

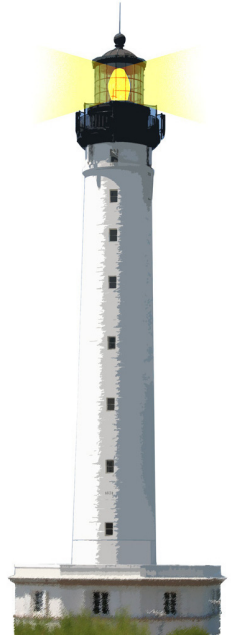
# Really Understanding Class Methods

Damien Cassou, Stéphane Ducasse and Luc Fabresse

W5S02



<http://www.pharo.org>



# What You Will Learn

- There is only one lookup algorithm
  - it works for both instance and class methods
- Class methods are not Java-like static methods



# There is Only One Lookup

The lookup starts in the **class** of the **receiver** then:

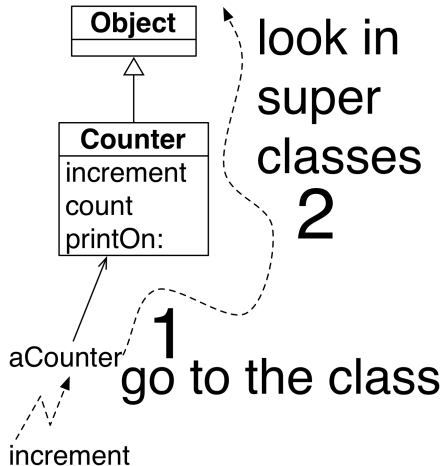
- if the method is defined in the class, it is returned
- otherwise the search continues in the superclass



# Sending a Message

Sending a message is a two-step process:

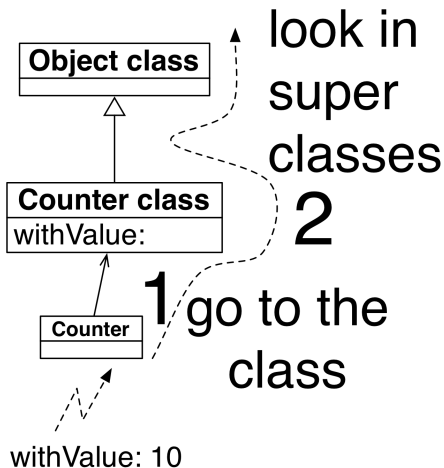
1. look up the method matching the message
2. execute this method on the receiver



# Sending a Message to a Class

Sending a message to  
Counter (a class)

- go to the class of the class (Counter class)
- follow the inheritance chain
- apply the found method to the receiver: the class Counter



# A Class is an Instance

A class is an instance of another class (called a metaclass)

- Counter is the unique instance of the class Counter class
- A metaclass is just one class whose instances are classes
- Counter class is automatically created after Counter
- The class of class XXX is named XXX class



# Browsing two Classes

The browser can edit two classes: a class and its class

The diagram shows a class hierarchy where `aCounter` inherits from `Counter`. A dashed arrow labeled `increment` points from `aCounter` to `Counter`. Another dashed arrow labeled `withValue: 10` points from `Counter` to a box labeled `Counter class`.

The top IDE screenshot shows the `Counter class>>#withValue:` editor. The `History Navigator` on the right lists `withValue:` as an instance creator. The code editor displays the following code:

```
withValue: anInteger  
  ^ self new count: anInteger
```

The bottom IDE screenshot shows the `Counter>>#increment` editor. The `History Navigator` on the right lists `count`, `count:`, `decrement`, `increment`, and `initialize`. The code editor displays the following code:

```
increment  
  count := count + 1
```

# What You Should Know

- A class is an object
- A class can receive messages
- A class is an instance of a class, called a metaclass
- Method lookup works the same
- More during the lecture Understanding Metaclasses





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/>