# Pharo, the immersive programming experience

Hilaire Fernandes

NCHC, Hsinchu, Taiwan

August 2015

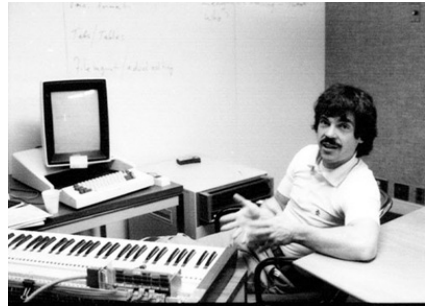
# About Pharo

- `http://pharo.org`
- Pure object language
- Inspired by Smalltalk
- Great community with talented researchers, engineers and programmers
- Powerful, elegant and fun to program
- Living system under your fingers

# Stephane Ducasse, Pharo project leader

- Delivering a clean, innovative, free open source environment to build mission critical application
- Research director at INRIA
- Personal page

# Alan Kay, Dynabook vision



- Grand project to invent modern computing at Xerox Parc in the ’70.
- You know the story about Steve Jobs visiting and picking up ideas from Xerox Parc’s lab to build Macintosh.
- Received 2003 ACM Turing award for work on OOP
- Lead of View Point Research Institute

# What it is

This hands on is about learning:

- the Pharo syntax,
- the standard class libraries,
- the Pharo object model (a bit),
- some IDE tools as Finder, Playground, Spotter, Nautilus and Inspector,
- write Roassal scripts.

# What it is not

This hands on is **not** about learning:

- oriented object programming,
- good design practices,
- to write a desktop application,
- to write a web application,
- to manage Pharo source code for team work,

# Acknowledge

Contents of this hands on course comes from different sources:

- Stephane Ducasse
- Damien Cassou
- Hilaire Fernandes

# Syntax in a postcard

```
exampleWithNumber: x
"A method that illustrates every part of Smalltalk method syntax
except primitives. It has unary, binary, and keyword messages,
declares arguments and temporaries, accesses a global variable
(but not an 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 $z #a "a" 1 1.0) do:
   [:each | Transcript show: (each class name); show: ' '].
^ x < y
```

# Pharo syntax

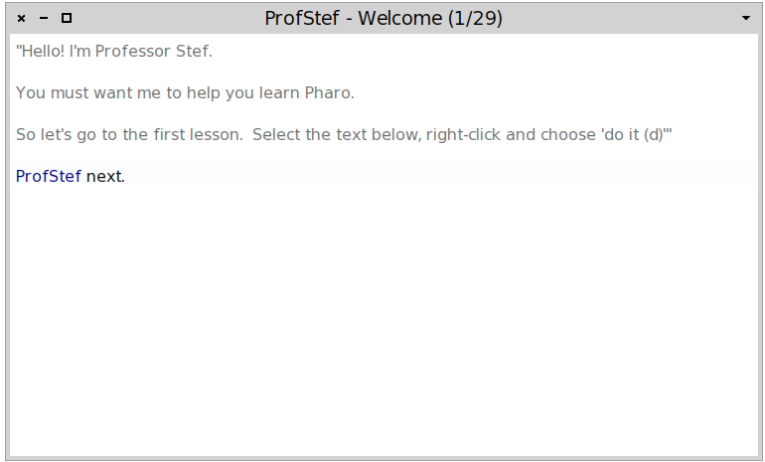
Pharo syntax is very small.

- Get started with the syntax
- The runtime architecture
- Variables and arguments are simple
- **Three** ways to send a message (*i.e.* call a function)
- Delayed code execution with Blocks [...]

