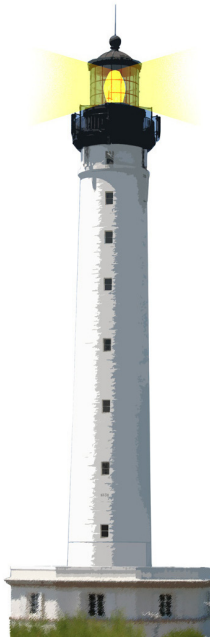
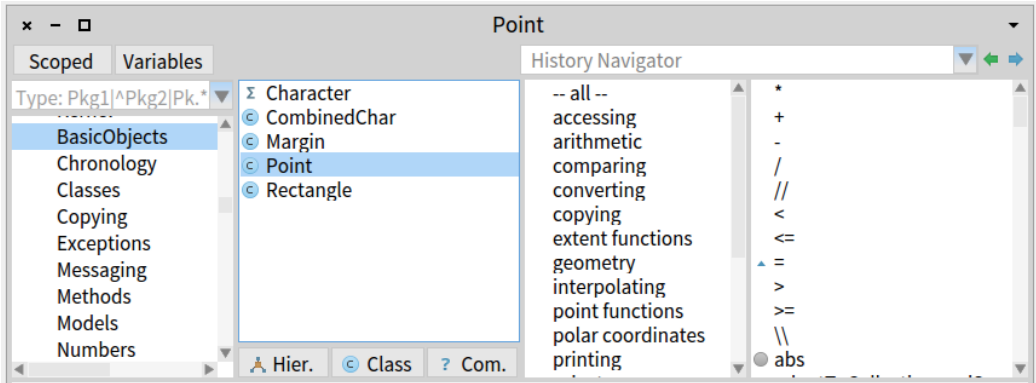
Class and Method Definitions



Class and Method Definitions in Pharo

- classes and methods are defined within tools
- there is no dedicated syntax

Class Definition in Pharo



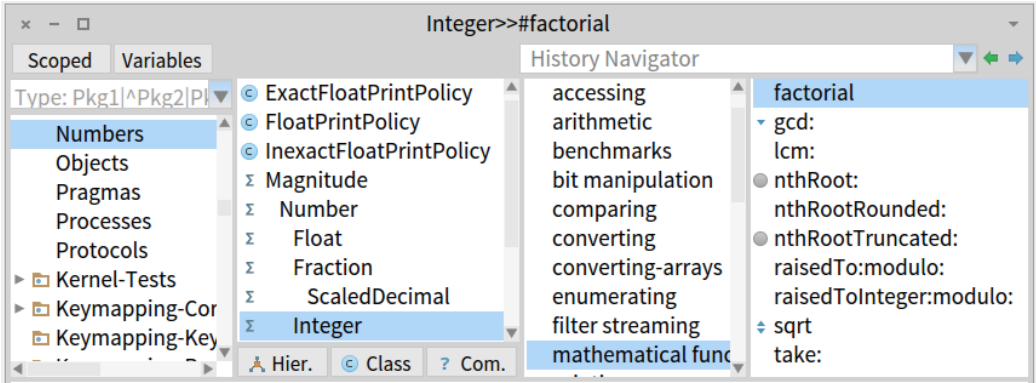

```
Object subclass: #Point
  instanceVariableNames: 'x y'
  classVariableNames: ''
  category: 'Kernel-BasicObjects'
```

Class Definition is a Message

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

We send the message subclass:inst.... to the superclass to create the class

Method Definition in Pharo



Method Definition in Pharo

```
factorial
  "Answer the factorial of the receiver."
  self = 0 ifTrue: [ ^ 1 ].
  self > 0 ifTrue: [ ^ self * (self − 1) factorial ].
  self error: 'Not valid for negative integers'
```

In which class is factorial defined?

