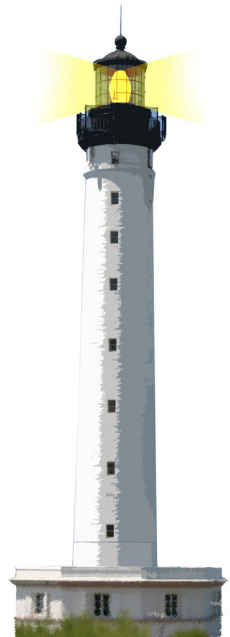
Reflection: Basic Introspection

Damien Cassou, Stéphane Ducasse and Luc Fabresse

W5S08

Pharo

http://www.pharo.org



What Definitions Say: Reflection

Reflection is the ability of a program to manipulate as data something representing the state of the program during its own execution.

- **Introspection** is the ability of a program to **observe** and therefore **reason** about its own state.
- **Intercession** is the ability of a program to **modify** its own execution state or alter its own interpretation or meaning.

What Definitions Say: Reification

Reification is the process to transform implicit to explicit (objects)

- e.g., getting the stack as an object
- e.g., getting a class as an object

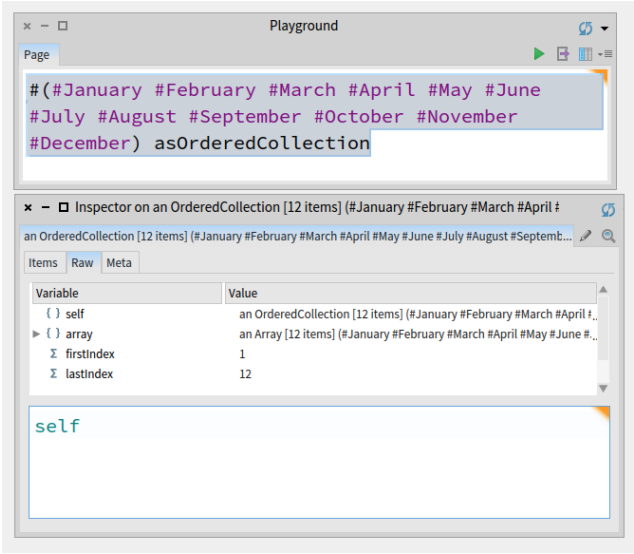
Pharo is a Reflective System

A system having itself as application domain and that is causally connected with this domain can be qualified as a reflective system [Pattie Maes, OOPSLA'87]

- A reflective system has an internal representation of itself
- A reflective system is able to act on itself with the ensurance that its representation will be causally connected (i.e. up to date)

Inspector

How does it access internal object state?



State Introspection

Accessing and setting object state

```
Object >> instVarAt: aNumber
Object >> instVarAt: aNumber put: anObject
Object >> instVarNamed: aString
Object >> instVarNamed: aString put: anObject
```

Accessing/Setting State Example

```
pt := 10@3.
pt instVarNamed: 'x'.
> 10
pt instVarNamed: 'x' put: 33.
pt
> 33@3
```

- Violates encapsulation
- But this is for tools and during development

Accessing Class

```
Object >> class
```

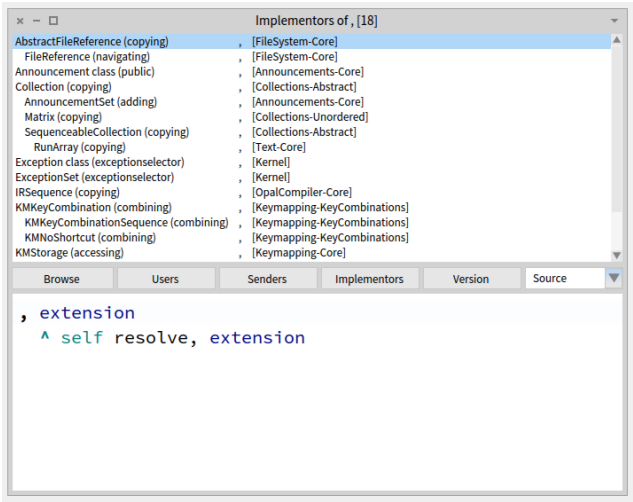
```
'hello' class
(10@3) class
Smalltalk class
Class class
Class class class
Class class class class
```

Querying the System

```
OrderedCollection allSuperclasses size.
OrderedCollection allSelectors size.
OrderedCollection allInstVarNames size.
OrderedCollection selectors size.
OrderedCollection instVarNames size.
OrderedCollection subclasses size.
OrderedCollection allSubclasses size.
OrderedCollection linesOfCode.
```

Querying the System

```
SystemNavigation default browseAllImplementorsOf: #,
```



Sending a Message by its Name

- How to implement a menu or a button?
- Need to send a message to a receiver given a message selector

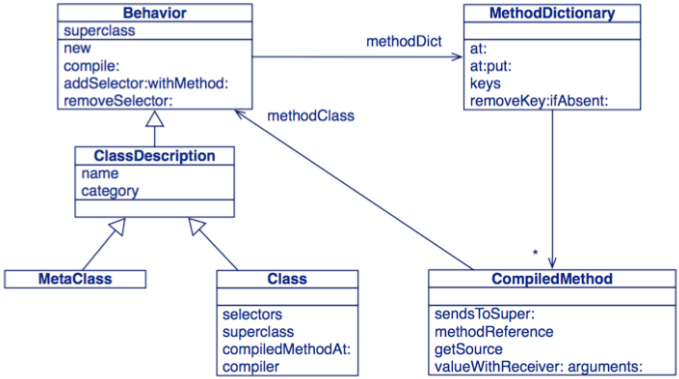
Sending a Message by its Name

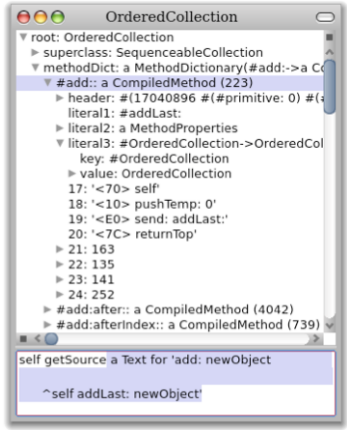
```
Object >> perform: aSymbol
Object >> perform: aSymbol with: arg
```

- Asks an object to execute a message
- Normal lookup is performed

```
5 factorial
5 perform: #factorial
```

Classes Hold Compiled Methods





Executing a Compiled Method

```
CompiledMethod >> valueWithReceiver:arguments:
```

- no lookup performed

```
(Integer>>#factorial)
  valueWithReceiver: 5 arguments: #()
```

```
(SmallInteger>>#factorial)
  valueWithReceiver: 5 arguments: #()
```

Summary

- Just a part of the reflective power!
- Everything is an object and can be introspected
- Grab objects and talk to them
- Have a look at inspector code

A course by

and

in collaboration with