Marc Chantreux

"Picasso, le polyglotte et le philosophe"

Pycon.fr 2012, Paris

Marc Chantreux

UNIVERSITÉ DE STRASBOURG

@marcchantreux

github.com/eiro

irc://eiro@freenode

metacpan.org/author/MARCC

Marc Chantreux

metacpan.org/author/MARCC

cpan

Comprehensive Perl Archive Network (CPAN)

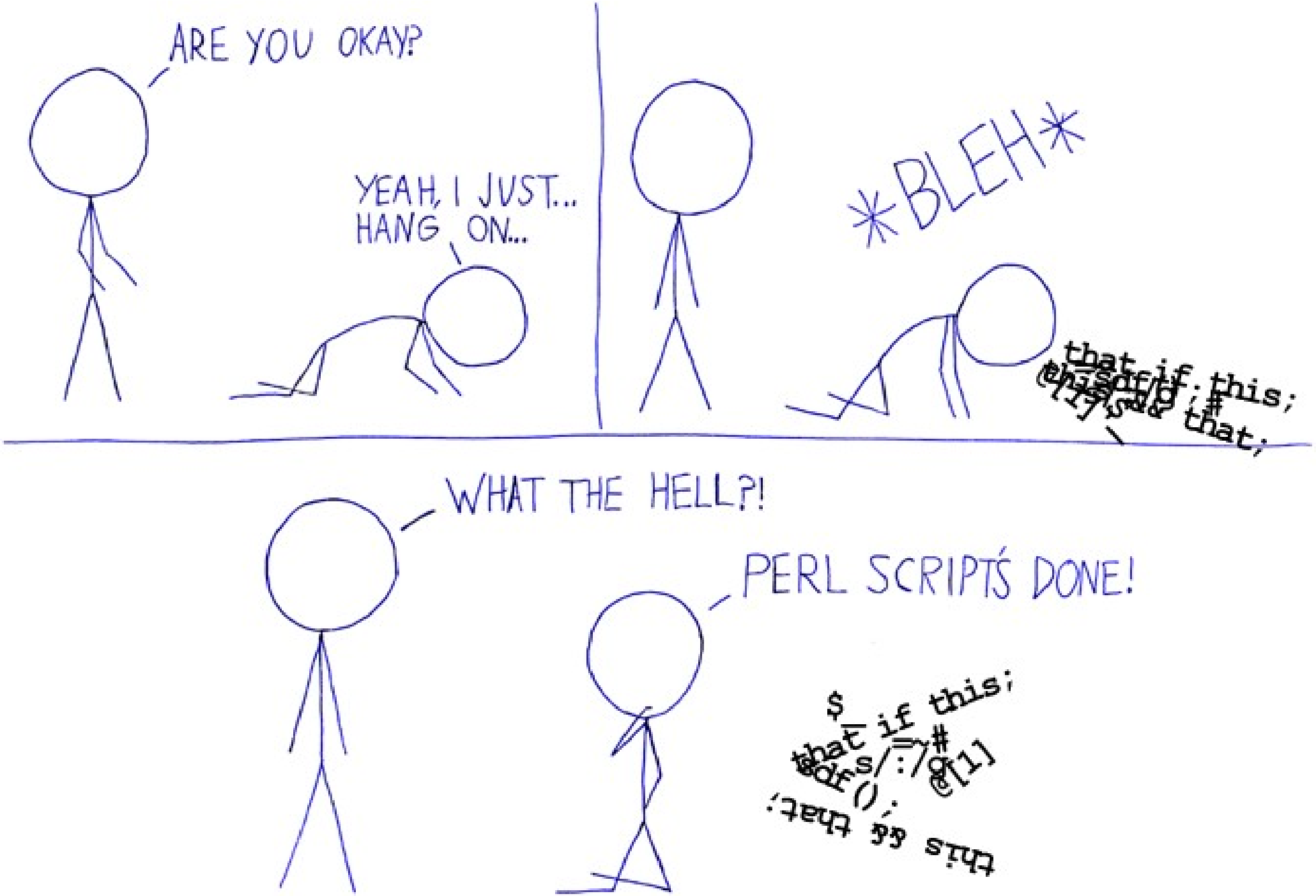
112,366 Perl modules

25,680 distributions

10,019 authors

274 mirroirs

en constante augmentation depuis 1995


ARE YOU OKAY?
YEAH, I JUST... HANG ON...
BLEH
WHAT THE HELL?!
PERL SCRIPTS DONE!

