

X hx

10 years of haxe

Hugh Sanderson

```
package cy;

import numerix.*;
import nme.display.BitmapData;
import nme.geom.*;

class ImageTools
{
    public static function getTensorRect(data:Tensor, rect:Rectangle, expandX:Float = 0)
    {
        var x0 = Std.int(rect.x -rect.width*expandX);
        var y0 = Std.int(rect.y-rect.width*expandY);
        var x1 = Std.int(rect.right+0.99+rect.width*expandX);
        var y1 = Std.int(rect.bottom+0.99+rect.width*expandY);
        if (x0<0) x0 = 0;
        if (y0<0) y0 = 0;
        if (x1>data.width) x1 = data.width;
        if (y1>data.height) y1 = data.height;

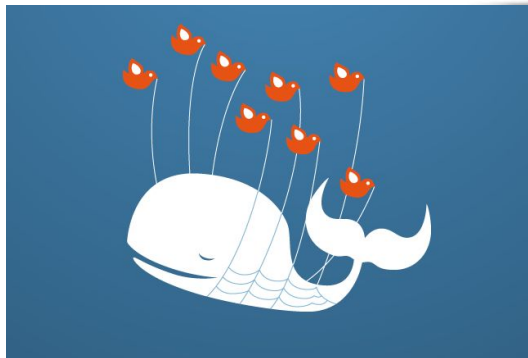
        if (x1==x0 || y1==y0) return null;

        return new Rectangle(x0,y0,x1,y1);
    }
}
```

2006 – a new hope



2006 – a twinkle in the eye



2007 – hacking haxe

Java?

Flash9?

haXe!

Added typing - 2x faster

haXe gets better swf output, so
redundant



2007 – desktop

Nme

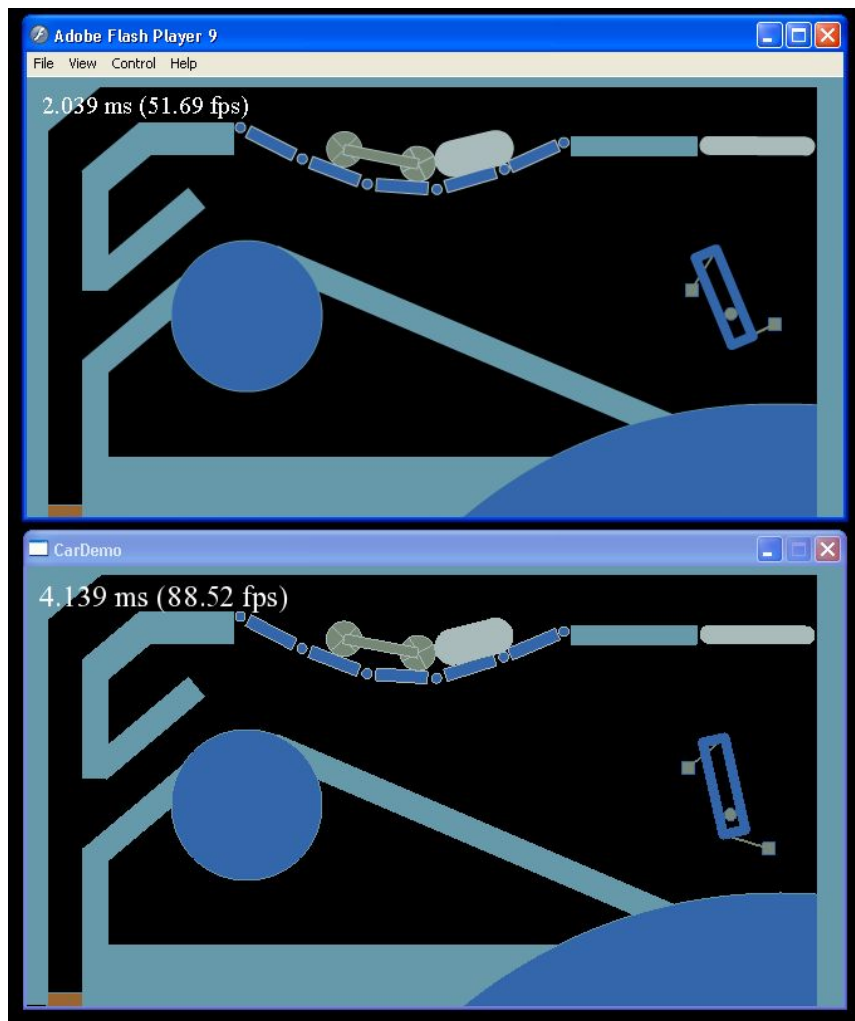
+

Neko

+

Flash API (skeleton)

