

haxe NME

vs

The World

Presented by Philippe “FlashDevelop guy” Elsass

Part I:

In the shadow of Flash

A bit of history

2004



AS2

MTASC*

2006

AS3

haxe*

2008

haxe C++

hxCPP/NME**

* created by **Nicolas Cannasse**

** created by **Hugh Sanderson**

Fast forward to 2011

Flash is the 1st indie game platform
but it **failed** to go mobile

HTML5 and **tablets** are the buzz

haxe NME has discreetly matured

In the shadow of Flash

Fast forward to 2011



Part 2:

Bunnies vs The World!



Bunnies vs The World

haxe NME under the spotlights,

thanks to...

haxe NME under the spotlights,

thanks to a professional evangelist,

Joshua Granick:

- haxenme.org
- haxejs.org
- joshuagranick.com

- [@haxelang](https://twitter.com/haxelang)
- [@haxenme](https://twitter.com/haxenme)
- [@haxejs](https://twitter.com/haxejs)

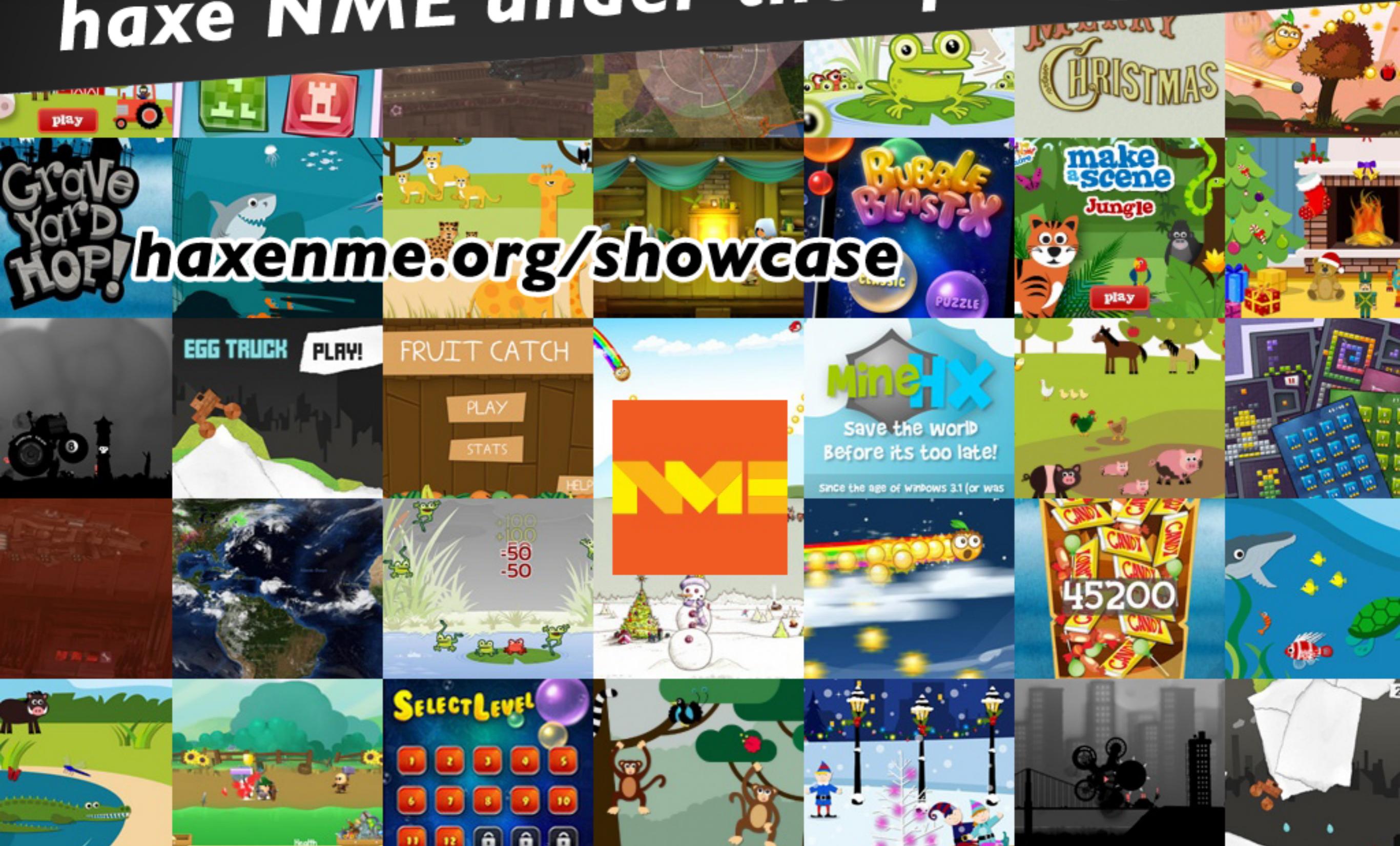
haxe NME under the spotlights,

thanks to a kick-ass
BunnyMark:

- showcases native-level performance,
- with source:
github.com/elsassph/nme-bunnymark