Perl

Le truc illisible que les sysop utilisaient dans les années 90 pour faire des CGI.
C'est du line noise même pas objet

Perl is dead

Mysts

religion

ignorance

croyances

Next?

Ruby, python, haskell, go, lua, ...

Next: Perl

configuration over code

HOP, Perlude

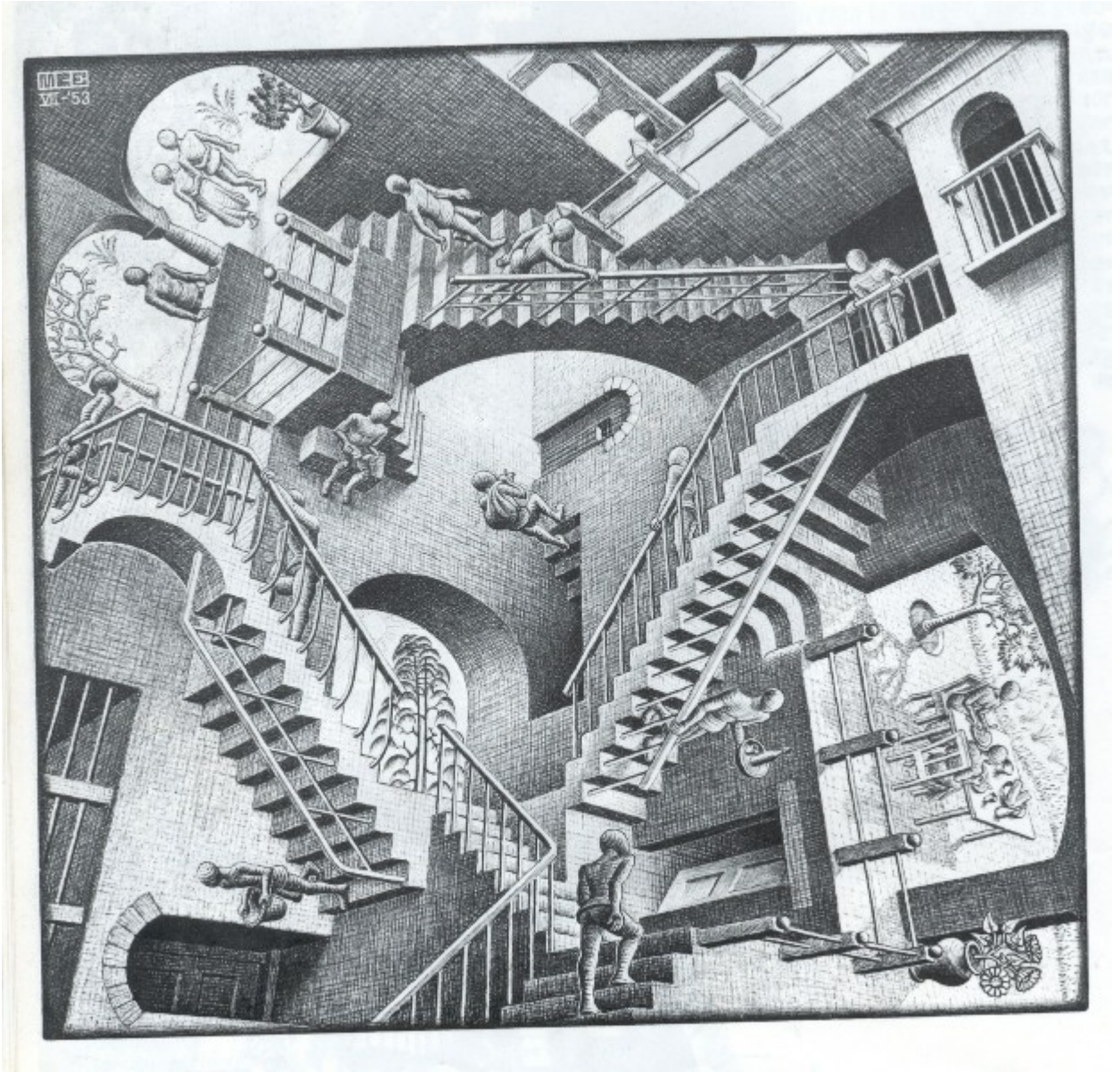
Monkey patching saynul

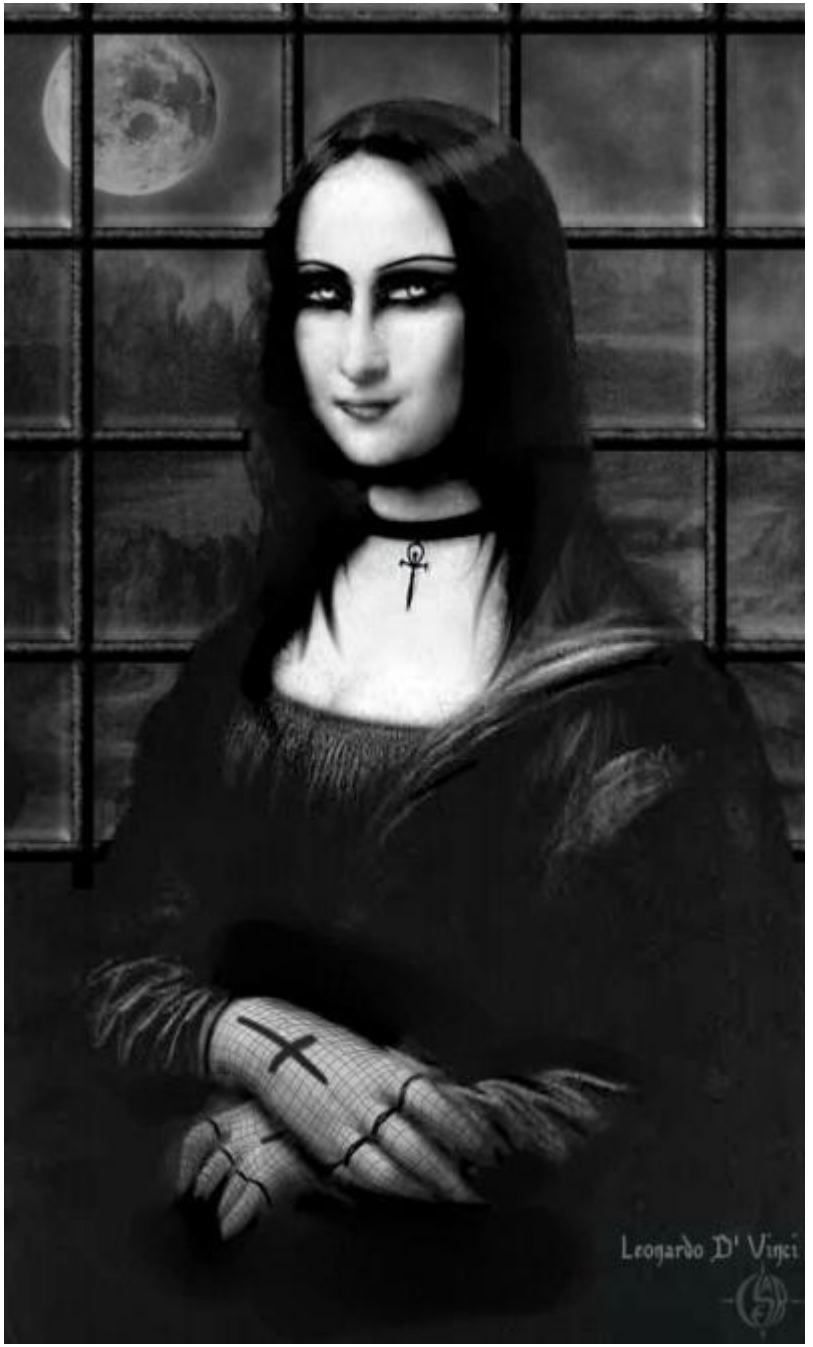
....