= Blink

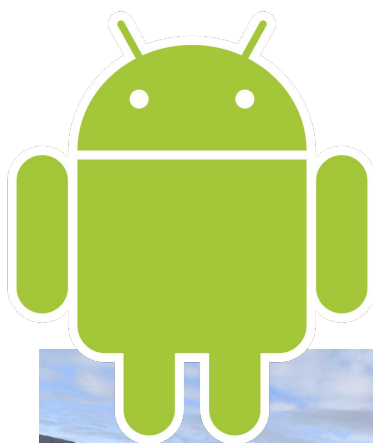


2008 – faster

Hijacked the NME project

Initial hack of hxcpp

Replaced reference counting with
Boehm Gc -> now faster than flash!



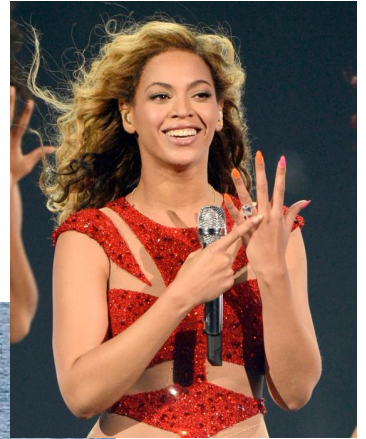
2009 – birth!