*Message sending and block are the building elements of what you may call language constructs in other programming languages.*

# Training with ProfStef

In a Playground window, execute the code `ProfStef go`, you get a tutorial window. It is already open in the hands on environment:



# Exercise 1

Given a collection of numbers from 0 to 50, connect each to its modulo 5:

```
| b |
b := RTMondrian new.
b shape label.
b nodes: #replaceMe .
b edges connectFrom: [ #replaceMe ].
b layout circle.
b
```

Imagine variations in the code:

- collection, consider only even number, etc.
- nature of the connection
- layout

# Exercise 2

- In this script, designate a unary, binary and keyword messages (3).
- The example below draws a French flag. How will you add an Italian flag?

```
| view shape elements flags |
view := RTView new.
shape := RTCompositeShape new.
shape add: (RTBox new color: #second ; height: 20; width: 15).
shape add: (RTBox new color: #third; height: 20; width: 15) translateBy: 15 @ 0.
shape add: (RTBox new color: #fourth ; height: 20; width: 15) translateBy: 30 @ 0.
shape add: (RTRotatedLabel new angleInDegree: 270; height: 8; text: #first) translateBy: −15@0.

flags := OrderedCollection new.
flags add: (Array with: 'France' with: Color blue with: Color white with: Color red).

elements := shape elementsOn: flags.
view addAll: elements.
RTGridLayout new on: elements.
view
```

# Exercise 3

- Designate 2 blocks in this example.
- This code is not well written, there are repeated sequences to build the menu. It deserves to be factored. With the help of collections (Array and OrderedCollection) and iterator rewrite the code.

```
| v map hsinchu taipei greenIsland movingCamera menu|
../..
"Place to go"
hsinchu := 24.8098@120.9689.
taipei := 25.083@121.55.
greenIsland := 22.666667@121.483333.
../..
"Adding a menu"
../..
mb menu: 'Hsinchu' callback: [ movingCamera value: hsinchu ].
mb menu: 'Taipei' callback: [ movingCamera value: taipei].
mb menu: 'Green Island' callback: [ movingCamera value: greenIsland].
../..
```

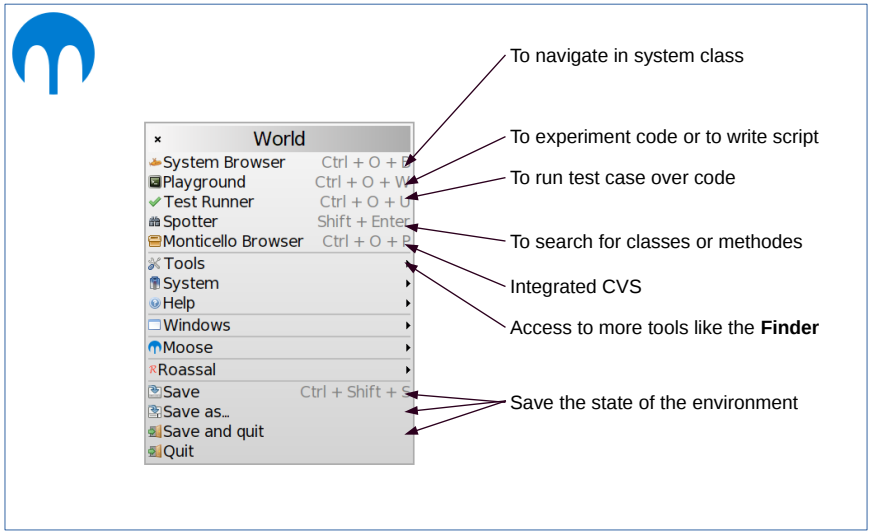
# Pharo IDE

We present the following tools:

- The **World menu**, to access to the tools
- The **playground**, the place to write script
- The **class browser**, to browser the source code
- The **finder**, to search for methods