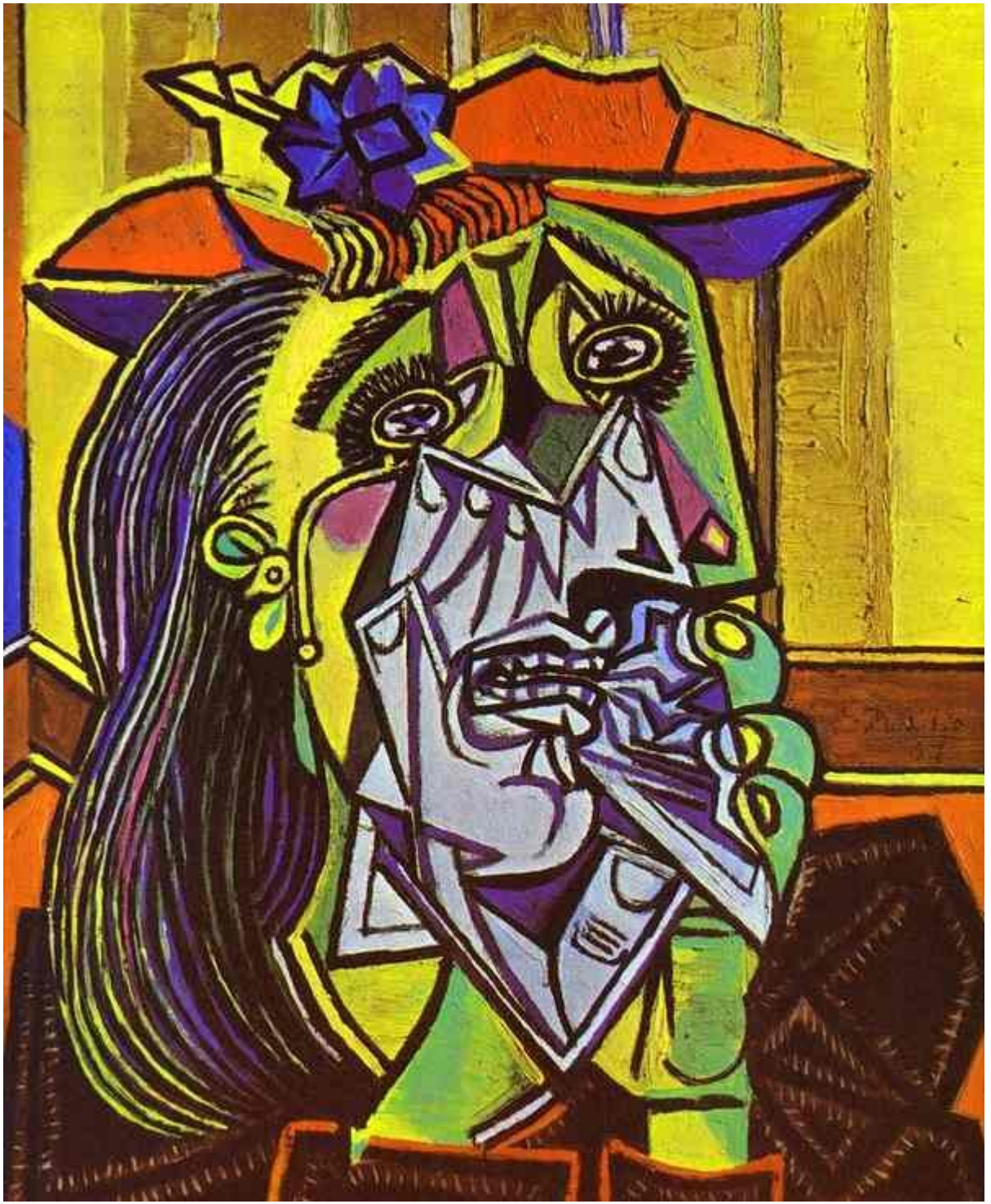
Perl

C'est puissant, expressif,
moche et illisible

Moche?
Picasso
c'est
moche!


MCE
VII-53


Leonardo D' Vinci



```
#!/usr/bin/perl -w
use strict;

     my$f=           $[;my
   $ch=0;sub       l{length}
 sub r{join"",   reverse split
("",$_[$[])}sub ss{substr($_[0]
,$_[1],$_[2])}sub be{$_=$_[0];p
 (ss($_,$f,1));$f+=l()/2;$f%=l
  ();$f++if$ch%2;$ch++}my$q=r
   ("\ntfgpfdfal,thg?bngbj".
    "naxfcixz");$_=$q; $q=~
      tr/f[a-z]/ [l-za-k]
        /;my@ever=1..&l
          ;my$mine=$q
            ;sub p{
              print
                @_;
                }

        be $mine for @ever
```

programming

= art

cpan

= art

perl

= art

python

= art

art

idées

idées

Concurrence, higher order, iterateur, générateur, stream, lazy list, design pattern (décorateur, proxy, factory, ...), composition, implicité, application partielle, monkey patching, metaprogramming, déconstruction, agilité, test driven developpement, do-ocracy, ORM, persistance,

....

art

message

art

émotion

message

culture

art

= subjectivité

Guernica:

pacifisme

Idées != oeuvres

"where good ideas
come from"

Slow hunch

Cobinaisons, incubation

"where good ideas

come from"

Cobinaisons

Deconstructivism

Matt S. Trout

- Shadowcat systems
- Catalyst
- CPAN (DBIx::Class, Moo, ...)

