

# Actionscript 3 Graphical Programming

Carly Gooch



# Introduction

## About Me

- Who am I — Carly Gooch
- My experience — Senior Developer at Workstar
- All the examples - <http://koali.com.au>



# What are we going to cover

- Display Objects – Getting things onto the screen
- Colour - Understanding and Generating colour
- Drawing API – lines, fills and more
- Bitmap API – pixels, filters and fun
- Using external assets – compiled into your movie and loaded at runtime
- Getting your illustrations into flex applications
- Pixel Bending – Flash 10

# How we are going to cover it

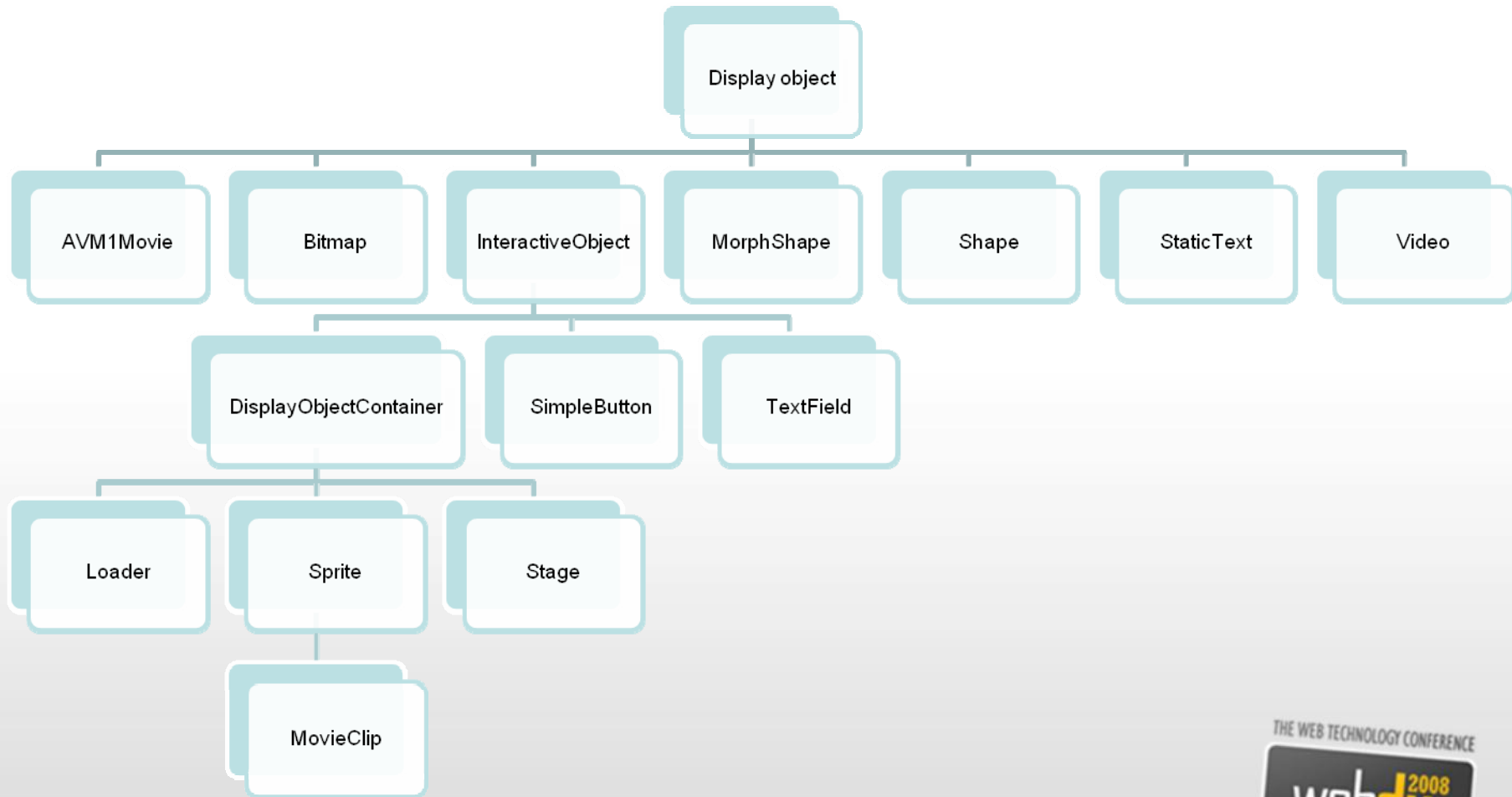
- Through 9 projects
  - 1. The basics
  - 2. Drawing and Colours
  - 3. Interactive drawing
  - 4. When lines go colour
  - 5. OoO bitmaps
  - 6. loading in assets
  - 7. Embedding assets
  - 8. Getting your work into flex
  - 9. The new stuff

