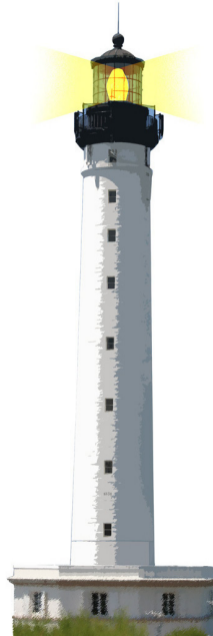


Advanced Object-Oriented Design

Loops



<http://www.pharo.org>



Loops

- Loops are expressed as messages
- Many different ones
 - Plain loops
 - Conditional loops
- Messages sent to numbers, collections or blocks
- Iterators



Loops: timesRepeat:

To repeat a given number of times an action

```
4 timesRepeat: [ self doSomething ]
```

Loops: to:do:

```
1 to: 100 do:  
  [:i | ... i ...]
```

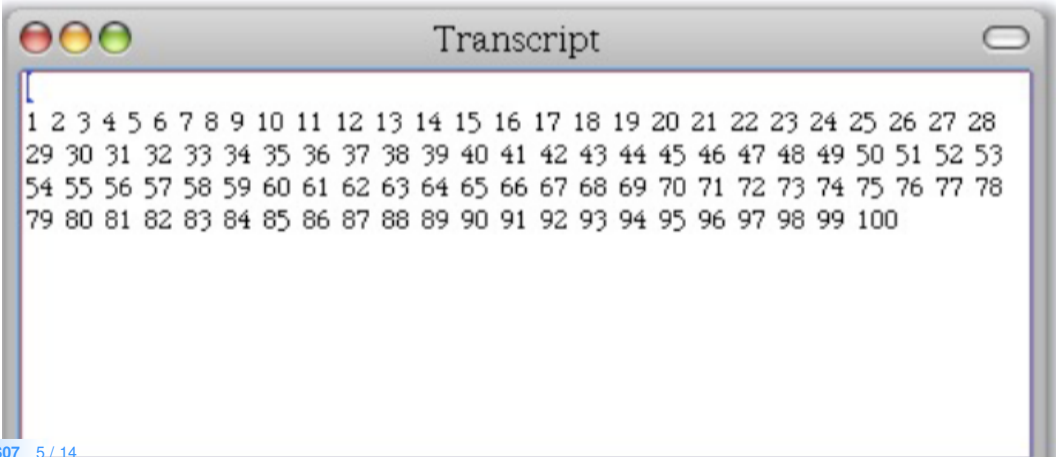
to:do: is a method defined on Number



Example: to:do:

The block is executed with the temporary `i` taking values from 1 to 100

```
1 to: 100 do:  
  [:i | Transcript show: i ; space ]
```



```
Transcript  
[  
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28  
29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53  
54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78  
79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
```

Loops: to:by:do:

```
0 to: 100 by: 3 do: [:i | ... i ... ]
```

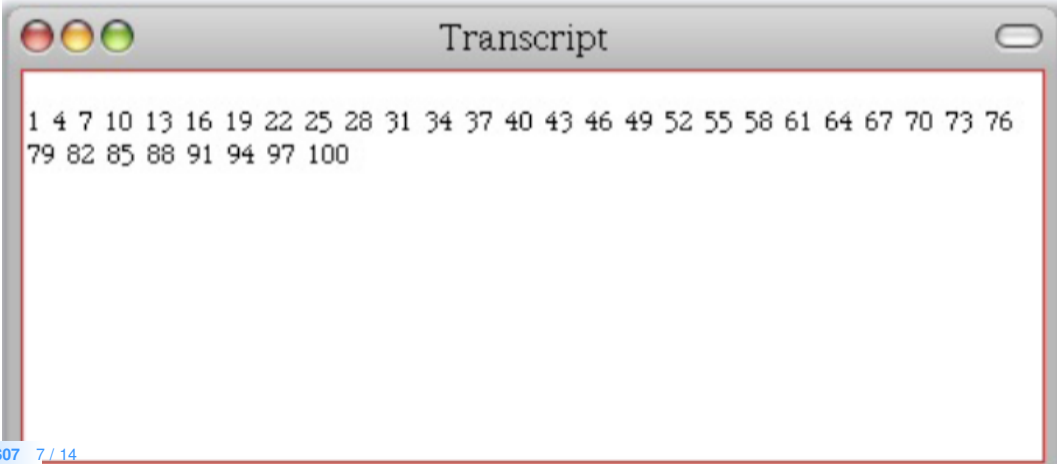
to:by:do: is also a method defined on Number



Example: to:by:do:

The block is executed with *i* taking values from 1 to 100 by step of 3

```
1 to: 100 by: 3 do:  
  [:i | Transcript show: i ; space ]
```



The screenshot shows a window titled "Transcript" with a red border. The window contains the following text:

```
1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46 49 52 55 58 61 64 67 70 73 76  
79 82 85 88 91 94 97 100
```

Basic Iterators Overview

- do: (iterate)
- collect: (iterate and collect results)
- select: (select matching elements)
- reject: (reject matching elements)
- detect: (get first element matching)
- detect:ifNone: (get first element matching or a default value)
- includes: (test inclusion)
- and a lot more...



Loops: do:

```
aCol do: [ :each | ... ]
```

The block is executed with `each` taking as value all the elements of `aCol`



Example: The iterator do:

```
 #(15 10 19 68) do:  
   [:i | Transcript show: i ; cr ]
```



Loops: whileTrue:

```
[ ... ] whileTrue: [ ... ]
```

Executes the argument, aBlock, as long as the value of the receiver is true

```
Color >> atLeastAsLuminantAs: aFloat  
| revisedColor |  
revisedColor := self.  
[ revisedColor luminance < aFloat ]  
  whileTrue: [ revisedColor := revisedColor slightlyLighter ].  
^ revisedColor
```



Loops: whileTrue

Executes the receiver, as long as its value is true

```
[...] whileTrue
```

Equivalent with whileFalse **and** whileFalse:



Summary

- Loops are expressed as messages
- Many different ones
 - Plain loops
 - Conditional loops
- Messages sent to numbers, collections or blocks
- Iterators



A course by

S. Ducasse, G. Polito, and Pablo Tesone



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/>