Deconstructivism

Le succès de PSGI et Plack


"THE BAD ARTISTS
IMITATE, THE GREAT
ARTISTS STEAL."
PABLO PICASSO
BANKSY

PSGI

Tatsuhiko Miyagawa

- spécification WSGI alike en Perl
- En collaboration avec les communautés python et ruby
- KISS: spécifier une chose et la spécifier bien
- Mécanique extensible par middleware

Plack

- une implémentation de PSGI
- Insatisfait? Recommencez en réutilisant

  la spec, les middlewares, la suite de tests, les outils qui communiquent avec PSGI
- Membre d'une autre communauté?

  Implémenter vs Réinventer

Richard Dawkins

"selfish gene"

Notre communauté, un réservoir mémétique

"selfish gene"

Partager

Partager

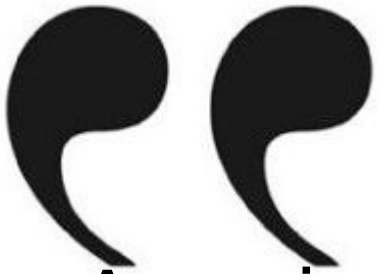
Avec des gens qui partagent ...

Acmeism

Ingy döt Net

- perl monger
- python, javascript ruby, ...
- inventeur de YAML
- Acmeism
- http://acmeism.org

Acmeism



Acmeism is the belief that language naturally tends to divide people and ideas, but that technology can overcome this tendency. People who create technology that is not limited to a particular language are known as Acmeists.

Acmeism

Ingy döt Net

- Pegex: une grammaire = 1 lib pour python, perl5, perl6 javascript, ruby, ...
- Cdent, le non-langage

Acmeism

Franck Cuny, Nils Grunwald

- Spore: décrire plutôt que programmer les clients REST
- (implementation perl, lua, ruby, javascript, java, php, python, ...)

Acmeism

moi

MARC::MIR::Template

décrire le contenu des enregistrements MARC en yaml ou json. Implémentations en cours en perl5, perl6. D'autres langages espérés

Acmeism

Vous?

Faites le nous savoir!