# Before we start

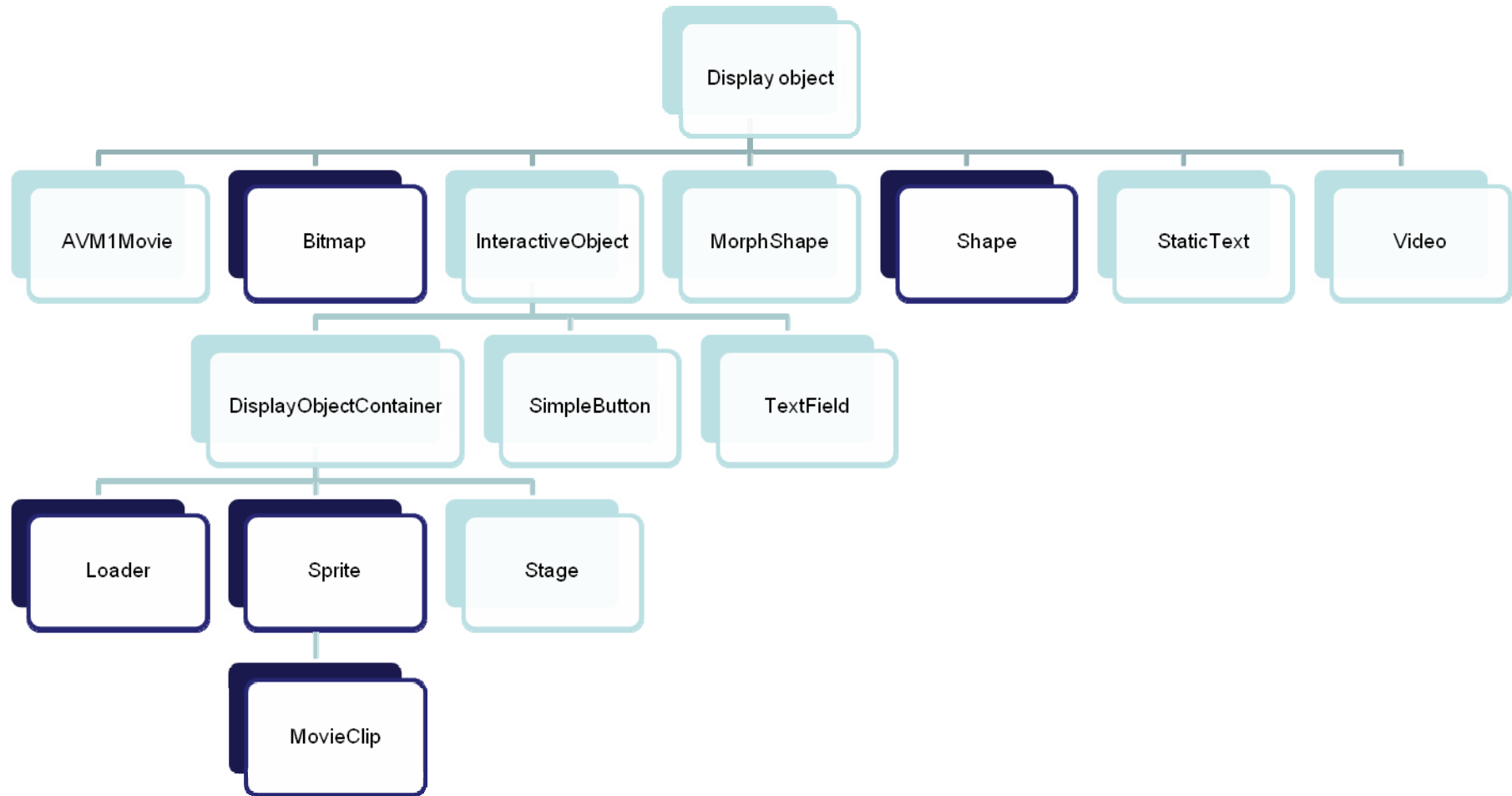
- Understanding Display Objects
- Understanding Colours