Boehm Gc -> Internal Immix version

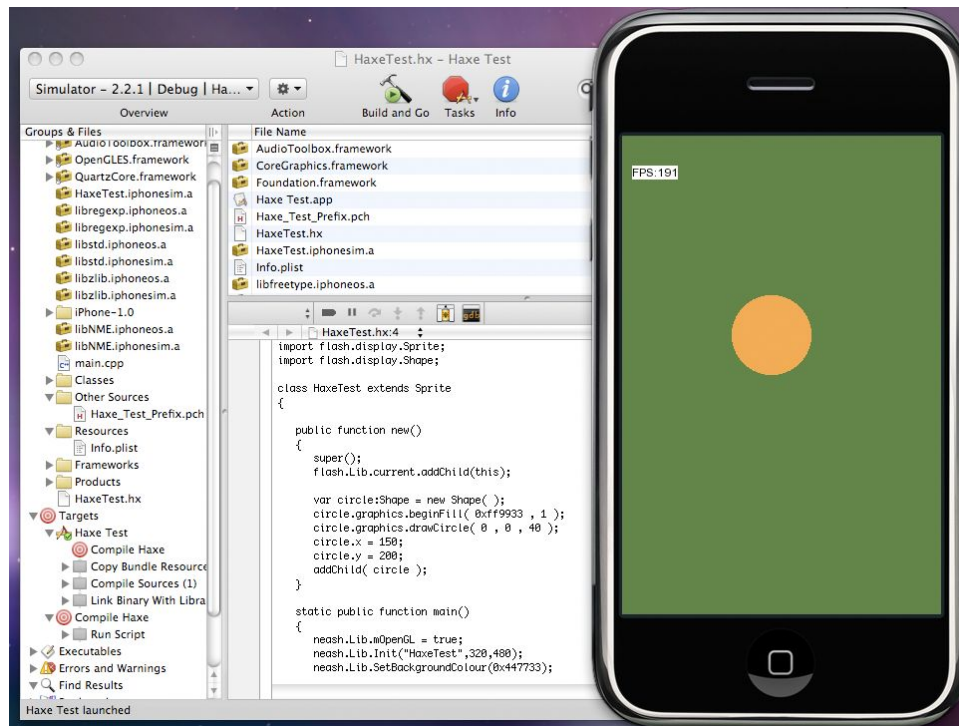
running on iPhone

Hxcpp version 1

Nme version 1



2009 – custom Gc does the trick



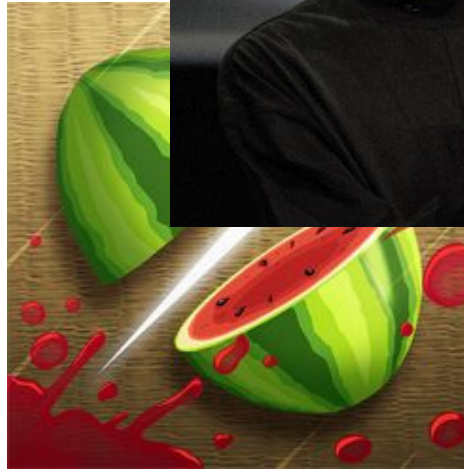
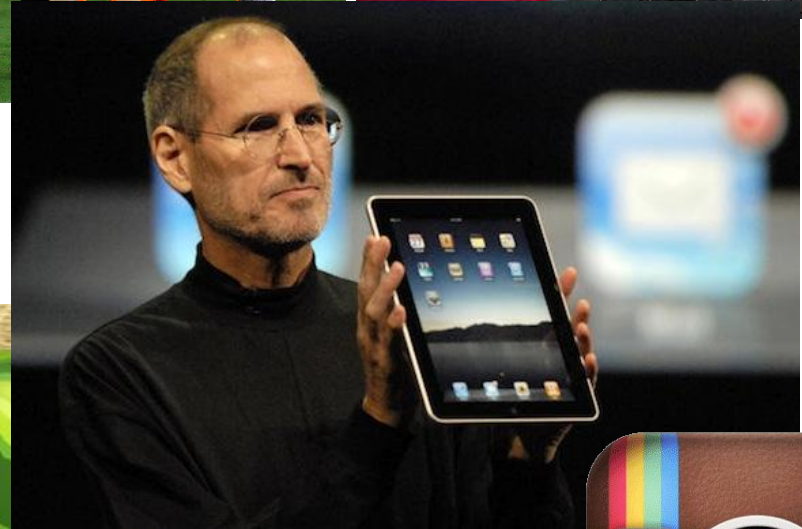
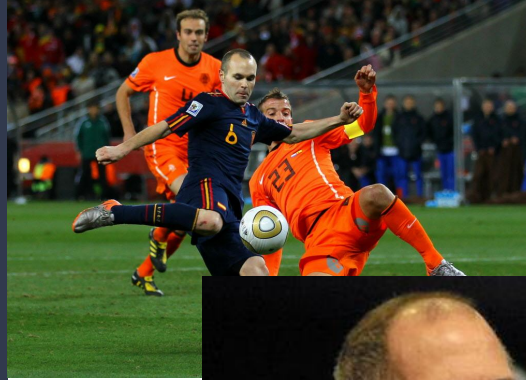
2010 – Nme

