

OOP in ActionScript 1, 2 & 3

A learn by example tutorial

Hans Van de Velde



Introduction

Goal = teach basics of OOP in Flash
Theory + examples (very condensed)

<http://www.novio.be/blog/?p=621>



Prerequisites

Some basic programming skills

You should know stuff like:

→ variables

→ functions

→ conditions (if clause, switch case, etc.)

→ loops (for loop, while loop, etc.)



Course outline

- The "object"
- Welcome to the class
- Inheritance and polymorphism
- Composition
- Class Interface
- Exception handling
- Event handling and dispatching



The “object”

Everything is an object

Data abstraction: properties and methods

An object is an instance of a class

ex. make abstraction of a bicycle



Welcome to the class

A class is a blueprint

Definitions:

- Constructor

- Properties and methods

- Public / private / protected accessors

ex. Person class [01_Class.zip]

Excercise: make a class Cat & create 2 instances



Inheritance

= inherit properties and methods from a superclass

Class hierarchy (hierarchical classification)

Vehicle > Bicycle

Mamals > Apes > Human

Variations

Polymorphism

ex. [02_Inheritance.zip] extending the MovieClip
polymorphism (subtypes of Shape)



Composition

“HAS A” versus “IS A”

How to choose between inheritance & composition?

- if you need services from another class: use composition
- if a class behaves very much like another class: consider inheritance
- if you can benefit from polymorphism: use inheritance

ex. [03_Composition.zip] “has a” MovieClip



Class Interface

A contract with a class,
definition of methods that need to be implemented

As if it's the datatype (marker interface)

→ enables exchangeable implementations

ex. [04_Interface.zip]



Exception handling

Errors abort code execution

→ put try – catch blocks around things that can go wrong

→ finally always executes

Errors bubble up

Custom errors and multiple catch blocks

ex. [05_Exceptions.zip]



Event-driven programming

Events = ?

GUI events: mouse, keyboard, selections, timers, etc.

File & Network: loading, remote procedure calls, etc.

System messages, stage events, etc.

Events determine flow in a program

Event is created

processed by `EventDispatcher`

associated with handler of objects that are showing interest

Loose coupling



Event handling

Listen & handle

ex. [06_EventHandling.zip]



Event dispatching

Custom events

Subclass Event class

ex. [07_EventDispatching.zip]



```
// The End
```

```
gotoAndPlay();
```

