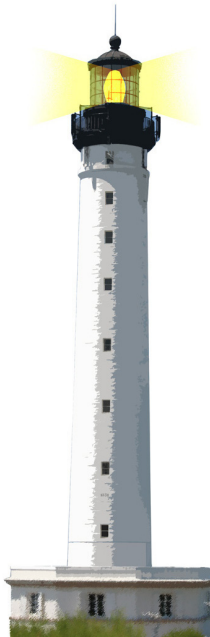
Learning Object-Oriented
Programming and Design with TDD

Methods and Messages

How vs. What

Stéphane Ducasse
http://stephane.ducasse.free.fr



Pharo
http://www.pharo.org

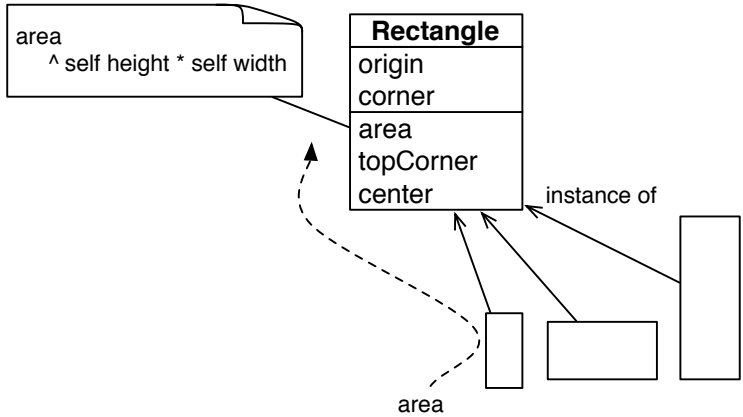
A method

Named sequence of instructions that will be executed on the message receiver

```
FrenchPerson >> makeCrepes
```

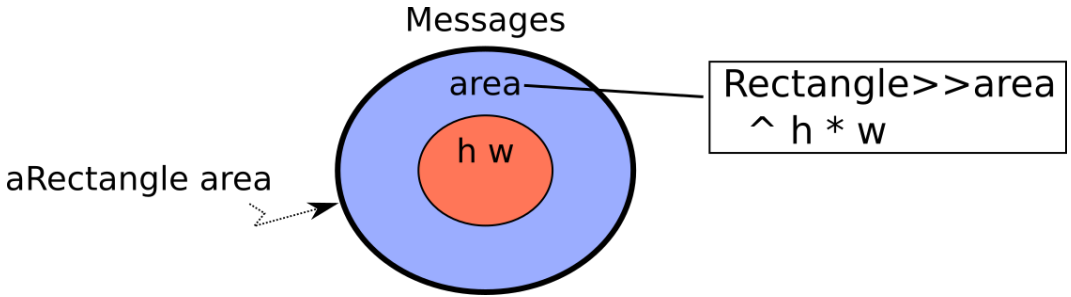
Instances Share the Same Behavior

- All the instances of a class share the same behavior
- A class defines the methods that are executed when an instance receives a message



A method

- Defines **how** to respond to a message
- Is a sequence of executable statements
- Has name that is the same as message name
- Is dynamically selected via method lookup technique
- Returns an object as result of execution



Messages vs. Methods

Message: What to do?

```
stef makeCrepes.
robert makeCrepes.
peter putJamOnCrepes.
```

Methods: How to do it?

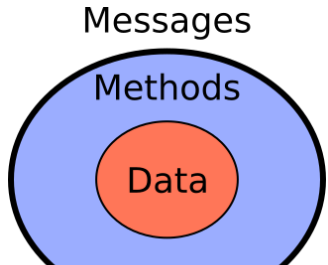
What vs. How

What: Messages

- Specify what behavior objects have to perform
- Details are left to the receiver

