Pharo Object Model in a Nutshell

Elegance and Simplicity

Damien Cassou, Stéphane Ducasse and Luc Fabresse

http://www.pharo.org
Objects: mouse pointer, booleans, arrays, numbers, strings, windows, scrollbars, canvas, files, trees, compilers, sound, url, socket, fonts, text, collections, stack, shortcut, streams...

Messages sent to these objects: size, +, at:put:, do:, ...
... and Block Closures

- Messages are **what** (intent)
- Methods are **how** to do it
- Closures are kind of anonymous methods
- Closures are called **blocks** in Pharo

```scheme
4 times Repeat:
  [ Transcript show: 'Hello World' ]
```

- `[ ... ]` delimits a block
A Simple and Uniform Model

- **Everything** is an object, instance of a class
  - Classes and messages are objects too!
- All computations between objects are done via **message passing**
- We use the term **sending a message** because:
  - methods are always looked up dynamically
  - only late binding, only virtual calls
- Only **ONE** method lookup for all objects
Pharo Object Model

- Instance variables are protected:
  - private to the object
  - accessible from subclasses
- Methods are public and virtually bound
- Single inheritance between classes
Computation between objects is done via message sends
Example of the cross product of two points:

\[(\text{point1} \times \text{point2} y) - (\text{point1} y \times \text{point2} x)\]
Object Creation: Creating a Point

A new object can be created by sending a message to another object

10@20

A new Point object is created by:

- sending the message @
- to the object 10 (SmallInteger)
- with the argument 20 (SmallInteger)
Object Creation: Creating a String

A new String is created as the concatenation of two strings by:

- by sending the message `,`
- to the string ’Pharo’
- with the string ’ is Cool’ as argument
Object Creation

Sending the messages `new` and `new:` to a class

```
Monster new
> aMonster
```

```
Array new: 6
> #(nil nil nil nil nil nil)
```

Here we get an array of size 6
Object Creation

Sending instance-creation messages to a class

Tomagoshi with Hunger: 10

This executes a class method
Less is More :) 

- No constructors 
- No static methods 
- No type declarations 
- No interfaces 
- No package/private/protected modifiers 
- No parametrized types 
- No boxing/unboxing 
- still really powerful :)
Summary

- Everything is an object
- Computation is done via messages sent to objects
- Methods are late bound (looked up dynamically in the inheritance chain)
- Blocks are kind of anonymous methods
- Instances are created by sending messages to other objects, or classes
A course by

Inria

and

UTOP

Université de Technologie
Ouverte Pluripartenaire

in collaboration with

uniscll

Fondation unit

Université de Lille

INSTITUT Mines-Télécom

MINES Douai

Inria 2016

Except where otherwise noted, this work is licensed under CC BY-NC-ND 3.0 France
https://creativecommons.org/licenses/by-nc-nd/3.0/fr/