# Display Objects



# Display Objects



# Display objects

## No interaction or display children

- Bitmap — to put bitmaps on the screen
- Shape — To put vectors on the screen

## Mouse interaction and display children

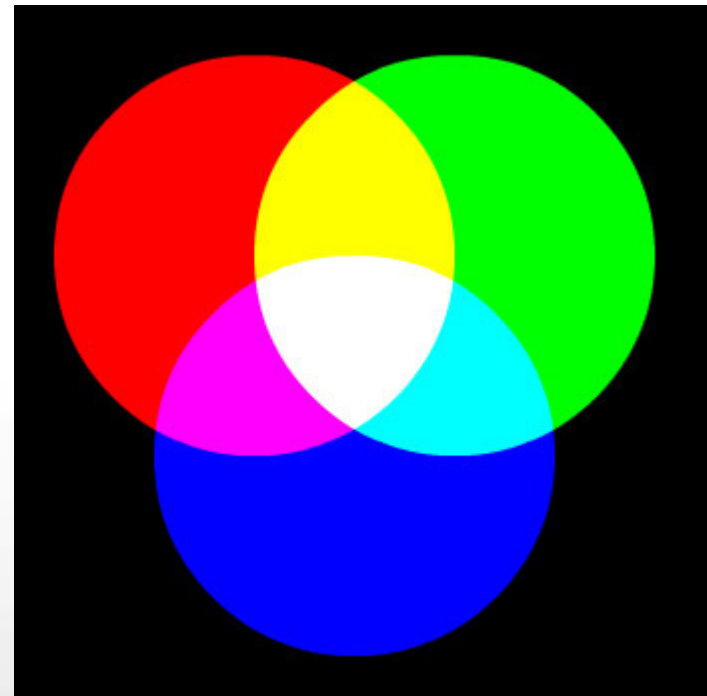
- Sprite — listens for mouse events
- Movieclip — same as sprite but also has enterframe event (for animation)
- Loader — Loads external assets at runtime

# Additive Colour – on Screen

Colour is made up of 3 parts

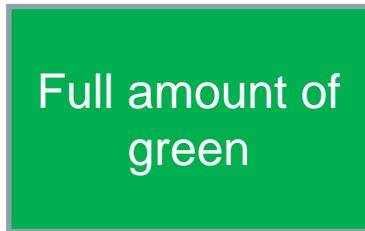
- Red
- Green
- Blue

[\[demonstration\]](#)



# Hexidecimal representation of colour

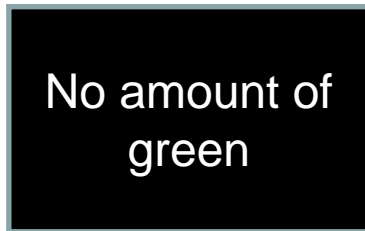
# FFF FFF



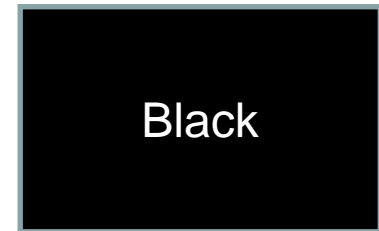
=



# 000 000



=



16 possible values:

0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

# Hexidecimal colours in flash

- In flash a hexidecimal number is represented by 0x
- Blue would be  
`var blue:Number = 0x0000FF;`
- Some functions in flash take 32bit colour values, this means the alpha is included in the number.