- Communiquer que acmeism.org
- Implémenter en perl5 et 6

La communauté = Open Source != notre langage

Notre communauté manque de visibilité

Notre communauté manque de moyens

Économiser
professionaliser
mutualiser

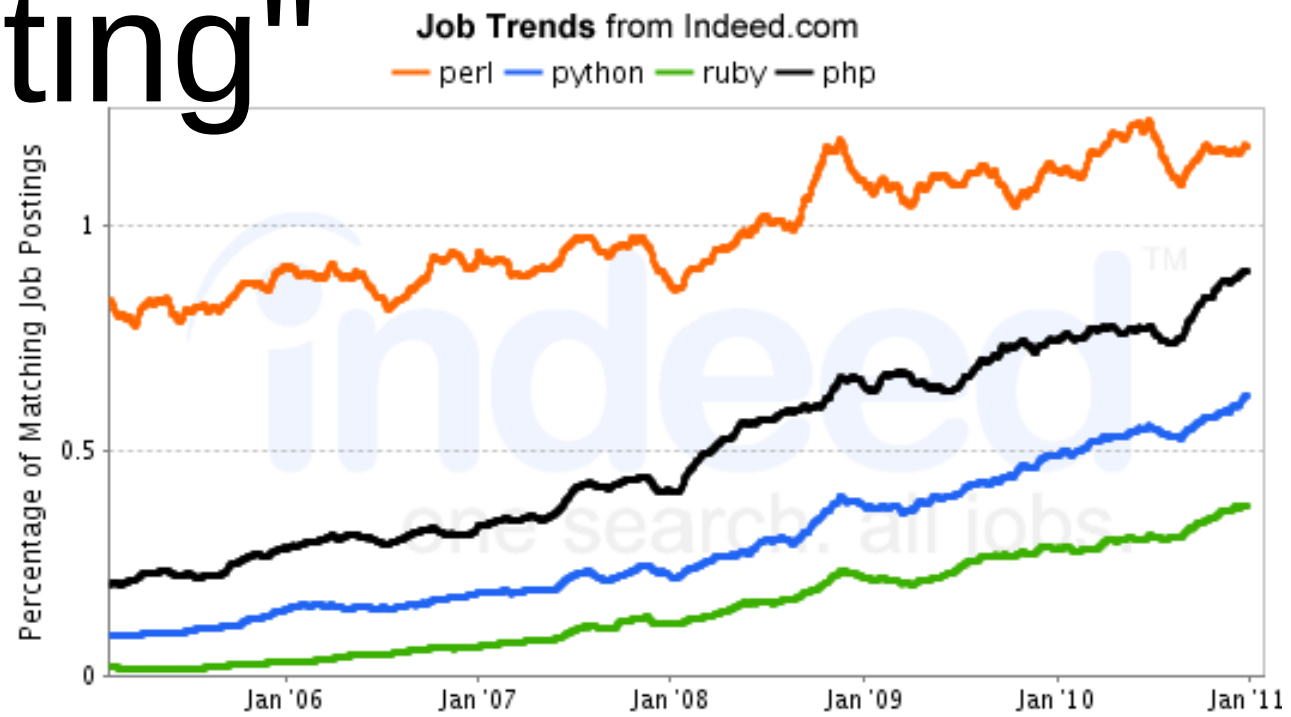
Économiser

- de troll =

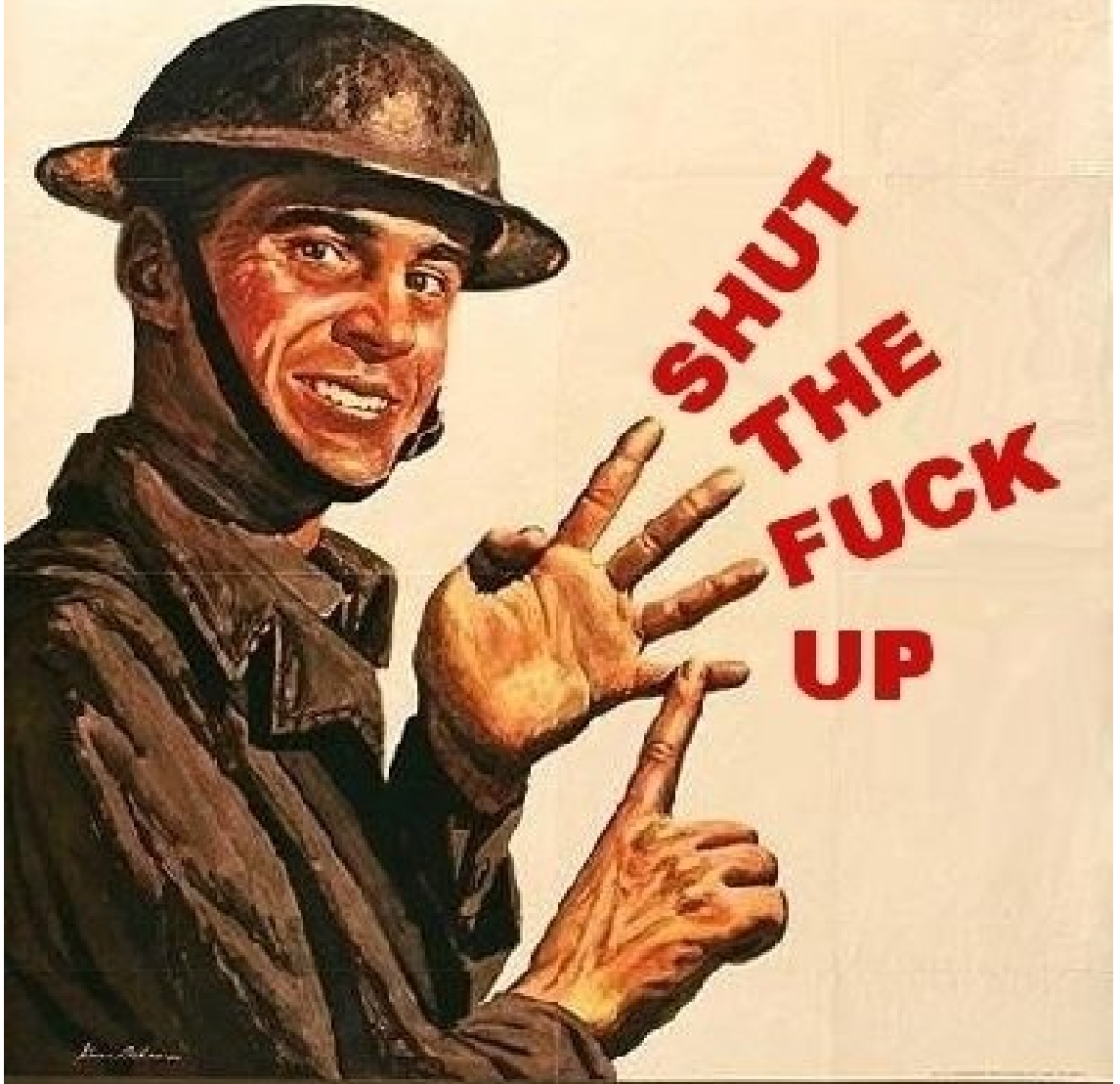
+ de code, spec, pubs, events,orga

+ de crédibilité

Troll , "We suck at marketing"



FOUR WORDS FOR YOU



TAKE THE HINT, PUNK!

- Communautés

Créer des ponts culturels

pour faire fonctionner

les ponts techniques

•Communautés

mixité culturelle

- Pros/cons des langages
- En fonction
  - Des besoins
  - Des goûts

•Communautés

- Fora
  - Aider objectivement

•Communautés

Challenge perso ?

Découvrir un langage par an

Connaître tous plusieurs paradigmes

Echanges technoagnostiques

- Support commerciaux
  - Modèles économiques
  - Visibilité commerciale des contributeurs

Communautés

Conférences

- Par la communauté
- Pour la communauté

•Communautés

- Organisation
  - Chronophage
  - Ingrat
  - Important

Laurent Boivin, un de mes héros

- 0 module CPAN
- Journée Perl
- Perl QA hackathon
- OSDC.fr
- ...

Qui a organisé vos confs ces dernières années ?

- Communautés

- Mutualiser
  - Équipes, contacts
  - Outils interopérables
  - Retours d'experience

- Communautés
  - Mutualiser
    - La présence
    - Le matos ?

•Communautés

Lieux et moments d'échange

•hackerspaces

OSDC.fr

Questions?

Extra time ? Exemples!

- url_dispatcher commun a sinatra, flask, dancer,...
