

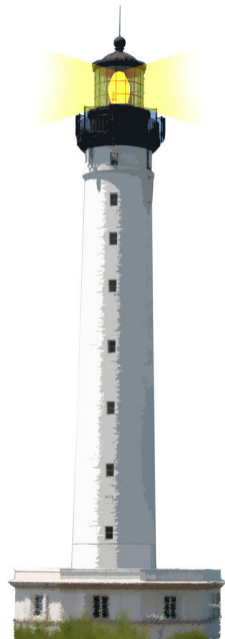
# Class Methods

Damien Cassou, Stéphane Ducasse and Luc Fabresse

W3S06



<http://www.pharo.org>



# Class Methods

1. in Pharo, everything is an object
2. objects can receive messages
3. classes are objects too

Classes can receive messages



# Examples

Time now

> 9:18:36.304688 pm

The message now is sent to the class Time

Date today

> 29 July 2015

The message today is sent to the class Date

# Examples

`FileLocator` `workingDirectory`

`ZnEasy` `getPng`: `'http://pharo.org/web/files/pharo.png'`

`ZnServer` `startDefaultOn`: 8080

# Class Methods are Defined on Class Side

Note the Class button pressed!

The screenshot shows an IDE window titled "Date class>>#today". The interface is divided into several panes:

- Left Pane:** A tree view showing the class hierarchy. Under "Morphic/Settings-", "Chronology" is selected. Below it are "Classes", "Copying", "Exceptions", "Messaging", "Methods", and "Models".
- Middle Pane:** A list of classes under "Date". The classes listed are DateParser, DosTimestamp, Duration, Stopwatch, Time, Timespan, Date (highlighted), and Month. At the bottom, there are buttons for "Hier.", "Class" (which is pressed), and "? Com."
- Right Pane (History Navigator):** A list of methods. The methods listed are readFrom:, readFrom:pattern:, starting:, today (highlighted), tomorrow, week:day:, year:day:, year:month:day:, and year:week:day:.
- Main Editor:** Shows the source code for the "today" method:

```
today  
  
^ self current
```
- Bottom Bar:** Shows "1/3 [1]" and a checkbox for "Format as you read" which is unchecked, along with "W" and "+L" buttons.

# Common Mistake

```
Counter class >> withValue: anInteger  
  self new  
    value: anInteger;  
    yourself
```

Counter withValue: 10 returns the class Counter instead of a new instance

# Why?

```
Counter class >> withValue: anInteger  
  self new  
    value: anInteger;  
  yourself
```

is equivalent to

```
Counter class >> withValue: anInteger  
  self new  
    value: anInteger;  
  yourself.  
  ^ self
```

self here is the class Counter (the receiver of the message)



# Solution

```
Counter class >> withValue: anInteger  
  ^ self new  
    value: anInteger;  
    yourself
```



# Summary

- Classes are objects
- Messages can be sent to classes too
- Class-side methods are no different from other methods
- Most class-side methods create new instances
- To define a class-side method, press the `class` button



A course by



and



in collaboration with



Inria 2020

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/>