Bunnies vs The World

Bamboo haxe NME under the spotlights,

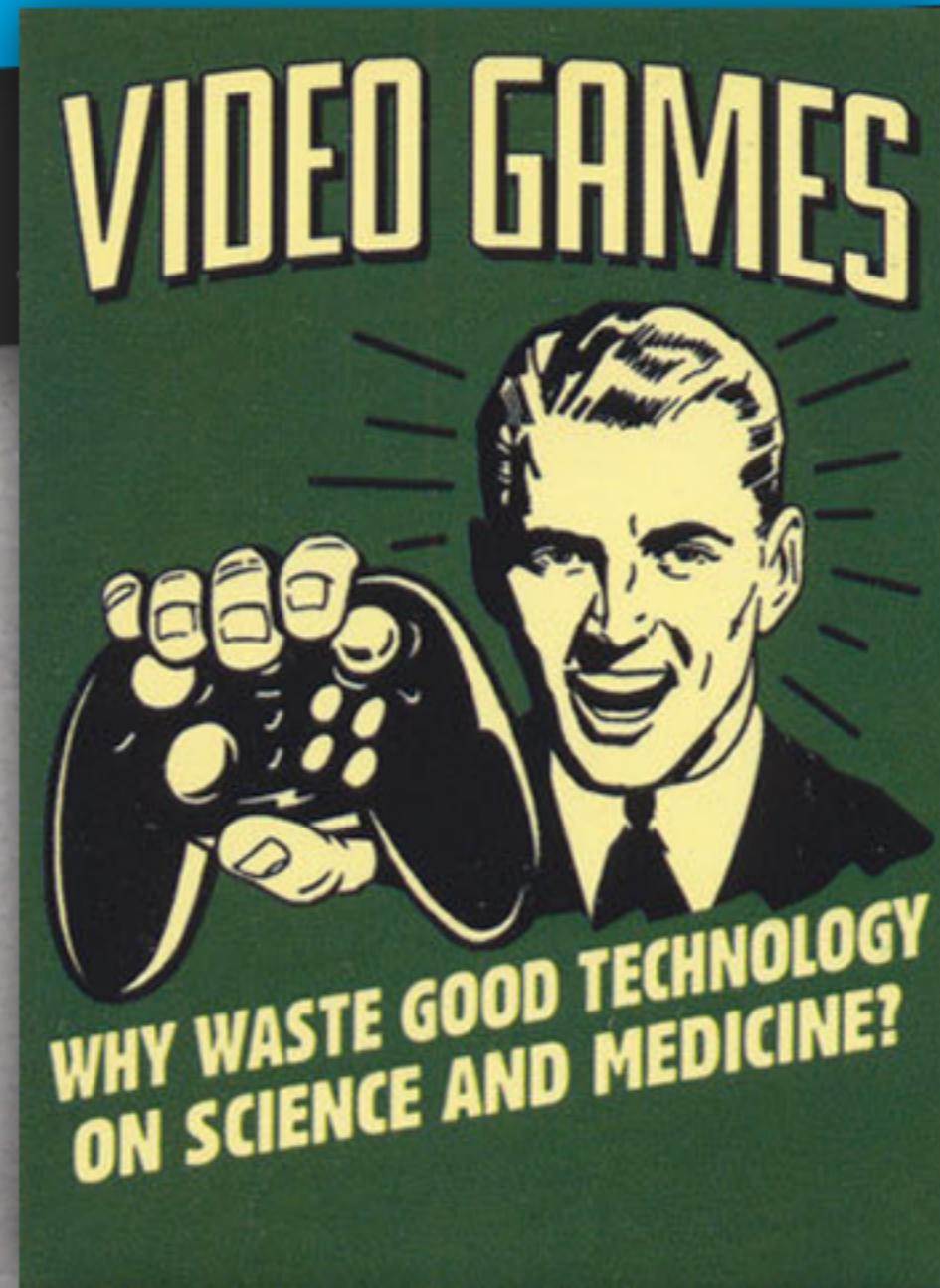


haxe NME vs The World

@elsassph

Part 3:

Options for



“Native” languages

Mono-platform (nowadays?)

Obj-C (Cocos2D...), Java (LibGdx...)

Cross-platform

C++ (Marmalade, Cocos2d-X...)

Native is not productive.

“Interpreted” languages

Cross-platform VM

LUA (Corona, Moai...),
Javascript (Phonegap, Spaceport.io...),
AS (ScaleForm),
etc.

Easy to start, hard to scale.

“Cross-compiled” languages

