Pharo: Object at your Fingertips

Marcus Denker

http://www.pharo-project.org

What is it?

- Language + Environment
- Simple Language (Smalltalk)
- Object-Oriented, Dynamic, Reflective
  - Explore + Change running systems

- **The Ultimate Programming Environment!**

Pharo

- MIT license
- Mac, Linux, Android, iOS, Windows
- Great community
- Improving steadily
- Many excellent libraries

Started 2008

- Pharo 1.0 released October 2009
- 2.0 is the current stable
- Pharo3: Early 2014

Plan: 1 Release per year

Language
(in 10 minutes)

Pharo started with Smalltalk-80 (Squeak)

Language still very much Smalltalk

But the goal is to develop it further

Example: Reflective Capabilities

Less is better

- No constructors
- No types declaration
- No interfaces
- No packages/private/protected
- No parametrized types
- **Yet really powerful**

Objects are instances of Classes

Objects are instances of Classes

(10@200)

Objects are instances of Classes

(10@200) class

Objects are instances of Classes

(10@200) class

Point

Classes are objects too

Classes are objects too

Point selectors

Classes are objects too

Point selectors

> an IdentitySet(#eightNeighbors #+ #isZero #sortsBefore: #degrees #printOn: #sideOf: #fourNeighbors #hash #roundUpTo: #min: #min:max: #max #adaptToCollection:andSend: #quadrantOf: #crossProduct: #= #nearestPointOnLineFrom:to: #bitShiftPoint: #* #guarded #insideTriangle:with:with: #grid: #truncateTo: #y #setR:degrees: #normal #directionToLineFrom:to: #truncated #nearestPointAlongLineFrom:to: #theta #scaleTo: #encodePostscriptOn: #> #asPoint #extent: #r #roundTo: #max: #interpolateTo:at: #triangleArea:with: #angleWith: #dotProduct: #isSelfEvaluating #'<=' #to:intersects:to: #'//' #isInsideCircle:with:with: #< #scaleFrom:to: #corner: #to:sideOf: #x #'>=' #roundDownTo: #onLineFrom:to:within: #transposed #ceiling #angle #basicType #translateBy: #asFloatPoint #'\\' #adaptToNumber:andSend: #abs #negated #octantOf: #asIntegerPoint #flipBy:centerAt: #scaleBy: #floor #onLineFrom:to: #isPoint #reflectedAbout: #/ #dist: #asNonFractionalPoint #bearingToPoint: #reciprocal #rotateBy:centerAt: #rotateBy:about: #rounded #setX:setY: #squaredDistanceTo: #normalized #veryDeepCopyWith: #- #storeOn: #rect: #deepCopy #isIntegerPoint #min #adhereTo: #adaptToString:andSend:)