Nme 2.0 - now with extra C++

haxe/hxcpp 2.0.6

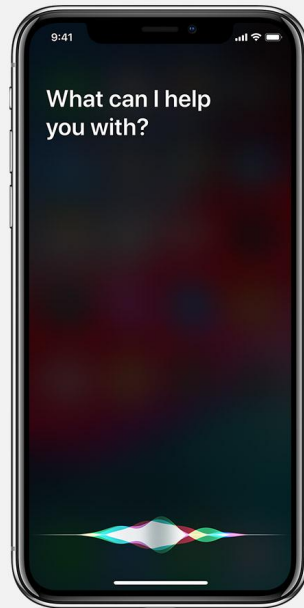
Made Steve Jobs change his mind

Android port - manual project



2011 – easier

Nme build tool following hxcpp's tool



button



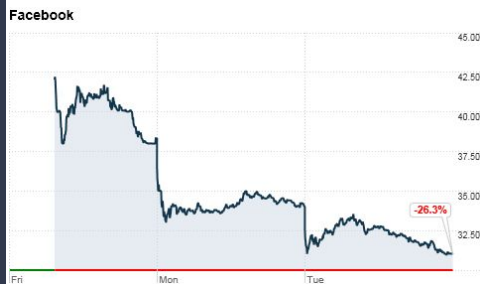
2012 – WWX

Gained some visibility at wwX

Settled on architecture

Listed some issues`

Built-in debugger



2013 – Nme fork

OpenFL forked from Nme

Quite different implementation now

Compatible in principle

Short hiatus from haxe



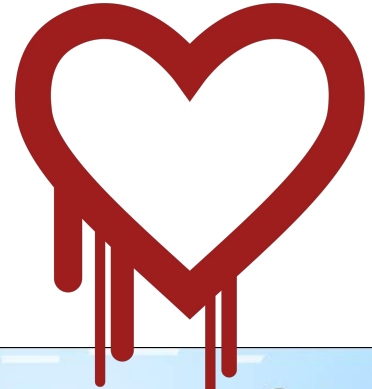
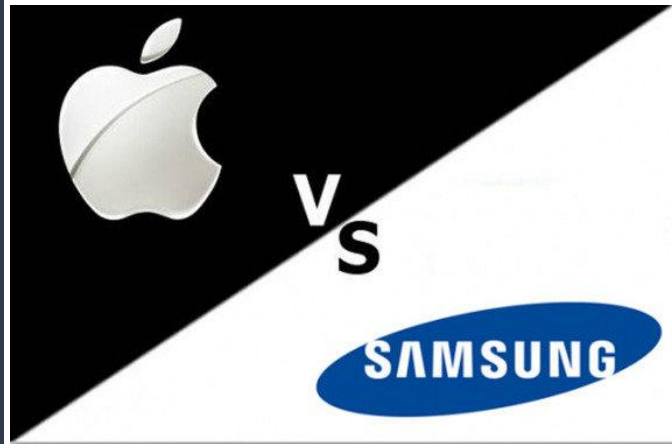
2014 – metadata

wwx2014

Expanded use of metadata

Native integrations

Rationalized Nme and dropped some targets

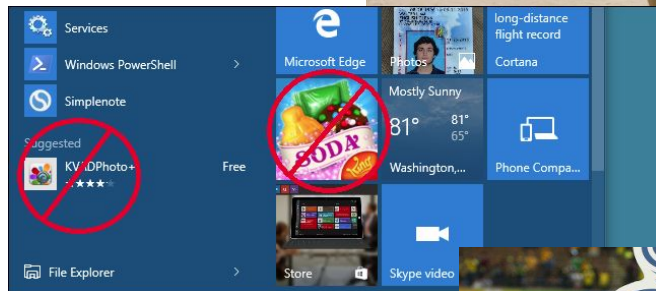
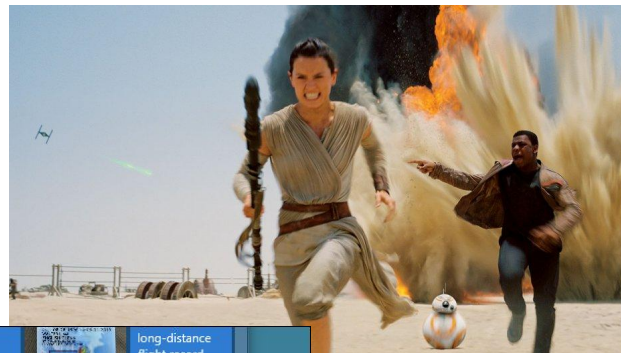


2015 – cppia

Cppia / Acadnme

Inline, parallel Gc

CFFI Prime



2016 – internals

Internal refactor

Better implementation of basic structures

GC Optimizations at many levels

Not shipping binaries

Compiler cache



2017 – web target again

mm / @:objc

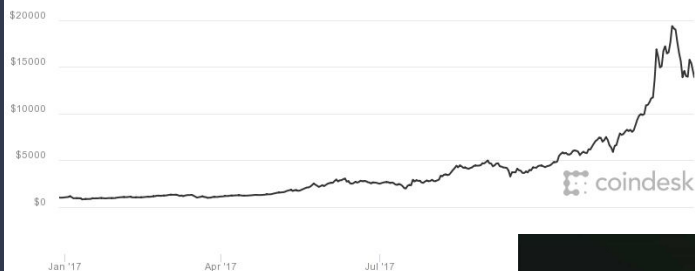
Nme + JsPrime

Move to cpp.Star

GC - cached thread local context

Generational GC

Cppia JIT



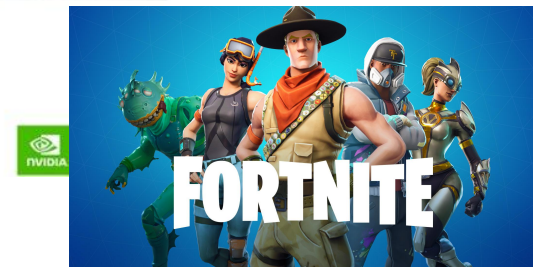
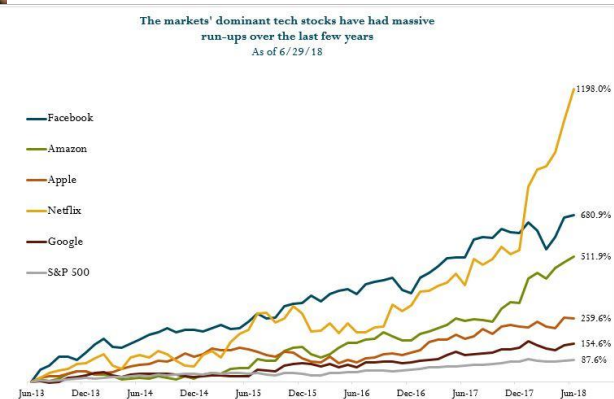
2018 – changing times

