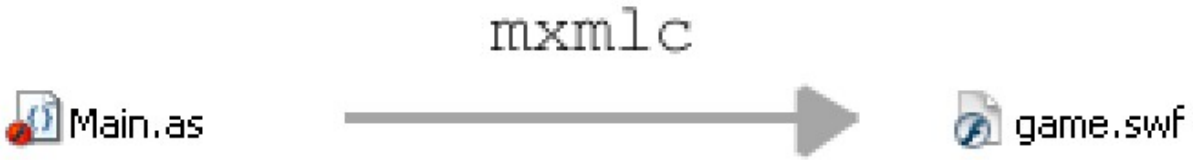
haXe for Flash Game Dev.

Nicolas Cannasse, Motion-Twin
http://ncannasse.fr
March 8, 2010

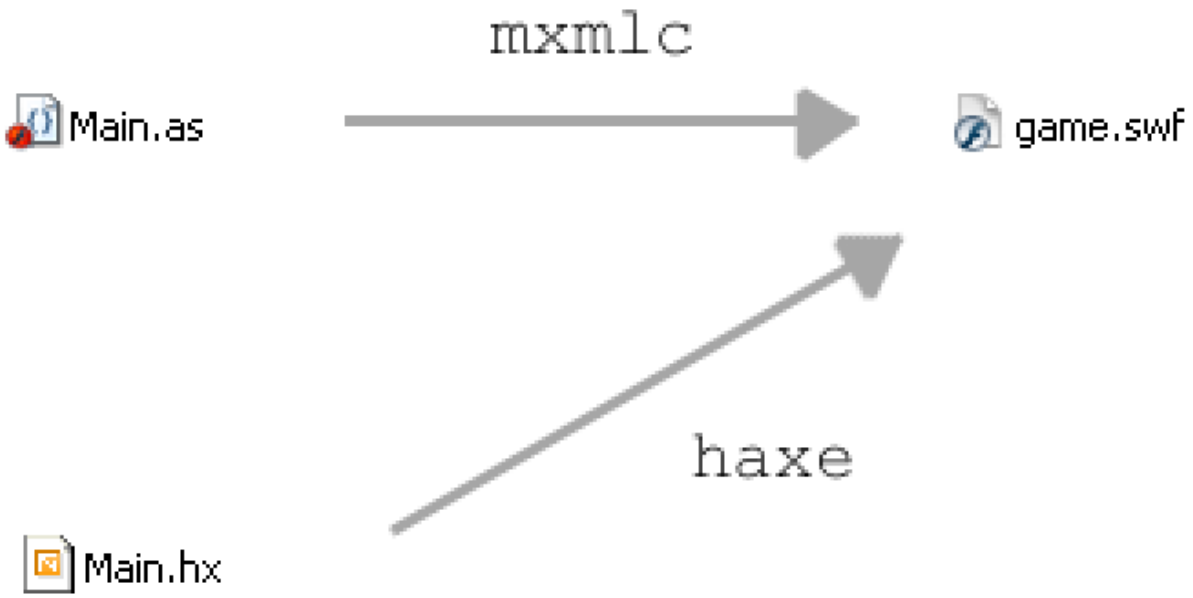
WARNING !!!

code(); // ahead !