AOT/cross-compilation

C# (Unity3D, MonoGame...),

AS3 (Adobe AIR, Stencyl...)

haxe (NME).

Sweet spot, if performance follows!

Choose your poison

Is your tech...

- performant on mobile?
- commercial? pay for beta?
- open source?
- extensible ?
- both desktop and mobile friendly?
- alive? (*beware of startups*)

Choose your poison

haxe NME is:

- performant on mobile
- completely free
- open source
- extensible
- both desktop and mobile friendly
- alive!



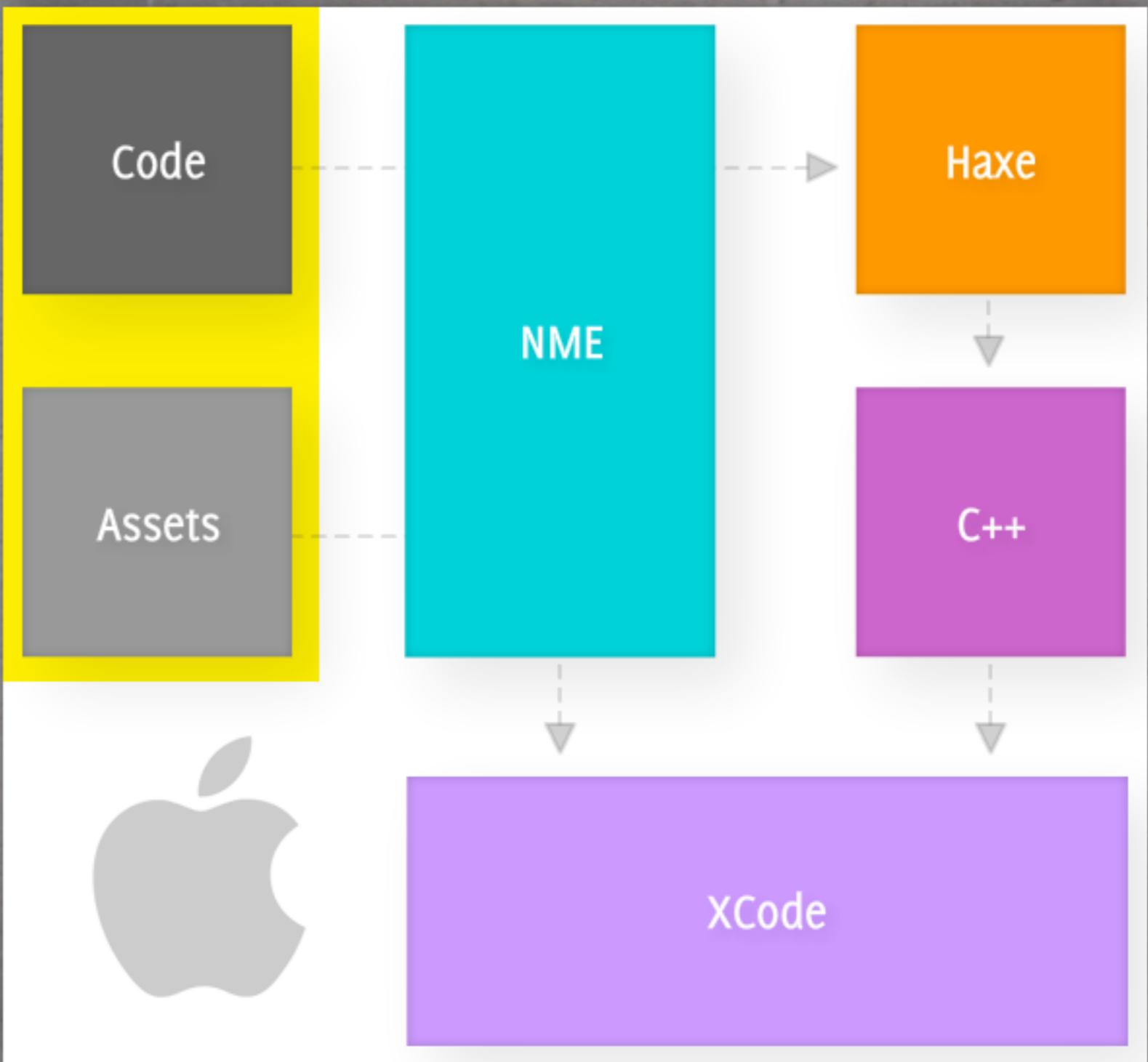
Part 4:

Inside the NME toolbox



“You are there”

The iOS workflow:



NME is... the framework

Cross-platform, **Flash-like API**:

ex: `nme.display.Sprite`

`nme.events.JoystickEvent`

Cross-platform **assets loading**:

ex: `Assets.getBitmap("imgs/logo.png")`

NME is... a command line tool

Configuration

```
$ nme setup android
```

Code generation

output templates, assets embedding,...

Compilation

NME is... entirely hackable

Framework, tools, templates:

0% *black magic*

100% open source

Part 5:

- Holy GPU, Batman!

said Rob-C++



Holy GPU, Batman!

Stick to GPU rendering

Avoid software-bound features
(copyPixels, filters, masks, blendModes,...)

Use images, and batching
(ie. bitmaps, bitmapFills, spritesheets)

Holy GPU, Batman!

Batching? Spritesheets?

Problem

GPU is fast, but CPU => GPU is slow

Solution

Give a lot of work at once to the GPU

Now do you homework

Holy GPU, Batman!

NME's secret ingredient

```
var spritesheet = Assets.getBitmapData("assets/spritesheet.png");
var tilesheet = new Tilesheet(spritesheet);
tilesheet.addTileRect(new Rectangle (0, 0, 32, 32));

var drawList = new Array<Float>();
drawList[0] = x;
drawList[1] = y;
drawList[2] = 0; // rect index
drawList[3] = scale;
drawList[4] = rotation;
drawList[5] = alpha;

graphics.clear();
tilesheet.drawTiles(graphics, drawList, smooth,
    Tilesheet.TILE_SCALE | Tilesheet.TILE_ROTATION | Tilesheet.TILE_ALPHA);
```

Tilesheet

the fastest path to render stuff en masse

Holy GPU, Batman!

NME's secret ingredient

```
var spritesheet = Assets.getBitmapData("assets/spritesheet.png");
var tilesheet = new Tilesheet(spritesheet);
tilesheet.addTileRect(new Rectangle (0, 0, 32, 32));

var drawList = new Array<Float>();
drawList[0] = x;
drawList[1] = y;
drawList[2] = 0; // rect index
drawList[3] = scale;
drawList[4] = rotation;
drawList[5] = alpha;

graphics.clear();
tilesheet.drawTiles(graphics, drawList, smooth,
    Tilesheet.TILE_SCALE | Tilesheet.TILE_ROTATION | Tilesheet.TILE_ALPHA);
```

MOAR
OF THAT!

*Note: it also supports RGB coloring
and additive blendmode.*

Holy GPU, Batman!

Dude, that's really low level!