- to_proc, un cas concret profitable surtout à python

Une dernière chose ...

Questions?

Rendez-vous à Strasbourg en 2014?

url_dispatcher

Comparaison

```
Flask      | python | '/user/<id>'
Spore      | acme     | '/user/:id'
Dancer     | perl5    | '/user/:id'
Bailador   | perl6    | '/user/:id'
Sinatra      | ruby   | '/user/:id'
```

url_dispatcher

Comparaison

```
Flask       | '/user/<int:id>'
Dancer      | qr{/user/(?<id>\d+}
Perl6       | rx( /user/[ $<id> = \d+ ] )
Perl6 mieux | rx(
    '/user/' $<id> = \d+ <?{ $<id> < 1000 }>
)
```

to_proc

Comparaison de syntaxes de filtres

```
Python      | lambda x: x < 100
Perl        | { $_ < 100 }
Haskell       | (<100)
Clojure       | #(%<100)
```

to_proc

- Une faiblesse de python
- Des bonnes idées piquées à ruby et Perl6
- Des avantages dans Perl5

to_proc

Utilisation d'un générateur fib en python

```
sum(select(where(take_while(fib(), lambda x:
x < 1000000) lambda x: x % 2), lambda x: x *
x))
```

to_proc

Version préfixée (piquée à LINQ, unix shell)

```
fib() | take_while(lambda x: x < 1000000) \
      | where(lambda x: x % 2) \
      | select(lambda x: x * x) \
      | sum()
```

to_proc

Comparaison de syntaxes de filtres

```
Python      | lambda x: x < 100
Perl        | { $_ < 100 }
Haskell       | (<100)
Clojure       | #(%<100)
```

to_proc

Ruby `String#to_proc`

```
'<100'.to_proc
```

to_proc

Une chaine, une grammaire, 1 soluc. Pour N langages ?

```
1,2, * + * .. *
|! % 100_000
| * 2
|> 0, * + *
```