0xAARRGGBB — think of a pirate to remember the order

# Project I – The Basics

-Drawing lines



# Simple drawing of lines

- Example draw line from one point to another
- Drawing is done on the graphics object of the display object
  - Set the line style
    - `this.graphics.lineStyle(thickness, colour, alpha);`
  - Set the starting point
    - `this.graphics.moveTo(100, 100);`
  - Draw the line
    - `this.graphics.lineTo(300, 300);`
- [\[See the code\]](#)

# Graphing

- Graph [example](#) & [source](#)



# Project 2 – Drawing and colours

- More on colours
- Dynamically generating colours
- Drawing Gradients
- Drawing shapes and fills



# More on colour

- The hexadecimal value actually represents a number between 0 and 255.
- 0 is empty
- 255 is full

# Generating Colour - bitwise operators

- $\text{Red}(0-255) \ll 16 \mid \text{Green}(0-255) \ll 8 \mid \text{Blue}(0-255)$

Or

- $\text{Alpha}(0-255) \ll 24 \mid \text{Red}(0-255) \ll 16 \mid \text{Green}(0-255) \ll 8 \mid \text{Blue}(0-255)$

Example

```
var pureCyan:uint = 0 << 24 | 255 << 16 | 255
```

More bitwise at:

<http://lab.polygonaal.de/2007/05/10/bitwise-gems-fast-integer-math/>

# Generating Colour - bitwise operators

You can also do:

```
var red:uint = 0x00;
```

```
var blue:uint = 0xFF;
```

```
var green:uint = 0xFF;
```

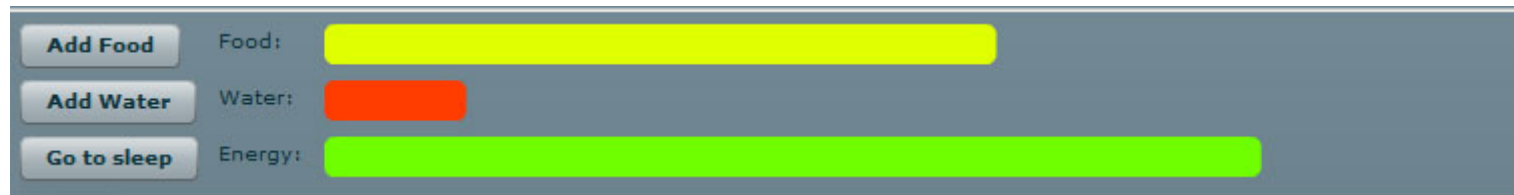
```
var pureCyan:uint = red << 16 | green << 8 | blue;
```

More bitwise at:

<http://lab.polygonaal.de/2007/05/10/bitwise-gems-fast-integer-math/>

# Drawing fills

- [Health/energy bar](#) example



- To generate fills:

```
this.graphics.beginFill(colour, 1);
```

- To draw shapes:

```
this.graphics.drawRoundRect(0, 0, _width*val,  
_height,10);
```

# Drawing Gradients

- [Health/energy bar](#) example with gradients



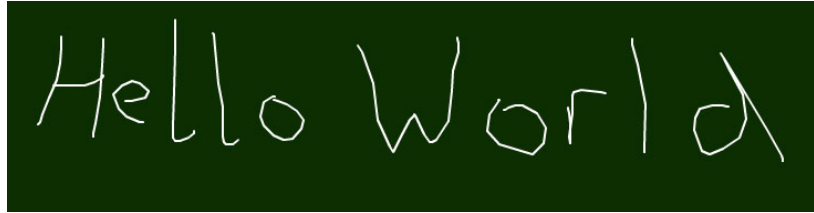
```
this.graphics.beginGradientFill(fillType, colors, alphas,  
    ratios, matr);  
this.graphics.drawRoundRect(0, 0, _width*val, _height,10);
```

# Project 3 – Interactive drawing

- Using mouse listeners
- The EnterFrame event



# Blackboard



Hello World

- [Blackboard example](#)
- **Mouse listeners**
  - `this.addEventListener(MouseEvent.CLICK, onBoardPress);`
- **EnterFrame listener (to call code every frame)**
  - `this.addEventListener(Event.ENTER_FRAME, onDraw);`
- **Drawing lines to your mouse**
  - `board.graphics.lineTo(board.mouseX, board.mouseY);`

