

Understanding Messages: Sequence and Cascade

Damien Cassou, Stéphane Ducasse and Luc Fabresse

W2S04



<http://www.pharo.org>



Expression Sequence

. is a separator

```
expression1.  
expression2.  
expression3
```

Example

```
Transcript cr.  
Transcript show: 1.  
Transcript show: 2
```

Expression Sequence

- . is a separator, not a terminator
- no need to put one at the end
- no point after temporary variable declaration

```
| macNode pcNode |  
macNode := Workstation withName: #mac.  
macNode sendPacket: 'Hello World'
```

Cascade: Sending Multiple Messages to an Object

```
Transcript cr.  
Transcript show: 1.  
Transcript show: 2
```

is equivalent to:

```
Transcript  
  cr ;  
  show: 1 ;  
  show: 2
```

- ; is called a cascade



Cascade Example

Sending Multiple Messages to an Object

```
| c |  
c := OrderedCollection new.  
c add: 1.  
c add: 2
```

is equivalent to:

```
OrderedCollection new  
  add: 1 ;  
  add: 2
```

- add: 2 is sent to the receiver of message add: 1
- this receiver is the instance of OrderedCollection



What You Should Know

- . is a separator
- ; (cascade) is useful to avoid repeating the receiver
- the cascade returns the last message returned value



A course by



and



in collaboration with



Inria 2016

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/>