

# Haxe: What Makes It Cool

## Dev Ideas, episode 1

The Haxe logo is displayed in a stylized, orange, blocky font. The letters are bold and have a slight shadow effect. The logo is centered within a yellow rectangular box that has a subtle gradient and a slight drop shadow.

**HAXE**

Brought to you by [Chicken Wing Software](#)

# Hello, my name is...

---

## Eddie Sullivan

Founder, *Chicken Wing Software*

[www.chickenwingsw.com](http://www.chickenwingsw.com)

[www.deideas.com](http://www.deideas.com)

See also: [UX Ideas](#)

# HAXE

- ▶ Haxe  $\approx$ 
  - JavaScript + static types + Java-style classes + type inference + algebraic types
- ▶ [www.haxe.org](http://www.haxe.org)
- ▶ Compiles to JavaScript, Flash, NekoVM, PHP, C++
- ▶ Standard library + platform libs + RPC

# Haxe is like JavaScript

## (really ActionScript)

- ▶ Similar syntax and keywords (like `function`)
- ▶ Entire DOM-tree available
- ▶ Flash API available (no timeline)
- ▶ Can compile to `.js` file
- ▶ Can compile to `.swf` file
- ▶ Closures
- ▶ Regular expressions

# DOM example

Example:

```
// Change an HTML element to a random color.
static function changeToRandomColor(id)
{
    var el = Lib.document.getElementById(id);
    var color = '#' + StringTools.hex(Std.random(0x1000000), 6);
    el.style.backgroundColor = color;
}
```

# Haxe is like... *Java*

## Example:

```
import js.Lib;
import StringTools;

class HaxeExample {
    // Required "main" function - called at startup.
    static function main() { }

    // Change an HTML element to a random color.
    static function changeToRandomColor(id)
    {
        var el = Lib.document.getElementById(id);
        var color = '#' + StringTools.hex(Std.random(0x1000000), 6);
        el.style.backgroundColor = color;
    }
}
```

# Similarities to Java

- ▶ Java-style classes & inheritance
- ▶ Statically & strongly typed
- ▶ Single inheritance & interfaces
- ▶ Can compile to NekoVM, PHP, or C++

# Some familiar keywords

if/else	do/while
true/false	new
var	this
try/catch	finally
return	switch*
enum*	for*

\* Different behavior

# Haxe is like... *ML*



- ▶ Type inference
  - Types with less typing

```
var x = "hello"; // x is a String
var y:String;    // y is also a String

x = 3;           // ERROR!
```

- ▶ Algebraic types (With `enum` and `switch`)

# Haxe's enums

Defining the enum:

```
enum Command {  
    sit;  
    speak(word:String);  
    move(x:Int, y:Int);  
}  
  
// ... inside a function  
var cmd = Command.speak("Arf!");
```

# Haxe's switch

Using the enum:

```
// ... inside a function:  
switch (cmd) {  
  case sit:  
    this.sitting = true;  
  
  case speak(text):  
    trace(text);  
  
  case move(x, y):  
    this.location = new Point(x, y);  
}
```

# Haxe's for

More like "foreach" in C#

```
// ... inside a function:  
  
var names = ['Ed', 'Fred', 'Ned', 'Ted'];  
// Could have written:  
// var names:Array<String> = ...  
  
for(name in names) {  
    trace("Hello " + name);  
}
```

# Other features

- ▶ Local functions/closures
- ▶ Type parameters (generics)
- ▶ Exceptions
- ▶ **Dynamic** & **untyped** values
- ▶ "Batteries Included"
  - Remoting
  - XML parsing/validation
  - Sandboxing
  - ...and much more!

# Flash example (part 1)

```
import flash.display.Sprite;
import caurina.transitions.Tweener;

class Happy extends Sprite
{
    public function new(size:Float = 50)
    {
        super();
        graphics.lineStyle(1.0, 0x000000);
        graphics.beginFill(0xffff00);
        graphics.drawCircle(0, 0, size);
        graphics.endFill();
        graphics.moveTo(size * 0.3, size * 0.3);
        graphics.curveTo(0, size * 0.75, -size * 0.3, size * 0.3);
        graphics.drawCircle(-size * 0.225, -size * 0.3, size * 0.1);
        graphics.drawCircle(size * 0.225, -size * 0.3, size * 0.1);
    }
    // class continues on next slide...
```

# Flash example (part 2)

```
// ... continued from last slide
public function start()
{
    var that = this;
    Tweener.addTween(this, { rotation:360, time:2,
                             transition:"linear",
                             onComplete: start } );
    Tweener.addTween(this, { scaleX:.75, scaleY:.75, time:1 });
    Tweener.addTween(this, { scaleX:1, scaleY:1, time:1,
                             delay:1 } );
}

public function stop()
{
    Tweener.removeTweens(this);
}
} // End of class Happy
```

# Flash example (part 3)

```
import flash.external.ExternalInterface;
import flash.Lib;

class Main
{
    static var happy = new Happy(25.0);

    static function main()
    {
        happy.x = 50;
        happy.y = 50;
        Lib.current.addChild(happy);
        ExternalInterface.addCallback("startHappy", happy.start);
        ExternalInterface.addCallback("stopHappy", happy.stop);
        happy.start();
    }
}
```

# Flash example (part 4)

What it looks like:



Stop

-

Start

# The Haxe Community

---

- ▶ [Mailing list](#)
- ▶ [Forum](#)
- ▶ [Wiki](#)
- ▶ [Libraries](#) / [haxelib](#) ([lib.haxe.org](http://lib.haxe.org))
- ▶ [FlashDevelop](#)
- ▶ [More...](#)



# Thank you!

- ▶ Chicken Wing Software - [chickenwingsw.com](http://chickenwingsw.com)
- ▶ Dev Ideas - [devides.com](http://devides.com)
- ▶ Follow me on Twitter: [@eddieSullivan](https://twitter.com/eddieSullivan)
- ▶ [Sign up for the newsletter](#) to hear what's coming next
- ▶ UX Ideas - [uxideas.com](http://uxideas.com)