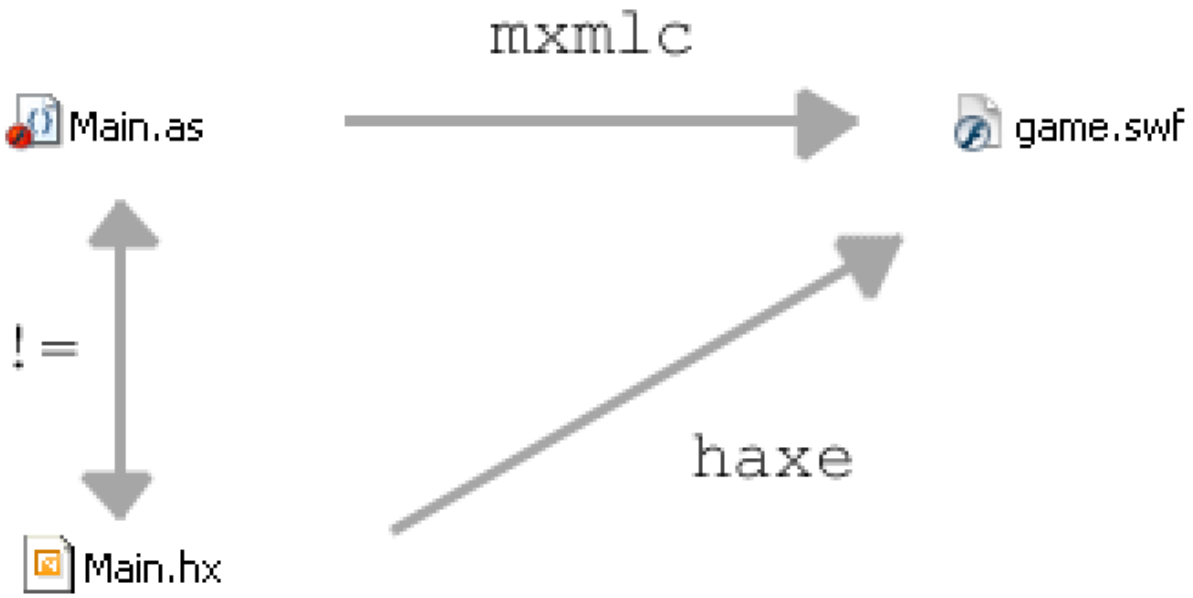
Compiler ?



Compiler ?



Compiler ?



Compiler ?

Before haXe : MTASC

haXe@FGS

D.R.Y.

Don't **R**epeat **Y**ourself

DRY

```
1  var a : int = 1 + 3;
2  var x : String = "I'm a string";
3  var y : Vector.<DisplayObject> = new Vector.<DisplayObject>();
```

DRY

```
1  var a : int = 1 + 3;
2  var x : String = "I'm a string";
3  var y : Vector.<DisplayObject> = new Vector.<DisplayObject>();
```

DRY

```
1  var a = 1 + 3;
2  var x = "I'm a string";
3  var y = new Vector<DisplayObject>();
4  var z = x.substr(1,3);
```

Type Inference

Data Structures

- AS : Array, Dictionary, Vector
  - (+) native speed
  - (-) not customizable
- Algorithms ~= Data Structures
- How to express your own (abstract) DS ?

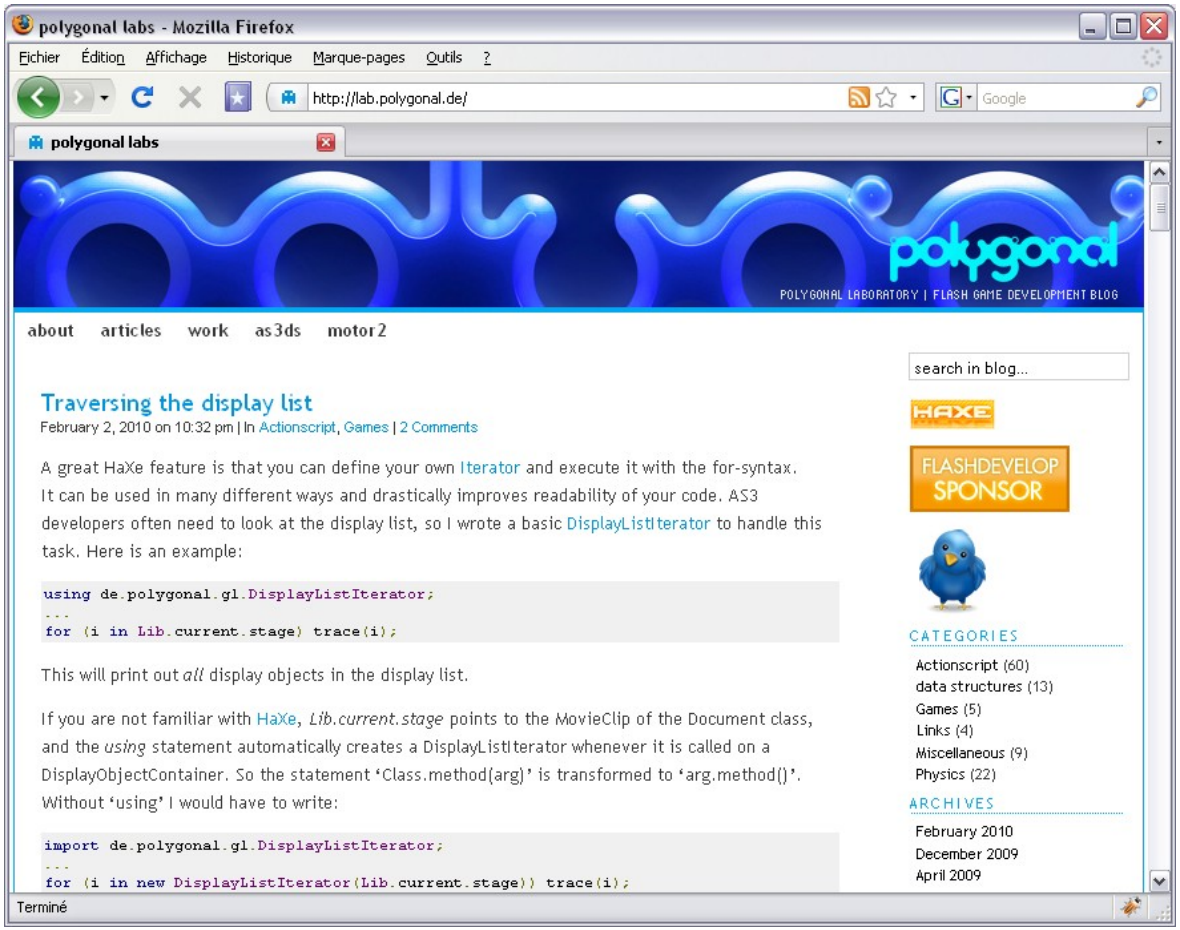
Generics

```
class Foo {
   var arr : Array<Sprite>;
   function new() {
      arr = new Array();
   }
}
```

