A Little Journey in the Pharo Object Model

Stéphane Ducasse
http://www.pharo.org

A pure and minimal object model

Less is more!

No constructors, no static methods, no operators

No type declaration, no primitive types,

No interfaces, no need for factory

No packages/private/protected modifiers

No parametrized types

No boxing/unboxing

Still powerful

Everything is an object

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

Classes are objects too

Point instVarNames

Classes are objects too

```
Point instVarNames
 >#('x' 'y')
```

Methods are public

Methods are all late-bound

Instance variables are protected

Single Inheritance

Single Inheritance

Object subclass: #**Point**

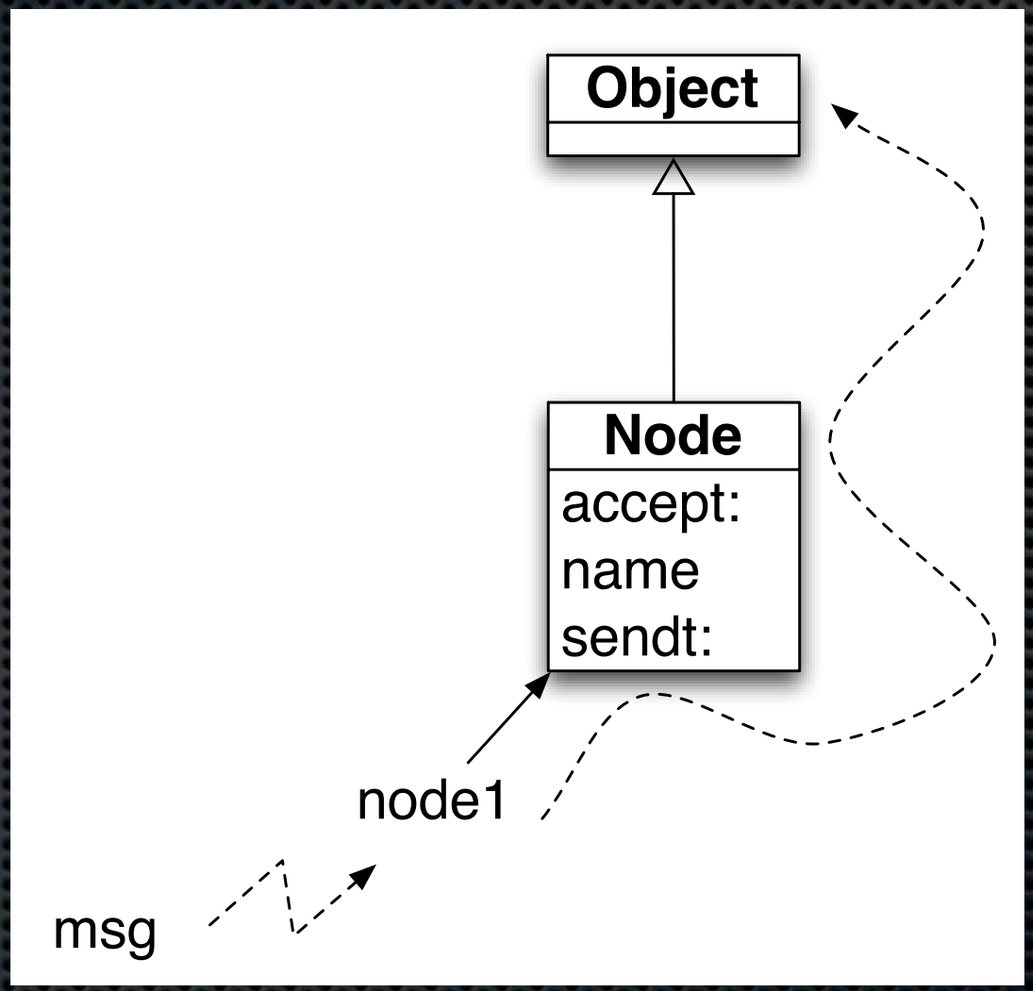
instanceVariableNames: **'x y'**

classVariableNames: ''

category: 'Graphics-Primitives'