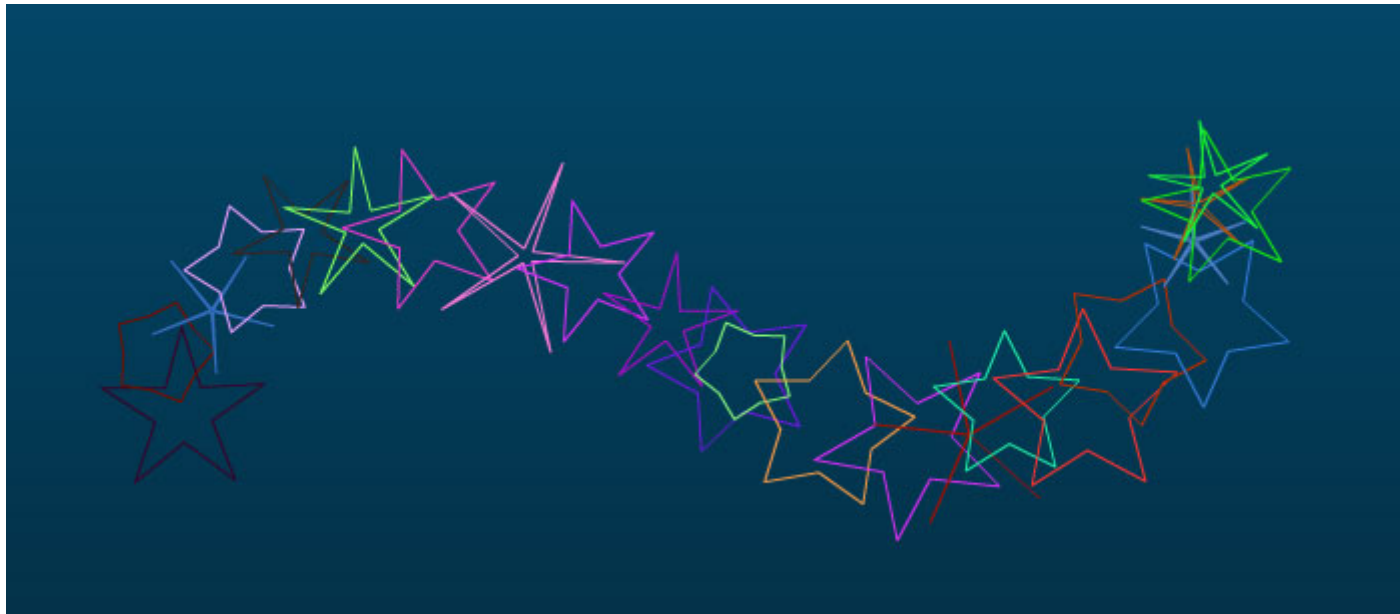
# Project 4 – When lines go crazy

- Random Colours
- Drawing Shapes
- Watch your CPU
- Turning Vector into a bitmap



# You are a star

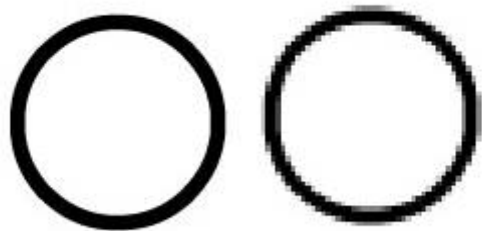
- Random colour:
  - `0xFFFFFFFF * Math.random()`
- Drawing lines that need to render every frame makes the cpu go wild



# Vector vs Bitmap

- Vector illustrations are generated by points, lines and fills
- Bitmap illustrations are generated by pixels

[\[demonstration\]](#)



# Stars bitmapped



- Each frame, draw the graphic onto a bitmap.

```
starBitmapData = new  
    BitmapData(WIDTH, HEIGHT, true, 0x00000000);  
starBitmap = new Bitmap(starBitmapData);  
starBitmapData.draw(stars);  
this.addChild(starBitmap);
```

# Project 5 – 0o0 bitmaps

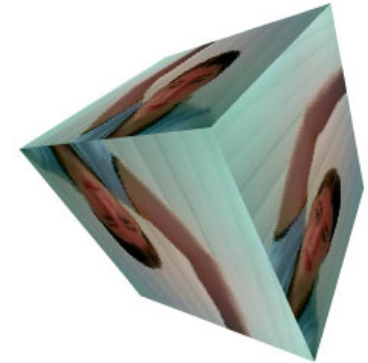
- using your bitmaps
- adding filters to bitmaps



# Same concept, different results

- You can draw any display object onto a bitmap
- [3D webcam example](#)
- Webcamera feed -> bitmap -> papervision material