Generics

```
1  class Cell<T> {
2      public var elt : T;
3      public var next : Cell<T>;
4      public function new() {
5      }
6  }
```

Generics in use



http://lab.polygonal.de

haXe@FGS

K.I.S.S.

Keep it **S**imple, **S**tupid

Enums

```
1  enum Color {
2      Red;
3      Green;
4      Blue;
5  }
```

Enums

```
1  var x = Red;
2  switch( x ) {
3  case Red:
4       // do something
5  case Green, Blue:
6       // something else
7  }
```

Enums

```
1  enum Color {
2      Red;
3      Green;
4      Blue;
5      Grey( amount : Int );
6  }
```

Enums

```
1  var x = Grey(20);
2  var hex = switch( x ) {
3  case Red: 0xFF0000;
4  case Green: 0x00FF00;
5  case Blue: 0x0000FF;
6  case Grey(x): x | (x << 8) | (x << 16);
7  }
```

Enums

In action !

PBJ Assembler

Iterators

```
1  for( var i = 0; i < arr.length; i++ ) {
2      var x : MyClass = arr[i];
3  }
```

Iterators

```
1  for( var i = 0; i < arr.length; i++ ) {
2      var x : MyClass = arr[i];
3  }
```

Iterators


```
1  for( x in arr ) {
2      //...
3  }
```

Iterators

```
1  var it = arr.iterator();
2  while( it.hasNext() ) {
3      var x = it.next();
4      // ....
5  }
```

Iterators

- Use them as you wish
  - By simply adding `.iterator()` method
- Optimized for common cases
  - Array, Vector, ...
- Iterator + Generics + Functional = Lambda

haXe@FGS

I.N.M.S.

I Need **M**ore **S**peed !