Messages + Objects

The key to everything



Classes are objects too

Classes are objects too

Point class

Classes are objects too

Point class

>Point class

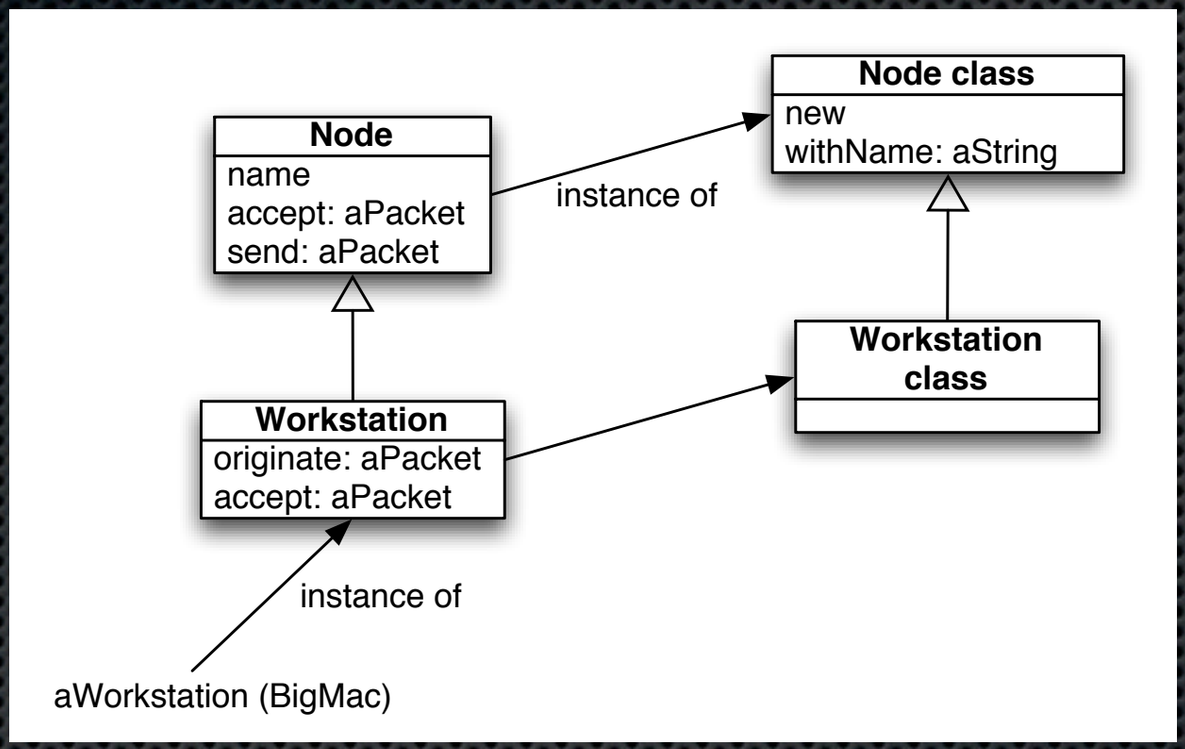
Classes are objects too

Point class

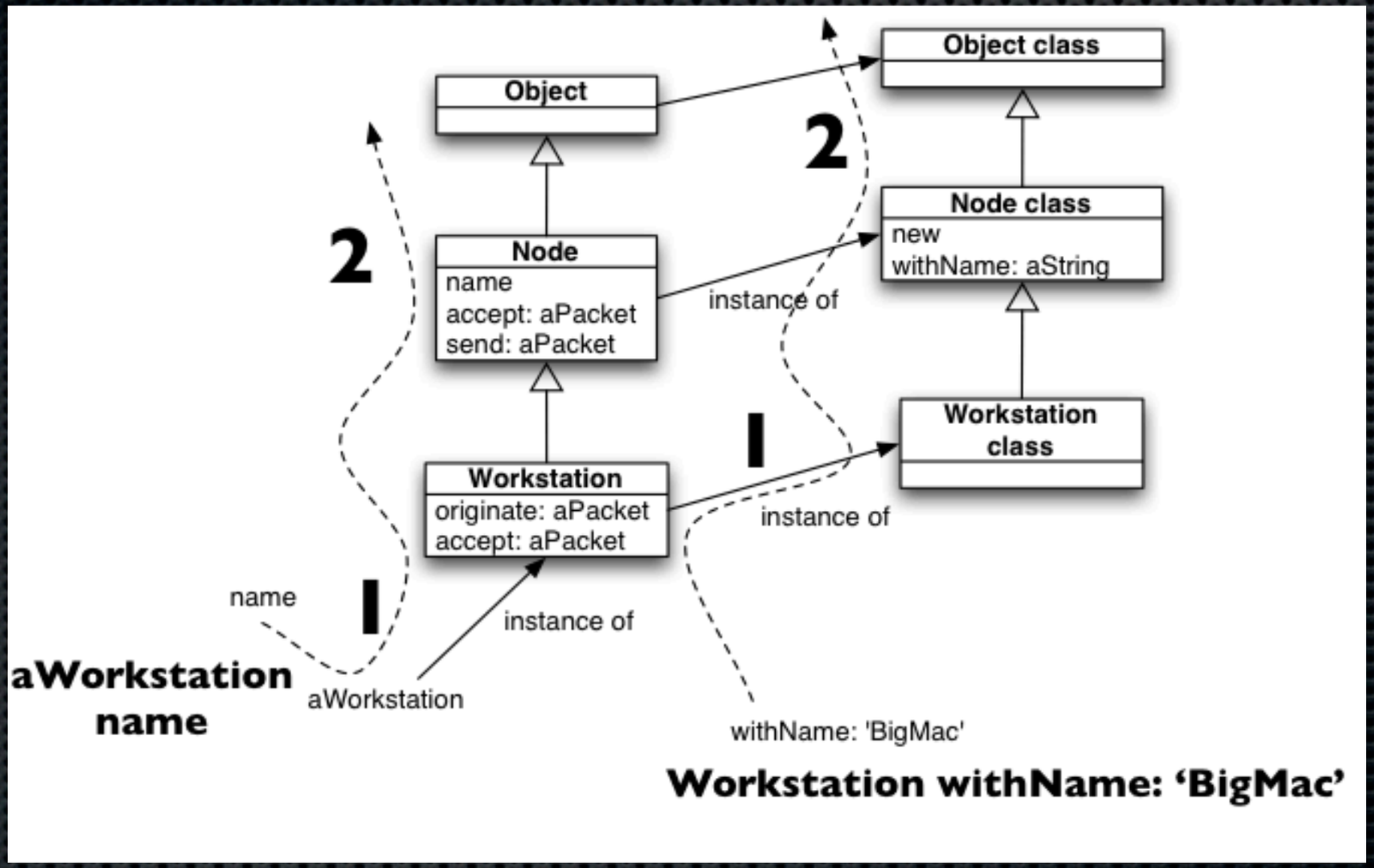
>Point class

“Point class” is an anonymous class with only one instance: Point