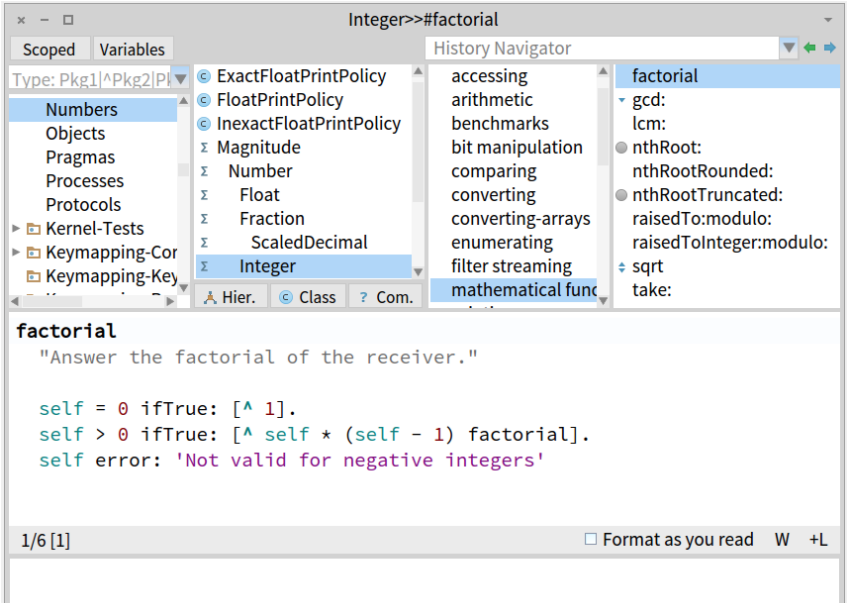
Presentation Convention

In this lecture, a method will be displayed as

```
Integer >> factorial
  "Answer the factorial of the receiver."
  self = 0 ifTrue: [ ^ 1 ].
  self > 0 ifTrue: [ ^ self * (self − 1) factorial ].
  self error: 'Not valid for negative integers'
```

- **Integer »** is not part of the syntax
  - it tells you the method's class

Presentation Convention



Remember Messages

```
Integer >> factorial
  "Answer the factorial of the receiver."

  self = 0 ifTrue: [ ^ 1 ].
  self > 0 ifTrue: [ ^ self * (self − 1) factorial ].
  self error: 'Not valid for negative integers'
```

- factorial is the method name
- =, >, * and - are binary messages
- factorial is an unary message
- ifTrue: and error: are keyword messages
- the caret ^ is for returning a value

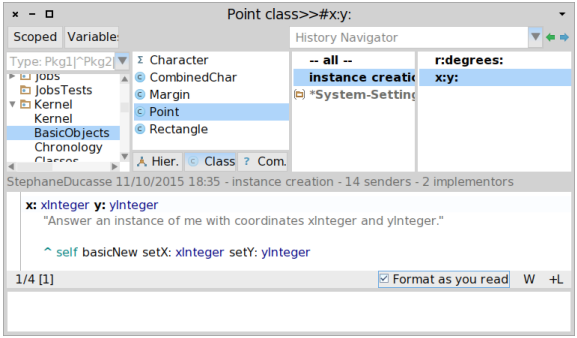
A Method Returns self by Default

```
Game >> initializePlayers
  self players
    at: 'tileAction'
    put: ( MITileAction director: self )
```

is equivalent to

```
Game >> initializePlayers
  self players
    at: 'tileAction'
    put: ( MITileAction director: self ).
  ^ self      "<-- optional"
```

Class Methods



- press the button class to define a class method
- in lectures, we add class

```
Point class >> x: xInteger y: yInteger
  "Answer an instance of me with coordinates xInteger and yInteger."

  ^ self basicNew setX: xInteger setY: yInteger
```

What You Should Know

- A class is defined by sending a message to its superclass
- Classes are defined inside packages
- Methods are public
- By default a method returns the receiver, self
- Class methods are just methods of the class side

A course by

S. Ducasse, G. Polito, and Pablo Tesone