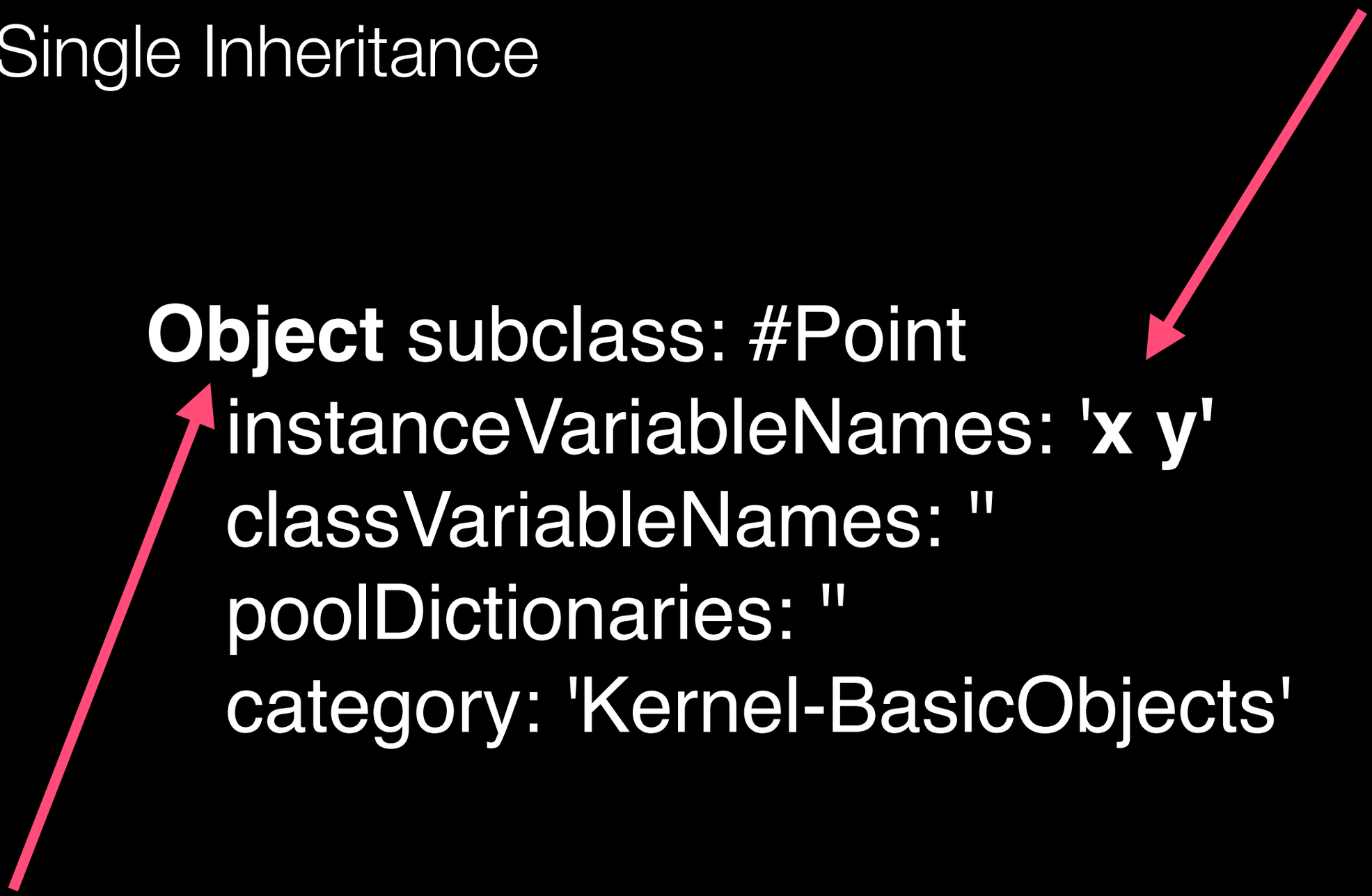
Methods are public

Instance variables are protected

Single Inheritance

Single Inheritance

2 instance variables



Object subclass: #Point
instanceVariableNames: '**x y**'
classVariableNames: ''
poolDictionaries: ''
category: 'Kernel-BasicObjects'

subclass of Object

3 kinds of messages

Unary messages

```
5 factorial
Transcript cr
```

Binary messages

```
3 + 4
```

Keywords messages

```
3 raisedTo: 10 modulo: 5

Transcript show: 'hello world'
```

Blocks

- Anonymous method
- Passed as method argument or stored
- Functions

fct(x)= x*x+3, fct(2).

```
fct :=[:x| x * x + 3].
fct value: 2
```

Control structures

Every control structure is realized by message sends

```
4 timesRepeat: [Beeper beep]
```

```
max: aNumber
   ^ self < aNumber
        ifTrue: [aNumber]
        ifFalse: [self]
```

A typical method in Point

Method name

Argument

Comment

```
<= aPoint
  "Answer whether the receiver is neither
  below nor to the right of aPoint."

  ^ x <= aPoint x and: [y <= aPoint y]
```

