Inheritance and Lookup

2: Lookup

Damien Cassou, Stéphane Ducasse and Luc Fabresse

http://www.pharo.org
Goal

- Understanding
  - message sending
  - method lookup
  - semantics of self
Inheritance

- Inheritance of state is static
- Inheritance of behavior is dynamic
Message Sending

Sending a message is a two-step process:

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

1 go to the class

2 look in superclasses
Some Lookup Cases

Sending the message color to aColoredRectangle
Some Lookup Cases

Sending the message area to aColoredRectangle
self Always Represents the Receiver

A new foo
> ...
B new foo
> ...

A

B

foo
bar

foo
bar

foo

^ 10
^ self foo
^ 50

aB
self Always Represents the Receiver

A new foo
> 10
B new foo
> 50
What is self/this?

Take 5 min and write the definition of self (this in Java).

- your definition should have two points:
  - what does self represent?
  - how is a method looked up when a message is sent to self?
• **self** represents the receiver of the message
• **self** in Pharo, **this** in Java
• The method lookup starts in the class of the receiver
self Always Represents the Receiver

A new bar
> ...
B new bar
> ...

A
foo
bar
A
foo
^ 10
B
foo
bar
^ self foo
B
foo
^ 50
self Always Represents the Receiver

A new bar > 10
B new bar > 50
Evaluation of aB bar

1. aB’s class is B
2. no method bar in B
3. look up in A - bar is found
4. method bar is executed
5. self refers to the receiver aB
6. foo is sent to self
7. look up foo in the aB’s class: B
8. foo is found there and executed
self Always Represents the Receiver

A new bar
> ...
B new bar
> ...
C new bar
> ...

A
foo
bar

B

C
foo

^ 10

^ self foo

^ 50

aC
self Always Represents the Receiver

```
A
  foo
  bar

B
  bar
    ^ self foo

C
  foo
    ^ 50

A new bar
> 10
B new bar
> 10
C new bar
> 50
```
**What You Should Know**

- **self** always represents the receiver
- Sending a message is a two-step process:
  1. Look up the method matching the message
  2. Execute this method on the receiver
- Method lookup maps a message to a method
- Method lookup starts in the class of the receiver
  - ...and goes up in the hierarchy