Like Flash,
NME provides a **low level API**.

Like Flash,
code or **use a library**.
hint: NME inventor also makes ‘gm2d’

Holy GPU, Batman!

I'm sold, anything I should know?

There are limitations...

- OpenGL ES 1.1, no 3D (*think Cocos2D*)
- No video or webviews (*for now*)
- Very limited HTML in TextFields

Holy GPU, Batman!

I'm sold, anything I should know?

...and stuff in-progress

- SWFs rendering
- OpenGL ES 2.0
- Blackberry target
- friendlier native extensions
(ads, analytics, game centers,...)

Native Extensions <3

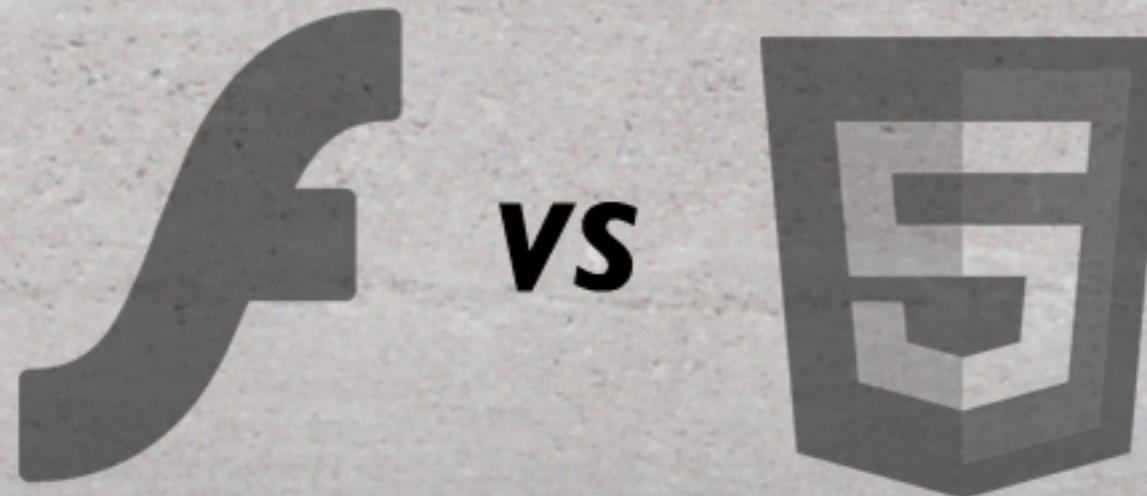
Some good resources

- a sample extension on haxenme.org
- nmex on google code
- nme-extensions on google code

*Not that hard, really!
(you should learn C++ anyway)*

Part 6:

What about the web?



Flash target

Great for interactive **debugging**,
quick iterations, demos...

But...

- no Stage3D support
- limited Tilesheet support

HTML5 target (Jeash)

Exciting and improving quickly

- source-mapping*
- smaller footprint

But...

- DOM+Canvas 2D only
- No Tilesheet, no drawTriangles

WTF is source-mapping?

*Look!
haxe
code in
Google
Chrome
here:

The screenshot shows the Google Chrome DevTools Breakpoints tab for a file named "ClientCtx.hx". The code is written in Haxe. A red arrow points to a breakpoint at line 51, which is part of a function named "call". The right panel displays the Call Stack, Scope Variables, and Breakpoints. The Call Stack shows the execution path from "ClientCtx.call" down to "ClientMain.main". The Scope Variables panel shows local variables like "args", "method", "onSuccess", "promise", and "this". The Breakpoints panel has a checked entry for the current line 51.

```
31
32     public static var msgs :MessageBundle;
33
34     // A cache of playerIds to profile photo images
35     public static var profilePics :Promise<AssetPack>;
36
37     public static var level :Level;
38
39     public static var collectables :IntHash<Bool>;
40
41     // Credentials used to login and sign all our remoting calls
42     public static var creds :Creds;
43
44     // Canonical server time
45     public static var time :ServerTime;
46
47     // Invoke a signed call to gameSvc by prepending our credentials
48     public static function call<A> (
49         method :String, args :Array<Dynamic>, ?onSuccess :A
50     {
51         var promise = new Promise<A>();
52         gameSvc.resolve(method).call([creds].concat(cast args));
53         if (onSuccess != null) {
54             onSuccess(result);
55         }
56         promise.result = result;
57     });
58     return promise;
59
60
61     public static function callNoAuth<A> (
62         method :String, args :Array<Dynamic>, ?onSuccess :A
63     {
64         var promise = new Promise<A>();
65         gameSvc.resolve(method).call(args, function (result
```