Return

Instance variable

Binary message

Keyword message

Block

```
(2@3) <= (5@6)
```

```
true
```

Complete Syntax on a PostCard

exampleWithNumber: x

“A method that has unary, binary, and key word messages, declares arguments and temporaries (but not block temporaries), accesses a global variable (but not and instance variable), uses literals (array, character, symbol, string, integer, float), uses the pseudo variable true false, nil, self, and super, and has sequence, assignment, return and cascade. It has both zero argument and one argument blocks.”

```
|y|

true & false not & (nil isNil) ifFalse: [self halt].

y := self size + super size.

#($a #a ‘a’ 1 1.0)

    do: [:each | Transcript show: (each class name); show: (each printString); show:
    ‘ ‘].

^ x < y
```

Environment

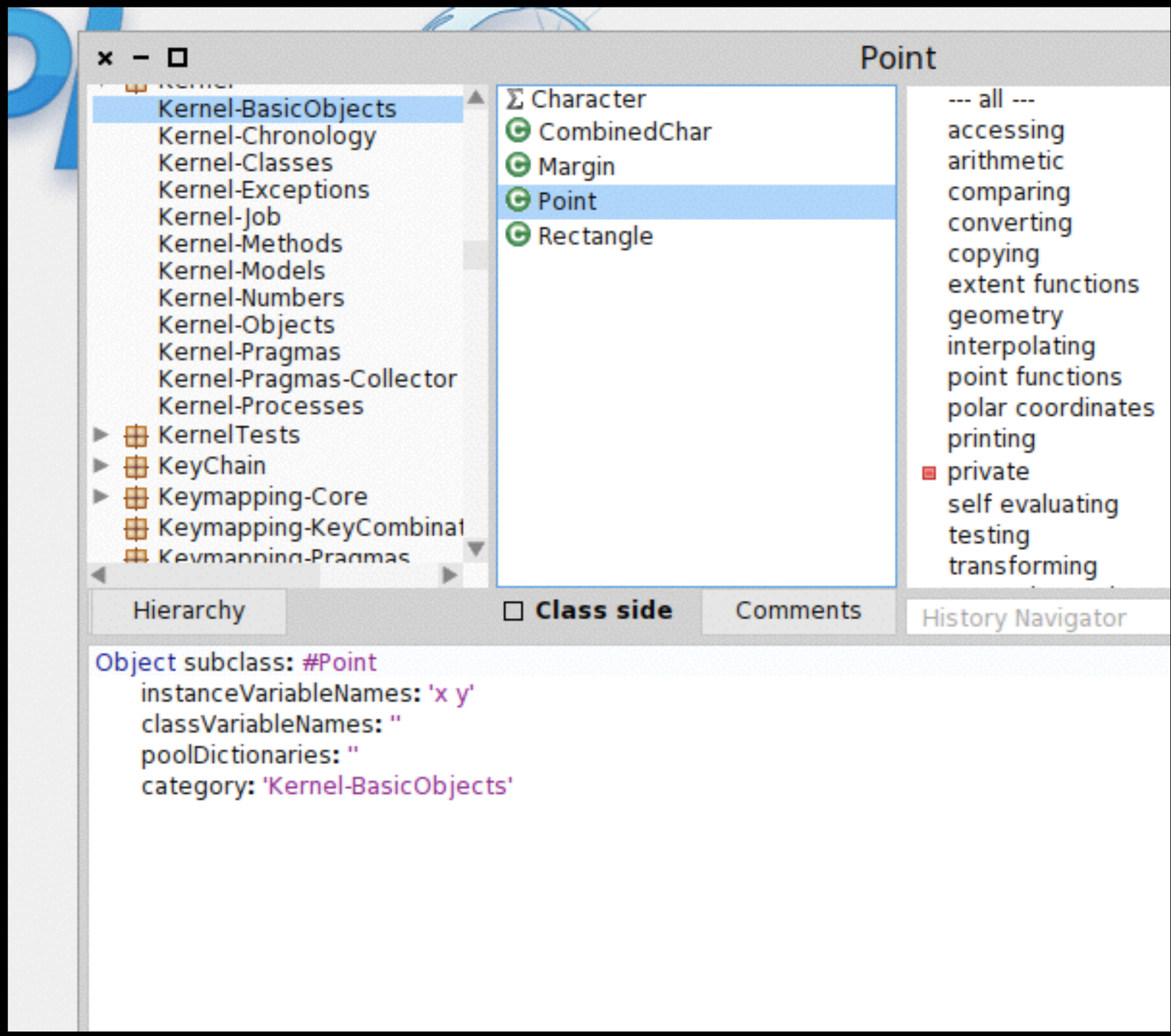
Language + Environment are closely linked