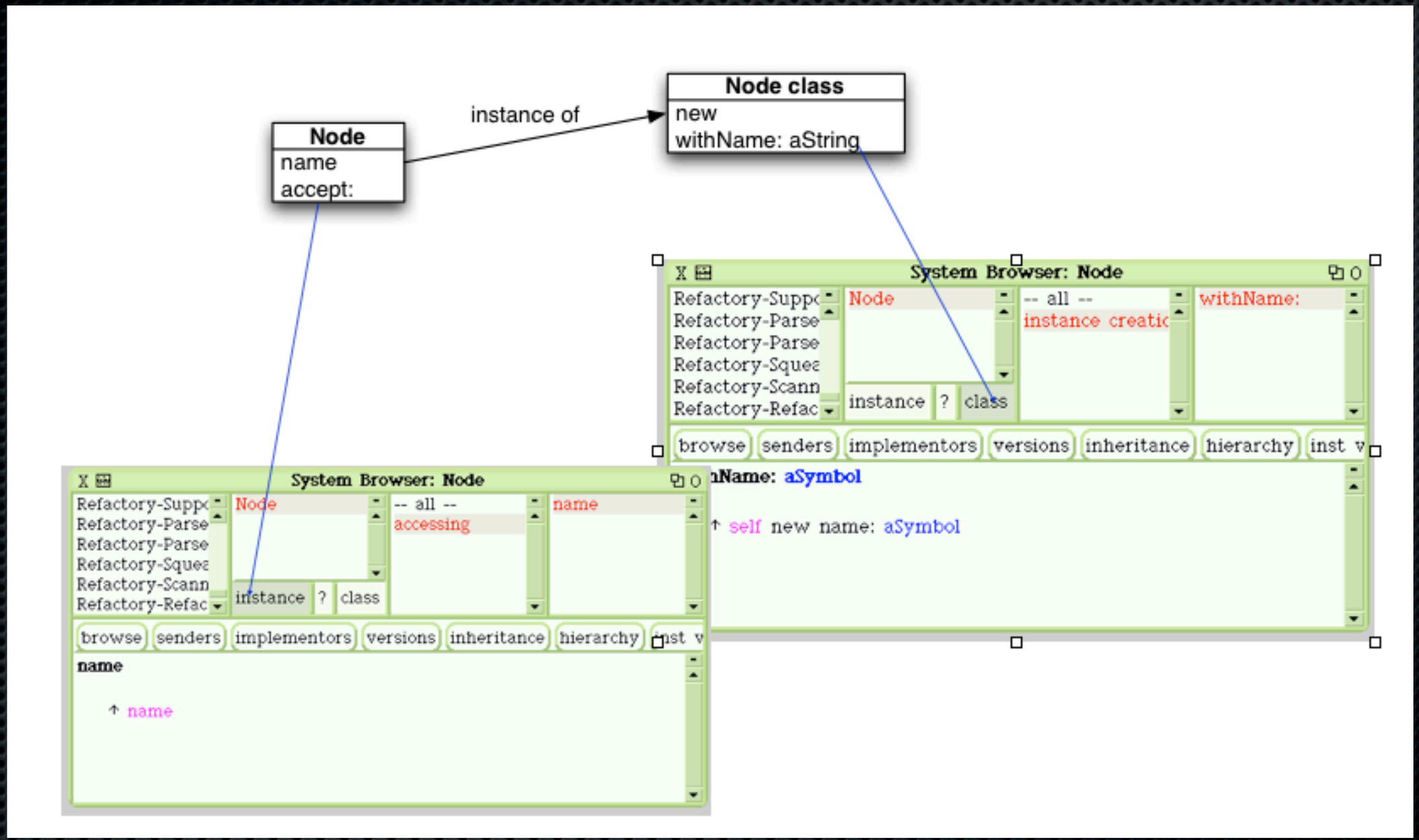
Class Parallel Inheritance



Lookup and Class Methods



About the Buttons



Class methods are plain late bound methods as any methods!

Package extensions

- A method can be defined in a class that is packaged in ***another*** package!
- Powerful to build layers

2 D20: two dice of 20 faces

Defined in the Dice package

```
Integer>>D20
	^ self D: 20
```

```
Integer>>D: anInteger
	| h |
	h := DiceHandle new
	self timesRepeat:
		[h addDice: (Dice faces: anInteger)].
	^ h
```

Summary

- Everything is an object
- Single inheritance, public methods, protected attributes
- One single model
  - Classes are simply objects too
  - A class is instance of another class
  - One unique method lookup, look in the class of the receiver