

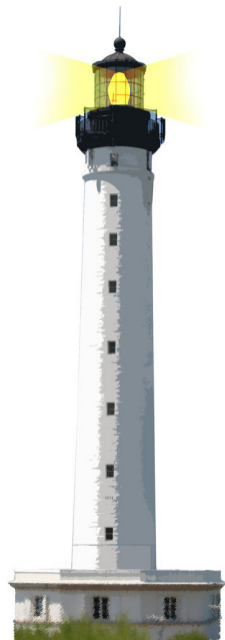
Yourself

Damien Cassou, Stéphane Ducasse and Luc Fabresse

W2S10



<http://www.pharo.org>



A Puzzle and Candidate for Cascade

We add 2 to a set

```
Set new add: 2  
> 2
```

We get 2 and not the set!



Why?

```
Set>>add: newObject
```

```
"Include newObject as one of the receiver's elements, but  
only if not already present. Answer newObject."
```

```
[...]
```

```
^ newObject
```

- The method `add:` returns its argument, not the receiver

```
Set new add: 2
```

```
> 2
```



A Verbose Solution

To get the collection back, we can use a temporary variable

```
| s |  
s := Set new.  
s add: 2.  
s
```

yourself

```
Object >> yourself  
  ^ self
```

```
Set new  
  add: 2;  
  yourself  
> aSet
```

- add: and yourself are sent to the new Set
- the cascade (;) returns the returned value of yourself



Another Idiom

```
Set class >> with: anObject  
  "Answer an instance of me containing anObject."  
  | instance |  
  instance := self new.  
  instance add: anObject.  
  ^ instance
```

is equivalent to

```
Set class >> with: anObject  
  "Answer an instance of me containing anObject."  
  ^ self new  
  add: anObject;  
  yourself
```

Using `yourself` makes sure the code returns the new instance



What You Should Know

- Some simple methods are powerful
- Cascade ; and yourself often go together



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/>