Inheritance and Lookup
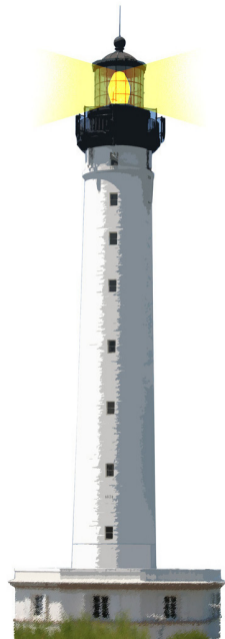
4: doesNotUnderstand: aMessage

Damien Cassou, Stéphane Ducasse and Luc Fabresse

Goal
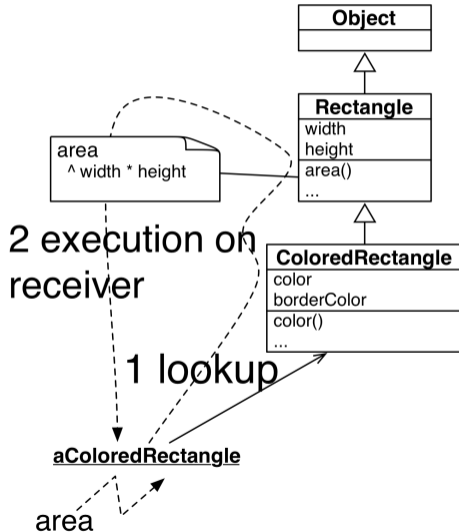
- A hook message is sent when the lookup fails
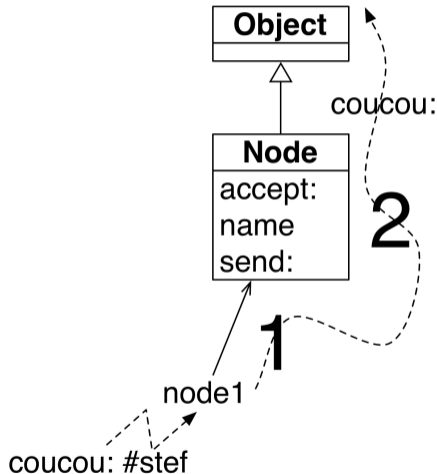- Classes can customize the reaction to this failure

Message Sending

Sending a **message** is a two-step process:

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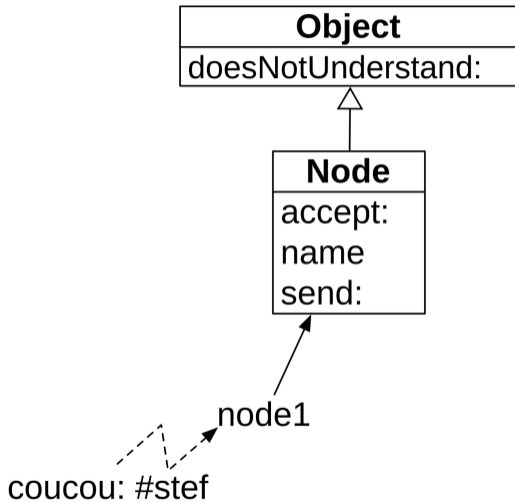2. execute this method on the **receiver**

When No Method is Found

- if no method is found in the hierarchy, *doesNotUnderstand:* is sent to the receiver
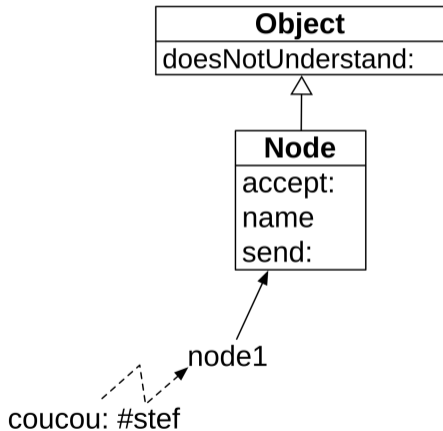- the initial message is passed as parameter

When No Method is Found

Step by Step...

node1 coucou: #stef

1. coucou: **looked up in** Node
2. **not defined in** Node **-> continues in** Object
3. **not defined in** Object **-> send** doesNotUnderstand: **to** node1
4. doesNotUnderstand: **looked up in** Node
5. **not defined in** Node **-> continues in** Object
6. doesNotUnderstand: **is found and executed**

doesNotUnderstand: is a Message

- doesNotUnderstand: is a message
- any class can implement the corresponding method
- used for automatic delegation, proxy, distribution

doesNotUnderstand: and the Debugger

When no class redefines doesNotUnderstand:

- doesNotUnderstand: in Object is executed
  - this method raises a MessageNotUnderstood exception
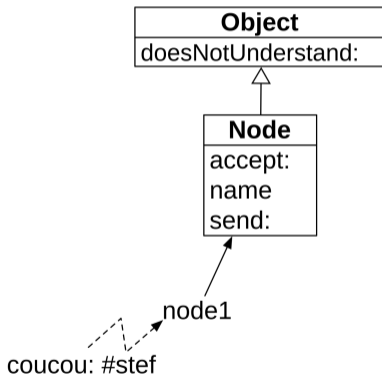 - when not handled, this exception opens a debugger

Catching MessageNotUnderstood Error

MessageNotUnderstood is just an exception, code can handle it:

```
Node >> sayHello
 self coucou: #stef

Node >> welcome
 [ self sayHello ]
 on: MessageNotUnderstood
 do: [ Transcript show: 'Something
       went wrong with a message' ]
```

What You Should Know

- The lookup sends doesNotUnderstand: when it can not find a method
- Each class can implement this method
- The default implementation of doesNotUnderstand: raises an exception that can be handled

A course by

and

in collaboration with