Utf16 strings

Career crossroads

Out of gaming, into AI/ML

Finally making desktop apps with Nme



2018 – back to the beginning

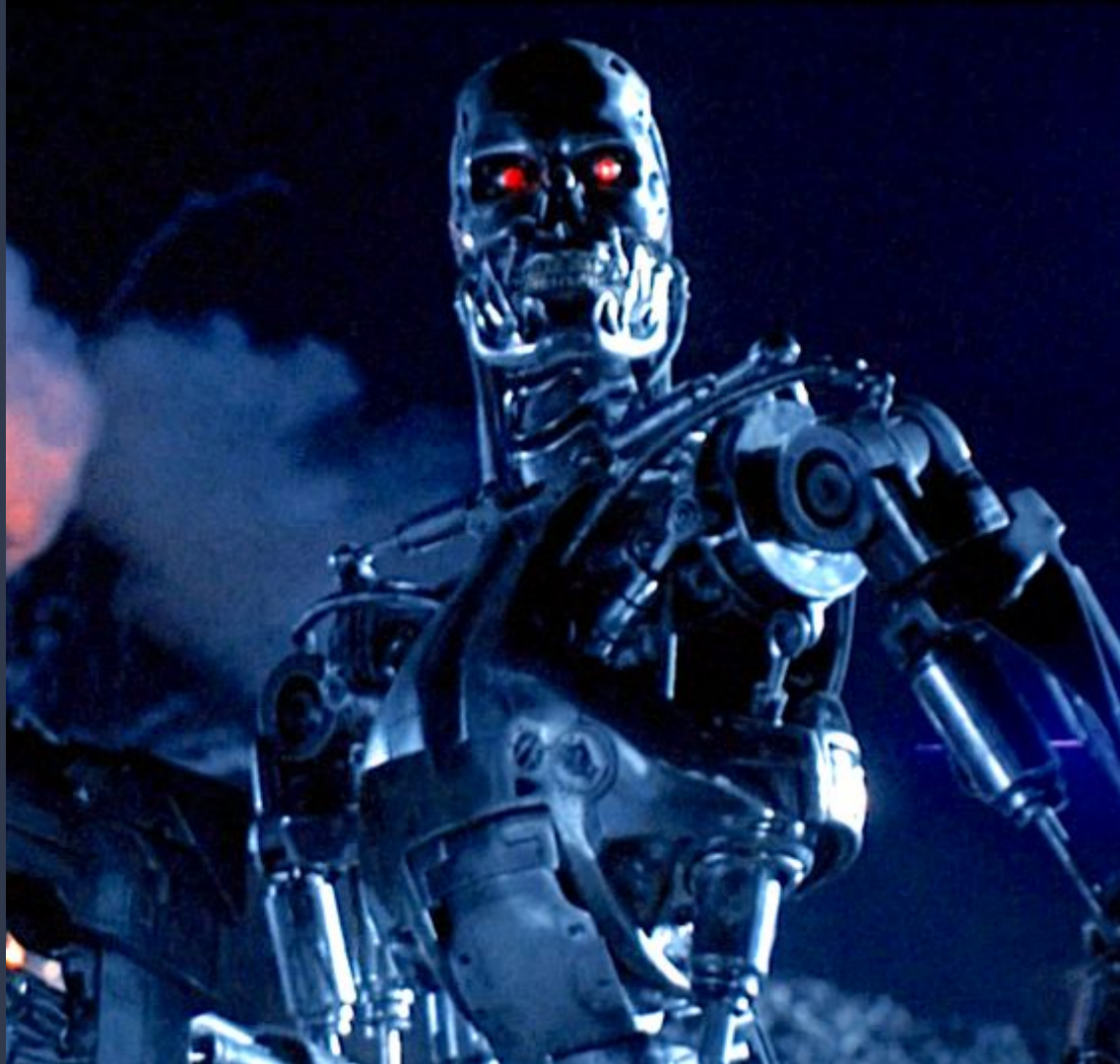


2019 and beyond

The future?

Just watch old presentations:

- Improve Gc
- Int64
- Better native integrations
- More strong typing
 - Functions



Questions?