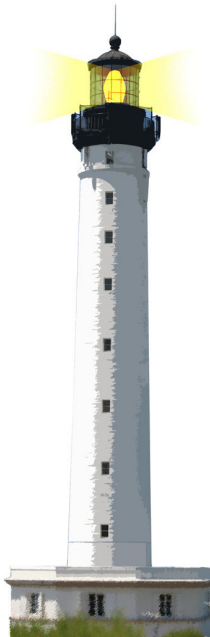
Learning Object-Oriented
Programming and Design with TDD

Key OO concepts in Java

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



http://www.pharo.org

WXSYY

Objectives

- Illustrate key OO concepts
- In Java but limited as much as possible to the essential points

Thanks Alexandre Bergel for parts of the materials used in this lecture!

Quotes of the day

"Perfection is attained, not when no more can be added, but when no more can be removed." Antoine de Saint-Exupéry
"I invented the term ’Object-Oriented’, and I can tell you I did not have C++ in mind." Alan Kay (nor Java :))

What we will not cover

Simple Java is an oxymoron.

- Java is gigantic and even more... (looks more and more like an old verbose language)
- Full of conceptual glitches (public fields, strange protected semantics, overloading...)
- Java mixes physical representation (files) with concepts
  - there is not need to have files to have classes
  - class definitions can be saved in databases

This lecture will not cover: packaging enums, lambdas, generics, inner classes, modules, private methods, visibility, synchronised, meta data, overloading, primitives vs. boxed....

- But we will provide extras slides on more advanced topics

Java

Not pure object-oriented programming language

- Static methods are not looked up
- Primitive types are not objects: int and Integer....
- Classes are not first class: cannot send messages to classes

Outline

- Instances, instance creation
- Classes / instance variables
- Methods
- Inheritance (single)
- Interface
- Cast
- Dynamic type vs. static types
- Constructor

Instances

- Created using new construct
- Remember: one state, identity, behavior

```
new Tomagotchi()
```

Often

```
Tomagotchi t = new Tomagotchi()
```

Class

- Mold/Generators of instances

```
public class Rectangle
  {
     protected int length;
     protected int width;
    ...
```

```
public class Box extends Rectangle {
     protected int height;
}
```

Class

- One public class per file
- File name should have the name of the public class
- Class import packages (group of classes)

Instance variables

- Describe instance structure
- Have a visibility: Avoid public, private and final :)
- Better use protected (see companion extra lectures)

```
public class Rectangle {
    protected int length;
    protected int width;
   ...
```

- Accessible by method of the class and subclasses

Methods

- this represents the receiver
- the lookup of methods at runtime starts in the class of the receiver.

```
public class Rectangle{
  protected int length;
  protected int width;

  public int getArea() {
    return length * width;
  }
```

Constructor (I)

- A Constructor is a static function, it is not a method!
- Responsible to properly initialize an object
- <class>() is a default constructor

```
public Rectangle() {
  length = 0;
  width = 0;
}
```

Multiple constructors in a class

```
public Rectangle(int length, int width) {
  this.length = length;
  this.width = width;
}
```

What you should know

- Class/Instances
- Methods
- Constructors = functions

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