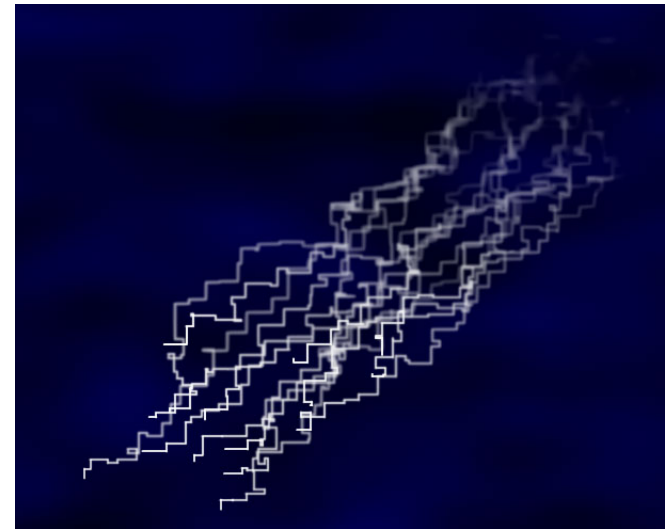
```
videoBitmapData = new BitmapData(320, 240,  
    false, 0xFF000000);  
videoBitmap = new Bitmap(videoBitmapData);  
videoBitmapData.draw(video);
```



# Add a filter

- [Squiggly lines example](#)

```
onScreenBitmapData.applyFilter(on  
    ScreenBitmapData, new  
    Rectangle(0, 0, _width,  
    _height), new Point(0,0), new  
    BlurFilter(1.1, 1.1, 1));
```



# Project 6 – Loading in assets

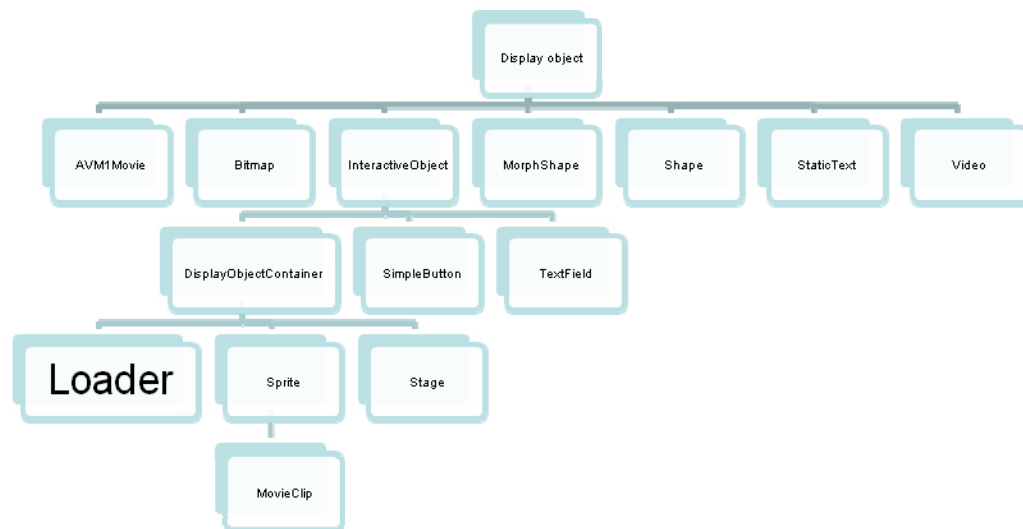
- How to load in assets at runtime
- Working with the loaded assets
- Applying filters to display objects



# Illusions of movement -Parallax

- [Parallax example](#)
- Loading files at runtime

```
imageLoader = new Loader();  
imageLoader.load(new URLRequest(image));
```



# Illusions of movement -Parallax

- [Parallax example](#)
- To add a filter to a sprite:

```
var array_filter:Array=new Array();  
var filter:BlurFilter=new BlurFilter(layer.speed *  
    percent,0, 1);  
array_filter.push(filter);  
layer.filters=array_filter;
```