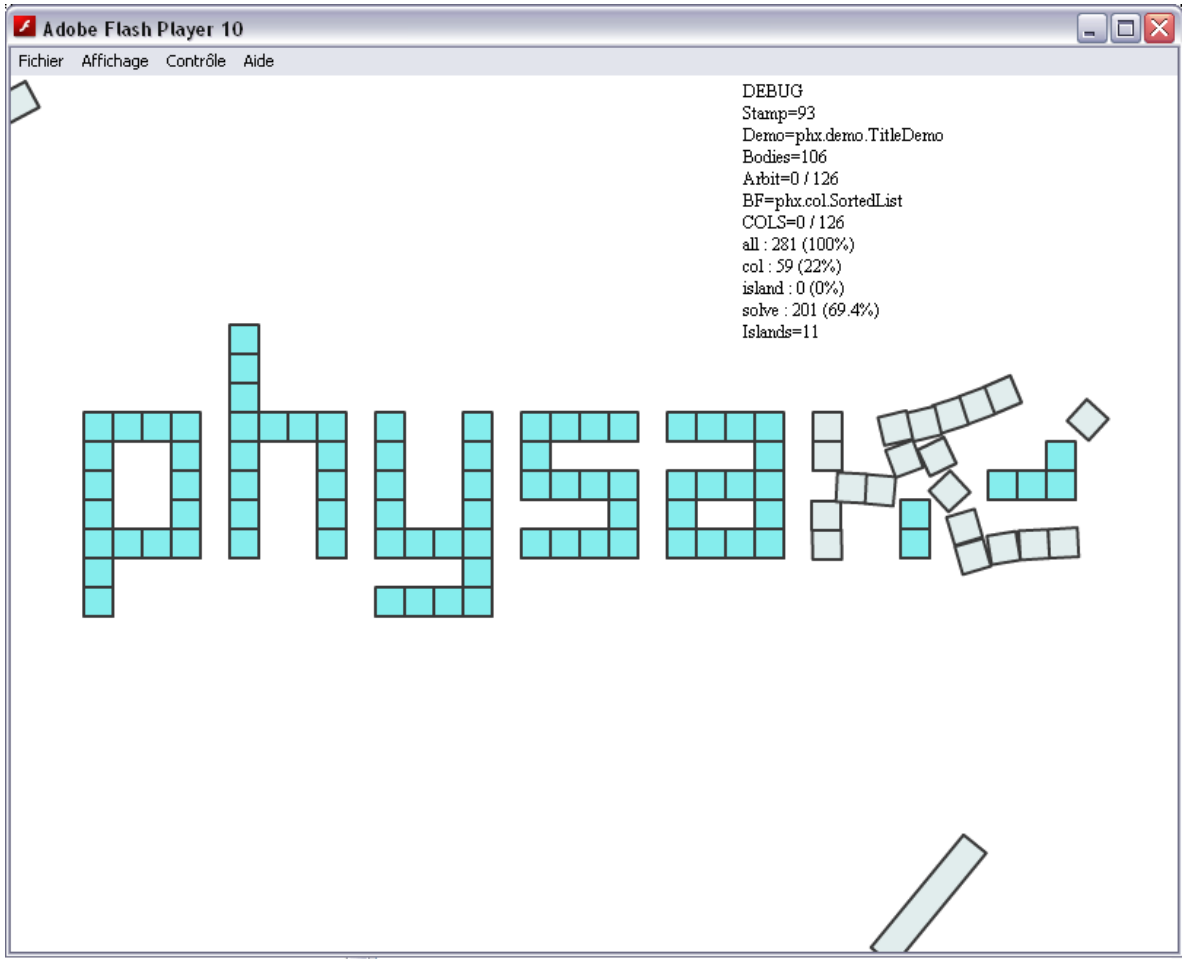
Compiler Speed

- hxFormat Benchmark
  - 64 files , 10.000 lines, 300KB
- haXe : 0.31s
- AS3 conversion
- MXMLC : 3.3s

Inlining

```
1  class Foo {
2
3      inline function addif( x, y, flag ) {
4          return flag ? x + y : x - y;
5      }
6
7      function bar() {
8          var x = addif(1,5,true);
9      }
10
11 }
```

Inline



In Action !

Back to Generics

```
1  class Cell<T> {
2      public var elt : T;
3      public var next : Cell<T>;
4      public function new() {
5      }
6  }
```

Generics

- The issue with Generics
- The solution : haxe.rtti.Generic
- Tradeoff

Poll

What is the most fast way to read/write data ?

A) Array
B) Vector
C) ByteArray
D) SomethingElse

Poll

What is the most fast way to read/write data ?

A) Array
B) Vector
C) ByteArray
D) SomethingElse

Poll

What is the most fast way to read/write data ?