Reflection is the basis

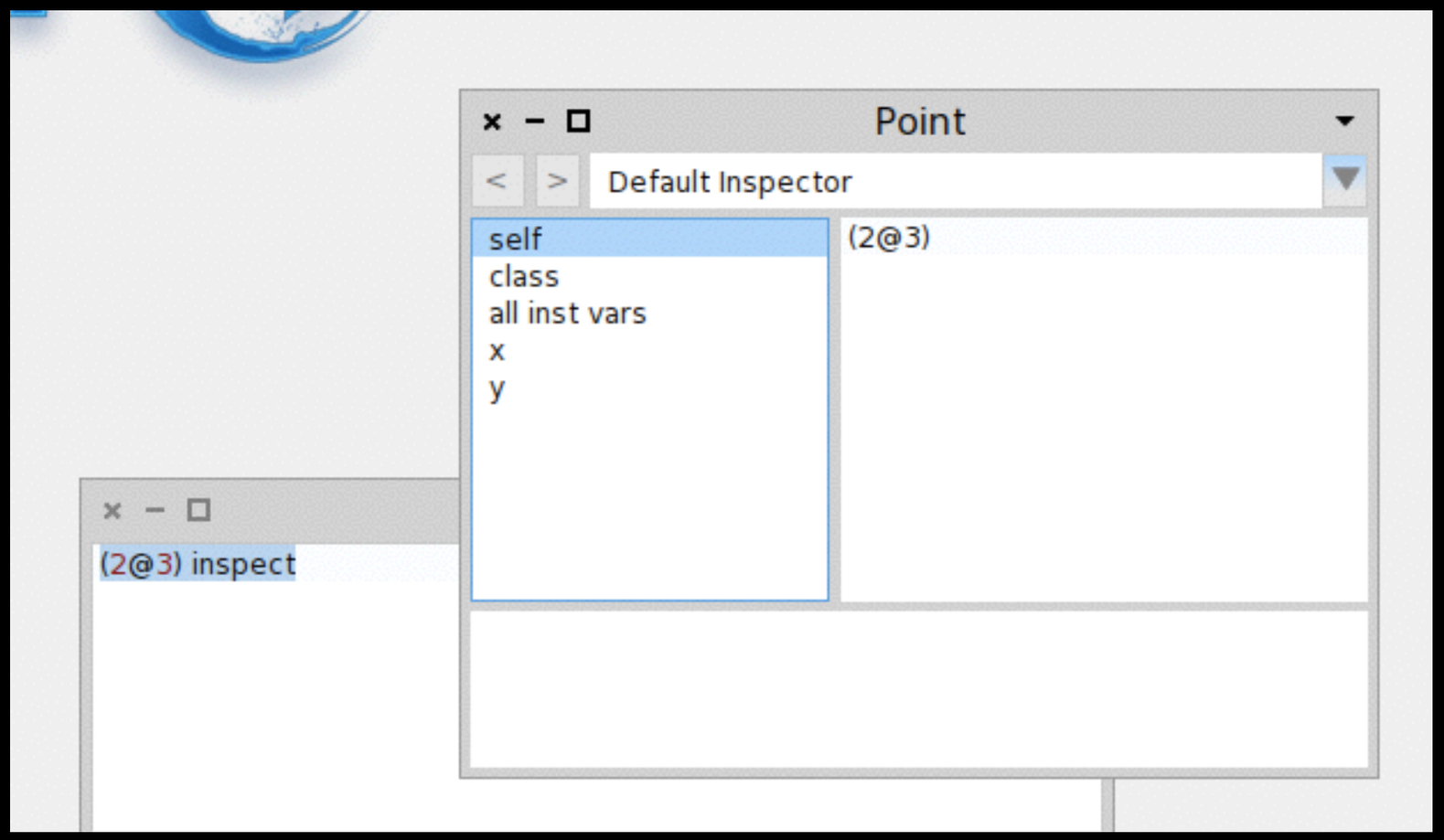
Classes, Methods, Packages are **Objects**