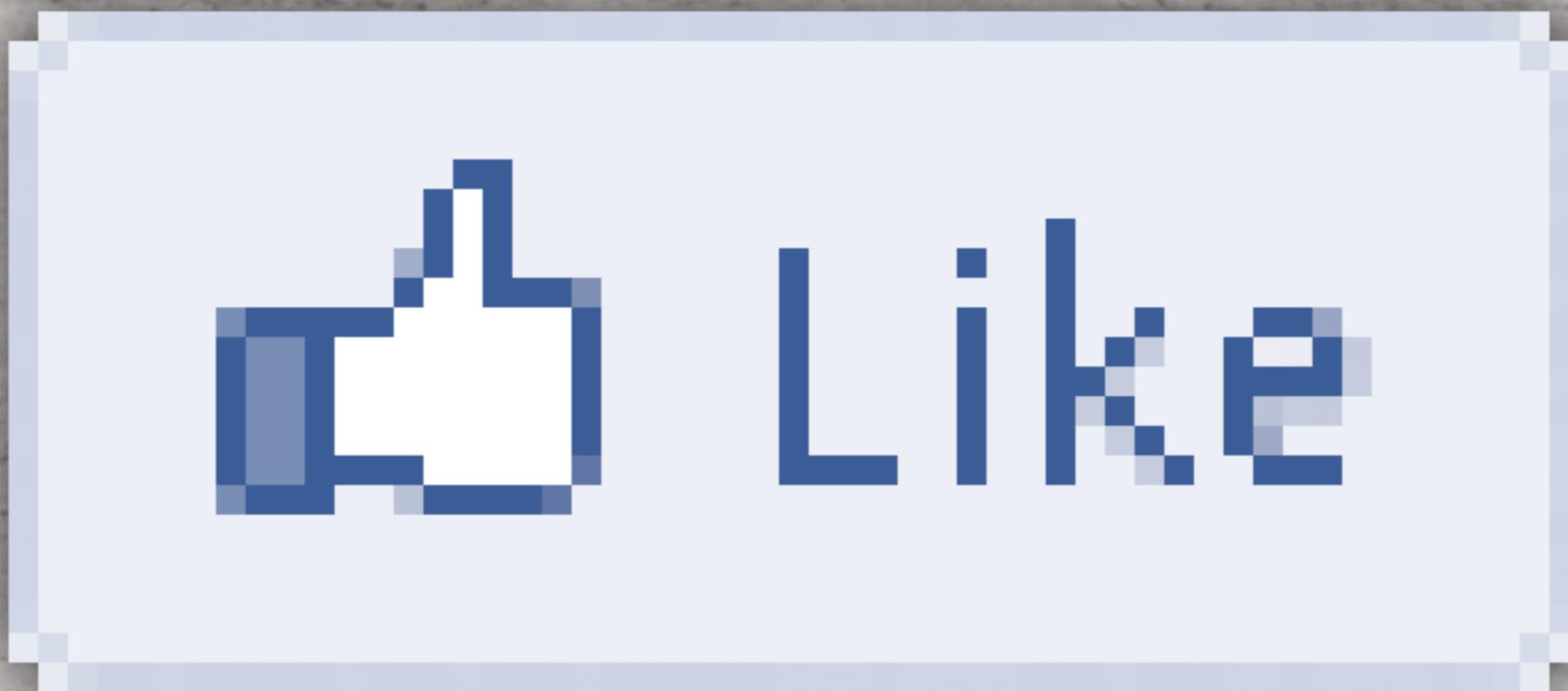
Future of the web targets (imho)

NME needs a web-friendly
batching API

HTML5/Jeash has a bright future,
contribute!

Is **Flash** worth our efforts?

Make it yours!



haxe NME now!