Perl6 primer

Juste assez pour voir la conférence

```
Parallelism, Concurrency, and Asynchrony in Perl 6
    -- Jonathan Worthington , YAPC::Asia 2016
```

reference a une fonction

reference

en python	perl6
`f`	`&f`

appel de fonction

en python	perl6
`f()`	`f`
`f(1,2,3)`	`f(1,2,3)`
	`f 1, 2, 3`
	`(f 1, 2, 3)`
`f(g(1,2),3)`	`f(g(1,2),3)`
	`f (g 1,2), 3`
`o.foo(34)`	`$o.foo(42)`
	`$o.foo: 42`

Tout est

- object, y compris
  - les classes,
  - les signatures,
  - les routines
- graduellement typé
- coercible

$ = Any

```
my $truc;
my Array $words = [< this is it >];
my Callable $callback = -> $x { $x.name };
```

Method resulution order (Class Hierarchy)

```
for $truc, $words, $callback -> $obj {
    $obj.^mro.say
}

# ((Any) (Mu))
# ((Array) (List) (Cool) (Any) (Mu))
# ((Block) (Code) (Any) (Mu))
```

From Any

```
my $truc;
my Array $words = [< this is it >];
my Callable $callback = -> $x { $x.name };
```

To sigils

```
my $truc;
my @words = [< this is it >];
my &callback = -> $x { $x.name };
```

Whatever

```
my &callback = -> $x { $x.name };
my &callback = { $_.name };
my &callback = { .name };
my &callback = *.name;
```

Lazyness (live demo)

```
my @evens = (0..*).map(* *2);
say @evens[^5];
```

Lazyness (with &say)

```
my @evens = (0..*).map: &say;
say @evens[^5];
```

Gather / take

- collecter des valeurs dans des itérations
  my @gni; for @evens [^5] { $_ > 2 and @gni.push: $_ }
  my @gni = gather for @evens [^5] { take $_ if $_ > 2 }

Ladies and Gentlemen

Parallelism, Concurrency, and Asynchrony in Perl 6

Jonathan Worthington :: jnthn.net

YAPC::Asia 2016

MOAR

Extra time

Array subscripts are callables

```
@users[0..5];  # Range
@users[^6];    # Range

my $half = 0..*/2;
my $last = *-1;

@users[$half];
```