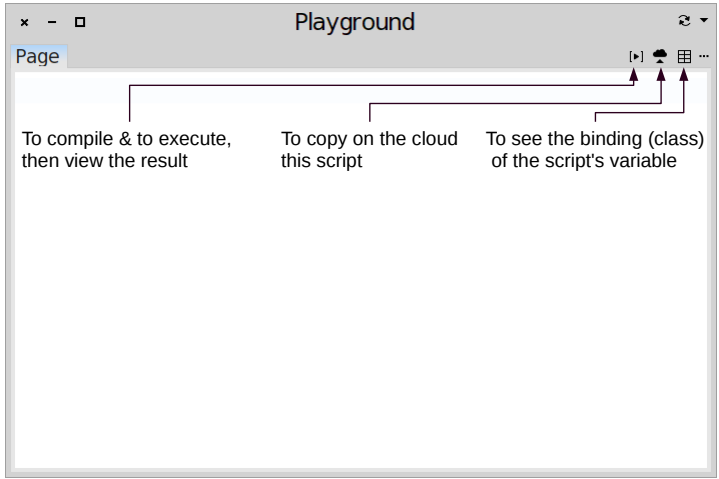
# The World

A mouse left button clic in Pharo opens the **World** menu:

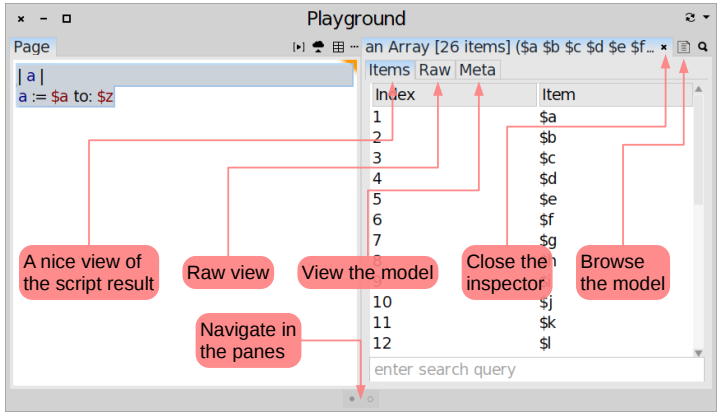


# Playground

- **It is** the preferred place to try out code idea or to write script.
- **It isn't** the place to write application or class.



# Playground extended



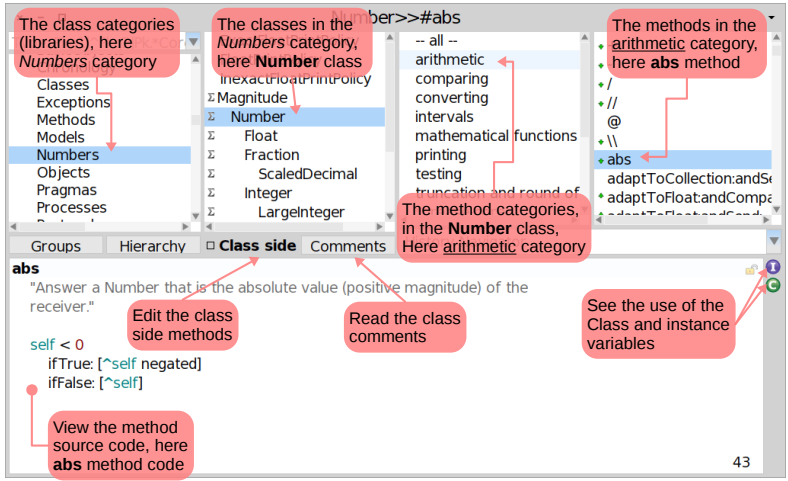
# Playground shortcut

Mouse right clic in the text area for action on the code (with partially selected code or not):

- `Do it and go`: compile, run and see the result of the code in a pane inspector, just at the right
- `Do it`: compile and run the code
- `Print it`: compile, run and print the result just after the code
- `Inspect it`: compile, run and inspect the result in a new inspector window
- `Debug it`: compile and run the code within a debugger
- `Profile it`: compile, run and profile the code. A new profiler window shows the result