The tools manipulate these Objects

Class Browser



Inspector


Point
Default Inspector
self
class
all inst vars
x
y
(2@3)
(2@3) inspect

Demo: Changing a class at runtime

Demo: Exploring the system

Demo: Inspect World

There is so much more…

Pharo Books


Pharo
by example
Andrew P. Black, Stéphane Ducasse, Oscar Nierstrasz, Damien Pollet
with Damien Cassou and Marcus Denker


DEEP INTO
Pharo
ALEXANDRE BERGEL - DAMIEN CASSOU
STÉPHANE DUCASSE - JANNIK LAVAL
Foreword by Dave THOMAS

Pharo Success Stories

2denker

eMCee interfacetesting

Continuous API Testing

keep your services under control 24/7

www.2denker.de

eMCee
interfacetesting

eMCee is a montoring service for backend interfaces

- Web application to define backend interfaces
- Monitors reliability of interface periodically
- Sends warning if status of interface changes
- Provides overview graphs about reliability
- Snapshots requests for debugging purposes

2denker

mail@2denker.de

T3 Easy

http://easy.t3-platform.net/

Google

Saturday, May 12, 2012 FM5300 G

Time: 5/12/12 14:05
Speed: 181 km/h

t3 EASY

Saturday, May 12, 2012 14:05:29 181 km/u

13:00 13:30 14:00 14:30 15:00 15:30 16:00 16:30 17:00 17:30 18:00 18:30 19:00 19:30

- Entry Level Track & Trace Product
- Complements T3 Full Product
- One Page Javascript HTML5 / Ajax Client
- REST Back End in Pharo Smalltalk
- Gateways to multiple data providers

BetaNine
software engineering

Pharo Consortium

- Managed by INRIA
- Who: companies, institutions, user groups
- Privileged access to the core development team
- Influence priorities of the next development

http://consortium.pharo.org

esug
EUROPEAN SMALLTALK USER GROUP

Inria
INVENTEURS DU MONDE NUMÉRIQUE

vmware

debris publishing

HRworks

netstyle.ch

2denker

COFTIC
Gagnez une Qualité d'Avance

Systems that make sense
Sensus

BetaNine
software engineering

CæsarSystems
Accelerate Confidently

Future + Research

More Reflection

- Instance Variables as Objects —> Slots
- Proxy model in the base language
- Structuring reflective API (—> Mirrors)

Beyond Text

- AST Everywhere
  - Used in Tools for Navigation
- Do we need to store text?
- Use for Behavioral Reflection

System - as - Objects

- Put “virtualization” in the language
- We already use “Images”
- Make the Image a first class concept in the language

Open Pharo Sprints

May 2008 Bern
July 2009 Bern
October 2009 Lille
November 2009 Buenos Ares
March 2010 Bern
May 2010 Buenos Ares
June 2010 Bern
June 2010 Bruxelles
July 2010 London
September 2010 **Barcelona**
September 2010 Lille
January 2011 Lille
July 2011 Lille
October 2011 Bruxelles
February 2012 Bern
April 2012 Lille
September 2012 Ghent
October 2013 Lille
November 2013 Buenos Aires