Pharo Object Model in a Nutshell

Elegance and Simplicity

Stéphane Ducasse and Damien Cassou

http://stephane.ducasse.free.fr/
stephane.ducasse@inria.fr

Table Of Content

Less is really more :)

- No constructors
- No static method
- No type declaration
- No interfaces
- No packages/private/protected modifiers
- No parametrized types
- No boxing/unboxing
- And still really powerful :)

Only Objects and Messages

- Only objects: mouse, booleans, arrays, numbers, compressed, strings, windows, scrollbars, canvas, files, trees, compilers, sound, url, socket, fonts, text, collections, stack, shortcut, streams,...
- and messages sent to these objects.

Simple and uniform

- Everything is an object instance of a class
- Classes are objects too
- Only message passing
- Only one method lookup
  - Only late binding (virtual call)

Pharo Object Model

- Instance variables are private to the object (instance-based).
- Instance variables are protected.
- Shared variables between classes.
- Methods are public
- Single inheritance between classes

Class Definition

```
Object subclass: #Point
instanceVariableNames: 'x y'
classVariableNames: ''
category: 'Graphics'
```

Method Definition

- Normally defined in a browser or (by directly invoking the compiler)
- Methods are public
- Always return self

```
Node>>accept: thePacket
"If the packet is addressed to me, print it.
Else just behave like a normal node"

thePacket isAddressedTo: self)
ifTrue: [ self print: thePacket ]
ifFalse: [ super accept: thePacket ]
```

Instance Creation are Messages Too!

- Messages sent to instance

```
'1', 'abc'
1@2
```

- Basic class creation messages are `new` and `new:` sent to a class

```
Monster new
Array new: 6
```

- Class specific message creation (messages sent to classes)

```
Tomagoshi withHunger: 10
```

Summary

- Everything is an object.
- We send messages to objects.
- Method selection is late bound.