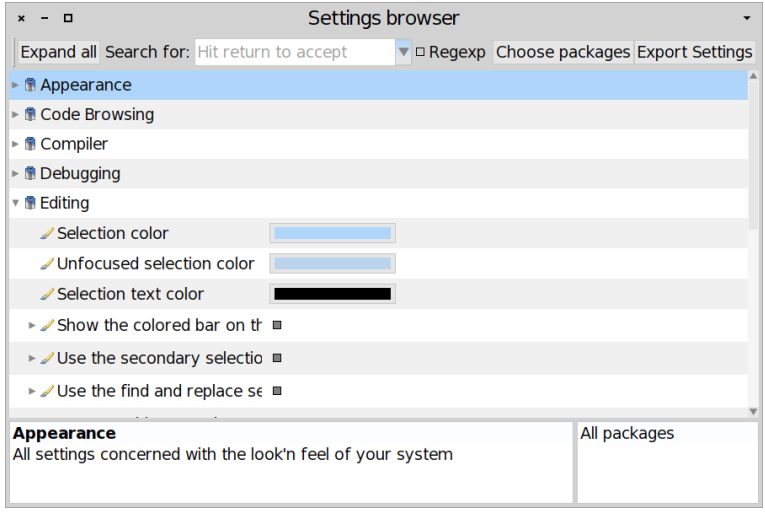
# The class browser

It is the tool to see class source code and to write new one. In our course, we just need it to explore the system classes. Mouse right clic on each pane open a specific menu.



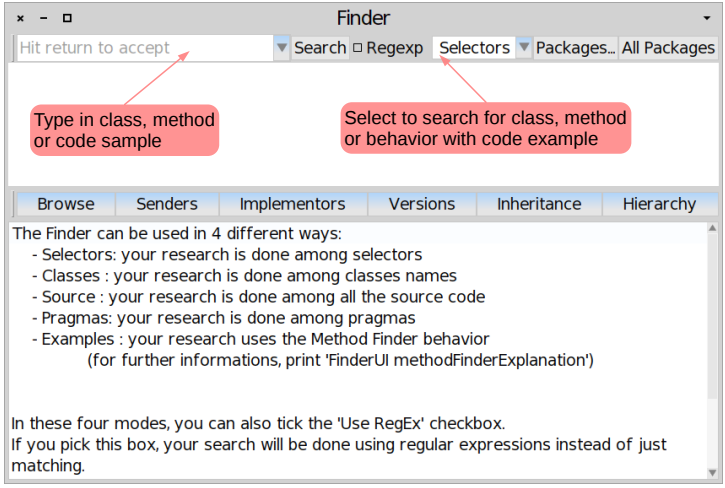
# Settings browser

Open from `World menu>System>Settings` to edit Pharo behaviours and user interface.



# The Finder

Open from `World menu>Tools>Finder` to search for code behaviours, classes, methods.



# The finder, examples 1/2

Try to search for behaviours with **Examples** option in the Finder:

- Find out addition, multiplication, square:

```
2 . 2 . 4
```

- Find week day:

```
'2015/8/17' asDate . 'Monday'
```

- How to concatenate to string:

```
'NCHC ' . 'Taiwan' . 'NCHC Taiwan'
```

- How to calculate square root:

```
9 . 3
```

- How to get the greatest between two numbers:

```
14 . 85 . 85
```

# The finder, examples 2/2

Try to search for behaviours with **Examples** option in the Finder:

- How to get the greatest number in a collection:

```
#(1 10 8 −5) . 10
```

- How to remove unnecessary space:

```
' hello   ' . 'hello'
```

- How to capitalize sentence:

```
'hello world' . 'Hello world'
```

- How to upper case:

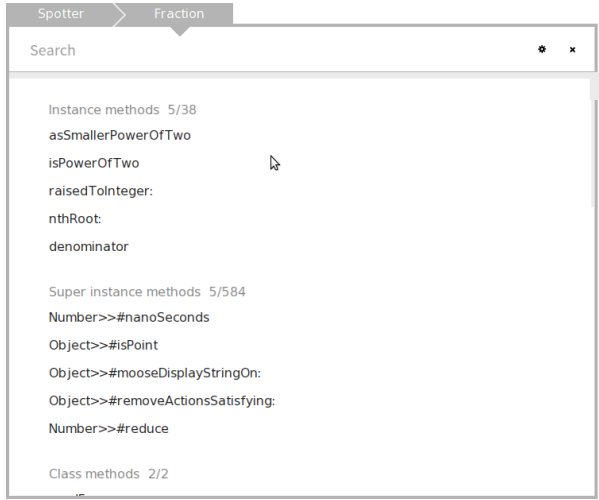
```
'hello' . 'HELLO'
```

- Find the substrings composing a string:

```
'hello my dear' . {'hello' . 'my' . 'dear'}
```

# Spotter

To quickly search for classes or methods, open **Spotter** with the short-cut `[Shift]+Space`, then type in a class or method name:



# Language constructs

What you call syntax in other programming language is implemented with Pharo itself in the **Pharo class libraries**:

- Numbers
- Booleans
- Collections (*i.e.* array/table/list/dictionary, etc.)
- Iterators
- (...) or [...]?

# Exercises 4

We want to display the directory hierarchy with a tree view. Each rectangle represents a file or directory. It should be coloured in red when it's a directory, otherwise in blue.

```
| b|
b := RTMondrian new.

b shape rectangle color: [ :aFile |
   aFile isDirectory
   "Paint me in red when directory
   otherwise blue"].
b nodes: FileLocator image parent allChildren.
b edges connectFrom: #parent.
b normalizer normalizeSize: #size using: #sqrt.
b layout tree.
b
```

# Exercise 5

Edit the script to display earth quake from last five days, or from Japan:

```
| tab b |
tab := RTTabTable new input:  'http://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/2.5_month.csv'
       asUrl retrieveContents  usingDelimiter: $,.

tab removeFirstRow.
tab replaceEmptyValuesWith: '0' inColumns: #(2 3 4 5).
tab convertColumnsAsFloat: #(2 3 4 5).
tab convertColumnAsDateAndTime: 1.

b := RTMapLocationBuilder new.
b shape circle
   size: [ :v | (2 raisedTo: v) / 2 ];
   color: (Color red alpha: 0.3).
tab values do: [ :row |
   b addPoint: row second @ row third value: row fifth].
b
"ignore me $"
```

# Exercise 6

Modify this script:

- Size is scaled logarithmically
- Over a certain threshold, the colour of the shape is different
- The popup text to contain both the country code and the hit in a nice sentence
- linear layout, elements sorted

```
| data view shape elements |
data := NeoJSONReader
  fromString: (FileLocator imageDirectory / 'WorldCountry.json') contents.
view := RTView new.
shape := RTCompositeShape new.
shape add: (RTEllipse new
  size: [:sample | sample second / 68000];
  color: (Color red alpha: 0.5)).
shape add: (RTLabel new text: #first).
elements := shape elementsOn: data.
view addAll: elements.
elements @ (RTPopup new text: [:sample | sample second ]).
../..
```

# Pharo Object model

The motto of Pharo object model is simplicity and understanding.

- The model introduction
- Class and method definition
- Instanciate object with Class methods

# Food for the mind

- How will you implement **not**?
- How will you implement **or** (| in Pharo)?
- Why this question? What these questions want to show us?

Need a course? Read Design, essence of dispatch

# Resources

- Pharo, visit documentation section
- Pharo by Example book. Will be updated soon.
- Roassal documentation
- Recommended reading:
  - Smalltalk with style
  - *Smalltalk design pattern companion book* (a must for OOP design)
  - More free books
- Pharo user forum. It is the mirror of the Pharo user mailing list. It is also possible to subscribe to newsgroup `gmane.comp.lang.smalltalk.pharo.user`