
a => 1;
```

```
:a<1>;
```

```
:a;
```



arrays

```
@array = (  
    1,  
    2,  
    3,  
);  
@array = <1 2 3>
```



hash

```
%hash = (  
    'a' => 1,  
    'b' => 2,  
    'c' => 3,  
);  
%hash = {a:<1>, b:<2>, c:<3>}
```



tipos básicos são objetos

`$pair.key`

`$pair.value`

`@array.elems`

`%hash.keys`



tipos básicos são objetos

- `int` "baixo nível"
- `Int` "objeto"

```
my int $x;  
say $x.sqrt; # autoboxing
```



referências

```
$array_ref = @array;  
$hash_ref = %hash;  
$function_ref = &func;  
$method_ref = &class.meth;
```



referências

- métodos se referem ao conteúdo

```
$array_ref.elems
```

```
$hash_ref.keys
```



tipos

```
my Int $n;
```

```
my Int @array;
```

```
my @array  
    is Array of Int;
```



contextos

```
say ?$x;      # lógico
say ~$x;      # string
say +$x;      # número
say int $x    # inteiro
...

```



junction

```
say $x if $x == 2 | 4;
```

```
$k = 4 & 5;
```

```
say $x if $x != $k;
```

```
say $j.pick;
```



blocos de código

```
    { ... }      # bare block  
-> { ... }      # "pointy sub"
```

```
for 1..10 -> $x { say $x }
```



blocos de código

```
sub x ( ) { ... }  
method x ( ) { ... }  
rule nome {  
    <upper><lower>+  
}
```



blocos de código

submethod

coro

rule

macro

multi sub

multi submethod, multi method



conjuntos de blocos de código

module

class

role

grammar



classes

```
class Xxx
{
    method xxx() { ... }
}
```



gramáticas

grammar Names

```
{  
  rule singlename {  
    <upper><lower>+  
  }  
  rule name :w {  
    <singlename> <singlename>  
  }  
}
```



mixin

```
my Dog $fido = new Dog;  
  
$fido does Tricks;      # role
```



subtype

```
subtype Even of Num
    where { $^n % 2 == 0 }
```

```
my Even $n;
$n = 2;    # Ok
$n = 3;    # Erro
```



propiedades (properties)

```
$result = 0  
        but true
```



características (traits)

```
my $pi
    is constant = 3.14;
my @array
    is dim(10,10)
```



hyper-operator

```
my @array = ( 1, 2, 3 );  
say @array >>*<< 2
```

```
# 246
```



reduce

```
$sum = [+] @array
```

- hyper-reduction

```
$sum =  
    [+]« ( @array1, @array2 )
```



prefix, infix, postfix, circumfix

```
sub postfix:<!> { [*] 1..$_ }
```

```
4!           # 24
```

```
(4, 5) >>!  # (24, 12)
```



parâmetros formais

```
sub japh (Str $lang) {  
    say "just another ",  
        "$lang hacker"  
}
```



currying

```
my &perl6Japh :=  
    &japh.assuming("Perl6");  
  
perl6Japh();
```



operadores e funções novas

nor

```
0 nor 0  
# bool:true
```

zip

```
zip ( 1, 2, 3 ),  
    ( 4, 5, 6 )  
# (1, 4, 2, 5, 3, 6)
```



segurança

- declaração de classes e variáveis 'finais'
- sistema modular permite a instalação de níveis de segurança entre o código e a máquina virtual



implementações e backends

em Parrot -

`http://parrotcode.org`

em Haskell - Pugs

`http://pugscode.org`

em Perl5 - Perl6::* Pugs::*

`http://search.cpan.org`

em Javascript -

`perl2js (Pugs)`



modelo de compilação

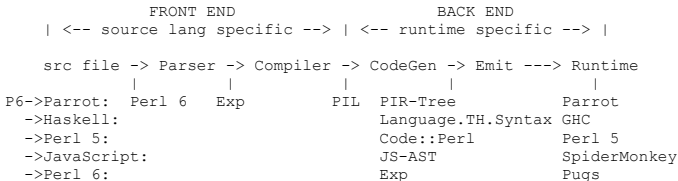
-> Perl6, Ruby, Scheme, Tcl,
Perl5, Python, ...

-> P6AST, PIL, PAST

-> Parrot, Perl5, Javascript,
JVM, Flash, YARV



modelo de compilação





otimizador

pode executar em um thread,
durante runtime



modelo de administração de projeto

Perl6 - "benevolent dictator",
@Larry

Parrot - meritocracia; projeto
administrado pela Perl
Foundation

Pugs - "wiki" - modelo anárquico



- "easy things easier and hard things more possible"

<http://svn.openfoundry.org/pugs/docs/quickref/>

Perl 6



- SPB - Sociedade Perl do Brasil

<http://www.perl.org.br>

- Grupo de usuários Brasil-PM

<http://brasil.pm.org>

<http://mail.pm.org/mailman/listinfo/cascavel-pm>

- IRC: #perl6 (freenode)

© 2006 - Flávio S. Glock

fglock@gmail.com