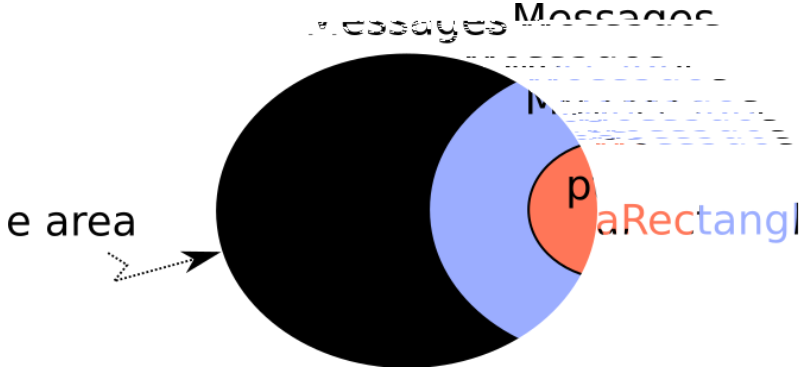
How: Methods

- Specify how an operation is to be performed
- Must have access to data
- Need detailed knowledge of data
- Can manipulate data directly

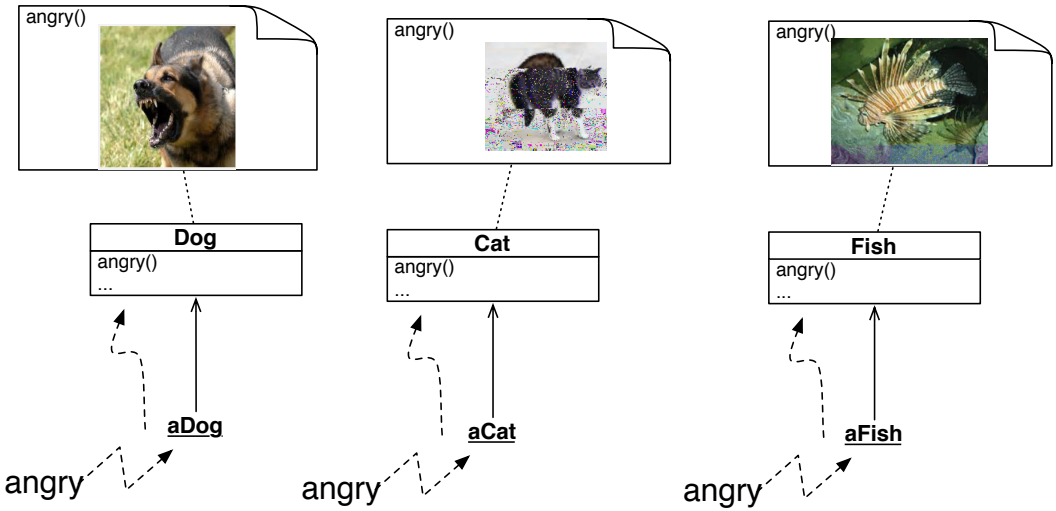


Message

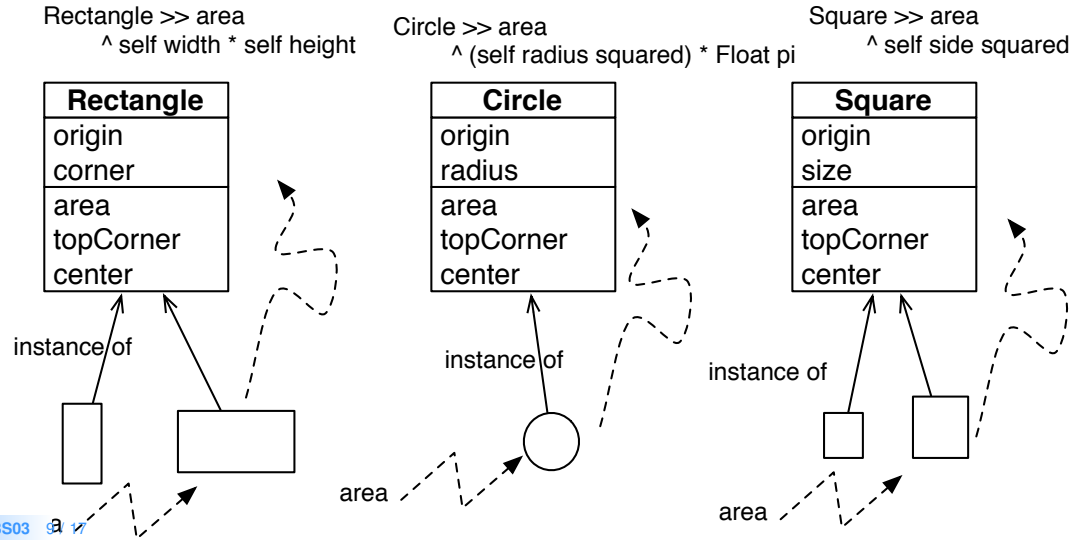
- Sent to an object: The message receiver
- A message may include parameters necessary for performing the action
- Message sends always return a result (an object)
- Only way to communicate with an object and have it perform actions



