



**Learning Object-Oriented  
Programming and Design with TDD**

# Blocks - the Friends of Conditionals and Loops

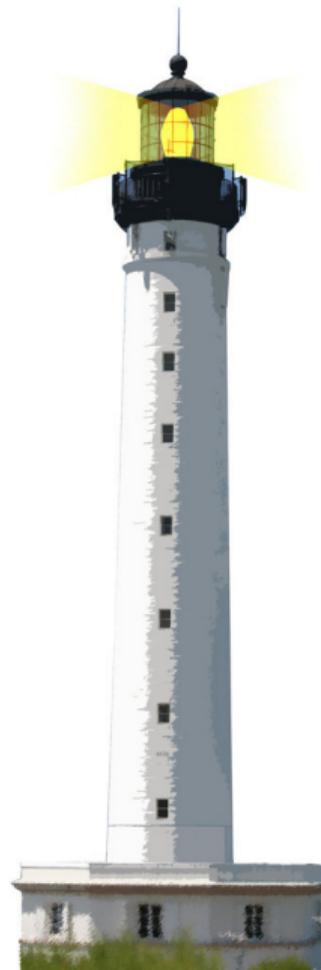
Stéphane Ducasse

<http://stephane.ducasse.free.fr>



<http://www.pharo.org>

**W1S07**



# Remember: A Block Definition Freezes its Body

- Defining a block does **not** execute its body
- E block definition returns a block

```
[ 2 + 6 ]  
>>> [ 2 + 6 ]
```

# Remember: Block Execution

- Executing a Block is explicit

```
[ 2 + 6 ] value  
>>> 8
```

- and repeatable

```
| b |  
b := [ 2 + 6 ].  
b value.  
>>> 8  
b value  
>>> 8
```

# Blocks are Used to Express Conditions

max: anObject

"Answer the receiver or the argument, whichever has the greater anObject."

^ self > anObject

ifTrue: [ self ]

ifFalse: [ anObject ]

Yes this is a message ifTrue:ifFalse: sent to a Boolean

# Blocks are Used to Express Loops

- Some simple loops
- Printing 10 dots

```
10 timesRepeat: [ File stdout << '!' ]  
>>> .....
```

## Blocks are Used to Express Loops (2)

```
1 to: 10 do: [:i | File stdout << i ]  
>>> 12345678910
```

## Blocks are Used to Express Loops (3)

- A traditional for loop for  $i=1,100, i++$

```
1 to: 100 by: 3 do: [:i | File stdout << i ]  
>>> 147101316192225283134374043464952555861646  
770737679828588919497100
```

- The message `to:by:do:` is sent to an integer
- `i` will get all the computed values one by one

# Blocks are Used For Iterators

- Basis for iterators

```
#(2 4 5 -4 3 -2) collect: [ :each | each abs ]  
>>> #(2 4 5 4 3 2)
```

- Here the message is sent to the collection itself
- See Lecture on Iterators

# Yes ifTrue:ifFalse: is a message!

```
Weather isRaining  
  ifTrue: [ self takeMyUmbrella ]  
  ifFalse: [ self takeMySunglasses ]
```

- Conceptually ifTrue:ifFalse: is a message sent to an object: a boolean!
- ifTrue:ifFalse: is in fact radically optimized by the compiler

# Implementation Note

- Note that the Virtual Machine shortcuts calls to Boolean such as condition for speed reason
- But you can implement your own conditional methods and debug to see that sending a message is dispatching to the right object
- Implement your own control structure such as `siAlors:sinon:` (in French) and try it

# Summary

- Blocks freeze and control computation
- Basis for
  - conditionals
  - loops / iterators
  - exceptions (see Mooc lectures)
  - concurrence



# Resources

- Pharo Mooc - W2S06 Videos
- Pharo by Example <http://books.pharo.org>
- Deep into Pharo <http://books.pharo.org>

A course by Stéphane Ducasse  
<http://stephane.ducasse.free.fr>

Reusing some parts of the Pharo Mocc by

Damien Cassou, Stéphane Ducasse, Luc Fabresse  
<http://mocc.pharo.org>



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/>