A) Array
B) Vector
C) ByteArray
D) **Alchemy Global Memory**

Alchemy

```
1  package flash;
2
3  class Memory {
4      public static inline function select( b : flash.utils.ByteArray ) : Void;
5      public static inline function getByte( addr : Int ) : Int;
6      public static inline function setByte( addr : Int, v : Int ) : Void;
7      // ....
8  }
```

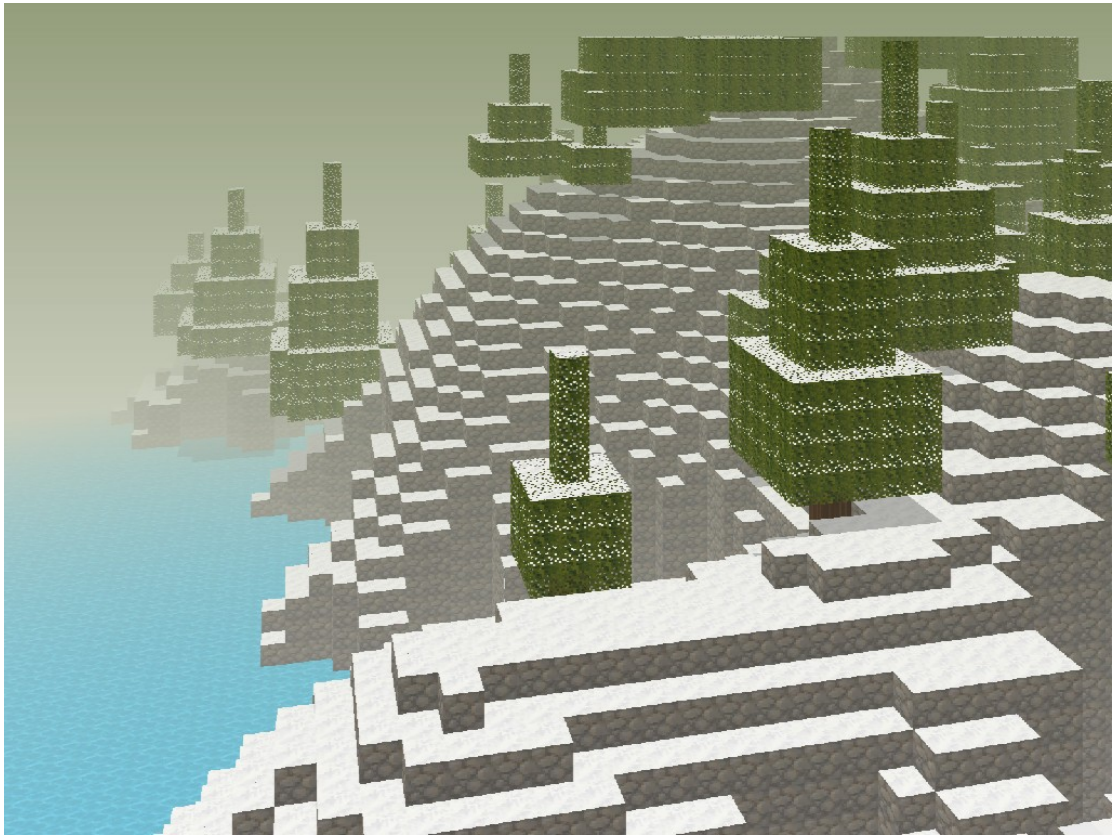
flash.Memory

```
// Quake Fast Inverse Square Root
function invSqrt( x : Float ) : Float {
    var half = 0.5 * x;
    flash.Memory.setFloat(0,x);
    var i = flash.Memory.getI32(0);
    i = 0x5f3759df - (i>>1);
    flash.Memory.setI32(0,i);
    x = flash.Memory.getFloat(0);
    x = x * (1.5 - half*x*x);
    return x;
}
```

flash.Memory

- classic invSqrt : 92.61
- « optimized » invSqrt : 56.71 (+62%)
- flash.Memory invSqrt : 11.72 (+686%)
  - ~8 times faster !
  - ~same time as a DIV !

flash.Memory



In Action !

Summary

In case you were sleeping...

Summary

- haxe != AS3
  - But both compiles to SWF
- Features DRY/KISS/INMS
  - Type inference, Generics, Iterators, Enums, flash.Memory
  - ... and actually more
- Open Source

Summary

Ooops !

More targets

- haXe can target SWF9 and SWF10
- But also SWF 6-8
- But also...

More targets

Javascript :

generates a single .js file

More targets

PHP :

write your websites in haXe !

More targets

NekoVM :

write your websites in haXe (faster) !

commandline and desktop apps

extensible with C/C++

haXe

In Action !

http://mybrute.com

More targets

C++ :

all that you can imagine !

including haXe/iPhone

check gamehaxe.com

More targets

The haXe philosophy

haXe@FGS

E.O.F.

Check http://haxe.org