Methods are looked up dynamically in the class of the receiver



Methods are looked up dynamically in the class of the receiver



Late binding: The method to be executed depends on the receiver

- The executed method depends on the receiver
- Different receivers of the same message may react differently

```
Rectangle >> area
  ^ self width * self height
```

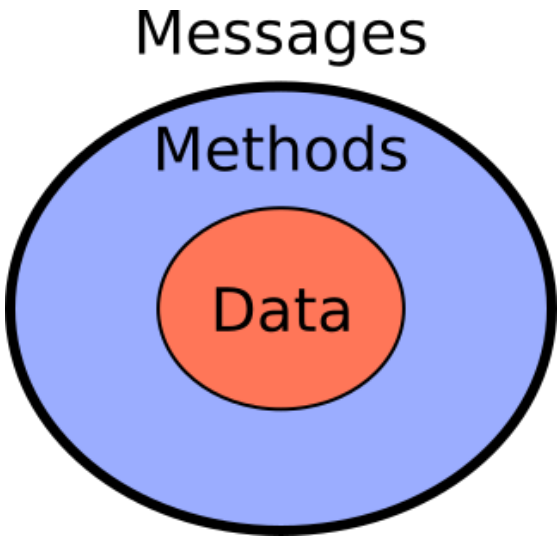
```
Circle >> area
  ^ (self radius squared) * Float pi
```

```
Square >> area
  ^ self side squared
```

Late binding

- The executed method depends on the receiver at runtime
- The receiver is often only known during execution

Data/Messages/Methods

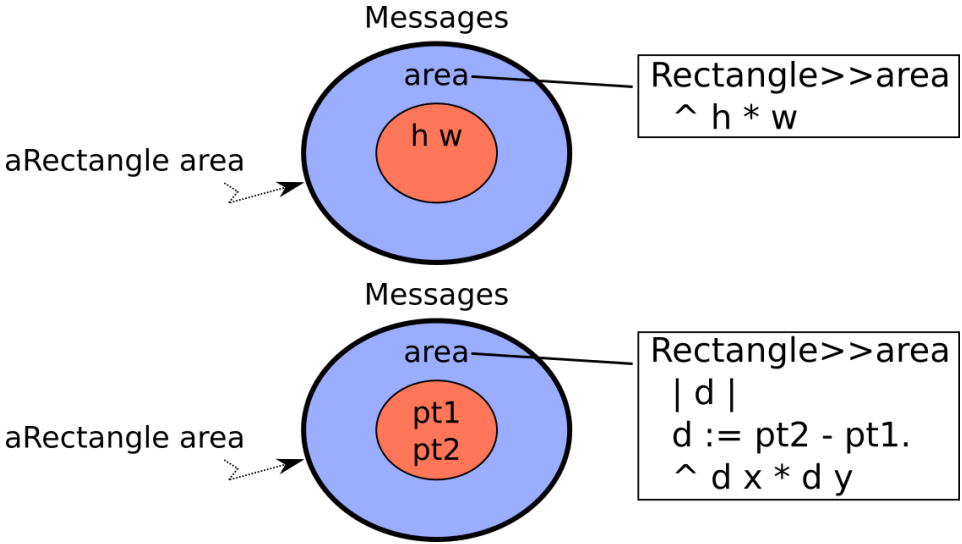


Object Encapsulation

Technique to

- Hide implementation details
- Protect the state of objects from clients
- Communicate/access object via a uniform interface

Object Encapsulation



Object Encapsulation

- Puts objects in control
- Facilitates modularity, code reuse and maintenance
  - External vs. Internal perspective
  - What vs. How
  - Based on Message vs. Method

Summary

- Messages define what should be done.
- Methods define how the computation should be performed.
- The same message sent to different objects may execute different methods.
- Methods are selected dynamically (during execution) based on the receiver.
- Methods are looked up dynamically in the class of the receiver.

A course by Stéphane Ducasse
http://stephane.ducasse.free.fr

Reusing some parts of the Pharo Mooc by

Damien Cassou, Stéphane Ducasse, Luc Fabresse
http://mooc.pharo.org

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/