- If a graphic is bigger then 2880px the filter will not apply



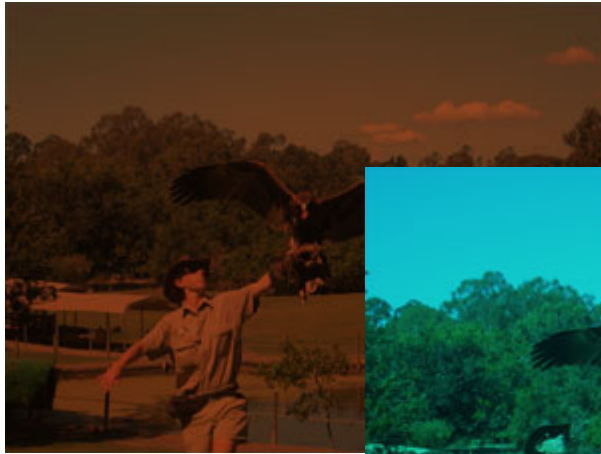
# Project 7 – Embedding assets

- Embedding assets into your project
- Using colour matrixes to manipulate graphics



# Embedding assets

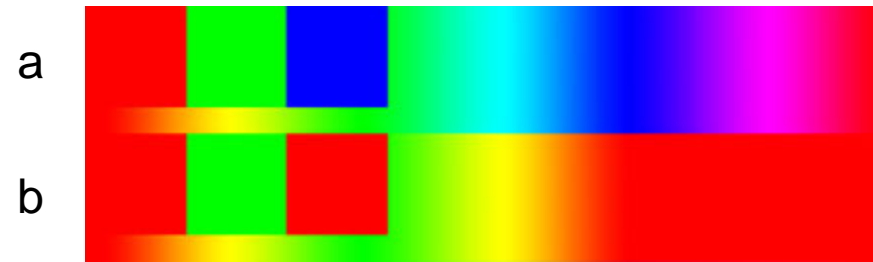
- [Colour wash example](#)



```
[Embed(source = "img/eagle.jpg")]  
private var EagleImage:Class;
```

```
var image:Bitmap = new EagleImage();  
this.addChild(image);
```

# Colour Matrixes



a) All the red, green and blue input are put in the output

	Input Red	Input Green	Input Blue
Output Red	1	0	0
Output Green	0	1	0
Output Blue	0	0	1

b) The blue input is put in the red output

	Input Red	Input Green	Input Blue
Output Red	1	0	1
Output Green	0	1	0
Output Blue	0	0	0

# Colour Matrixes

```
var matrix:Array = new Array();  
matrix = matrix.concat([1, 0, 1, 0, 0]); // red  
matrix = matrix.concat([0, 1, 0, 0, 0]); // green  
matrix = matrix.concat([0, 0, 0, 0, 0]); // blue  
matrix = matrix.concat([0, 0, 0, 1, 0]); // alpha
```

```
var filter:ColorMatrixFilter = new  
    ColorMatrixFilter(matrix);  
var filters:Array = new Array();  
filters.push(filter);  
image.filters = filters
```

# Project 8 – Getting your work into flex

- Putting display objects into uicomponents
- getting colour values from a picture

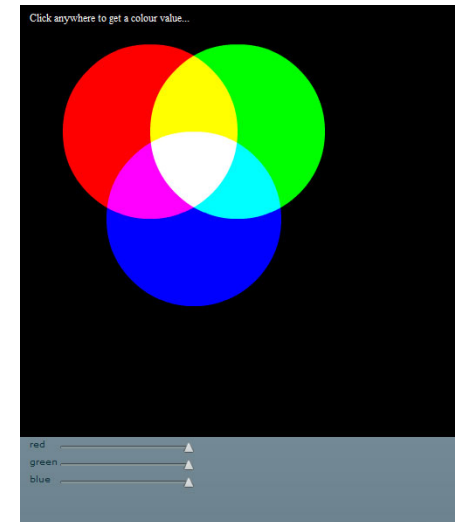


# Example

- [Colour example](#) from the start
- To put a displayobject into a flex project you must wrap it in an **UIComponent**.

```
circles = new Circles(500,500);  
var circlesComponent:UIComponent = new UIComponent();  
circlesComponent.addChild(circles);  
circlesComponent.x = 0;  
circlesComponent.y = 0;  
this.addChild(circlesComponent);
```

- [Censor project](#)



# Getting from colour value back to hexadecimal value

- Sometimes you need to turn your colour value back into a hexadecimal string. An example is when you need to use your colour in a css class.

```
var outputString:String = colour.toString(16);
```

Unfortunately this leaves out the beginning 0s

Solution...

# Getting from colour value back to hexadecimal value

- Sometimes you need to turn your colour value back into a hexadecimal string. An example is when you need to use your colour in a css class.

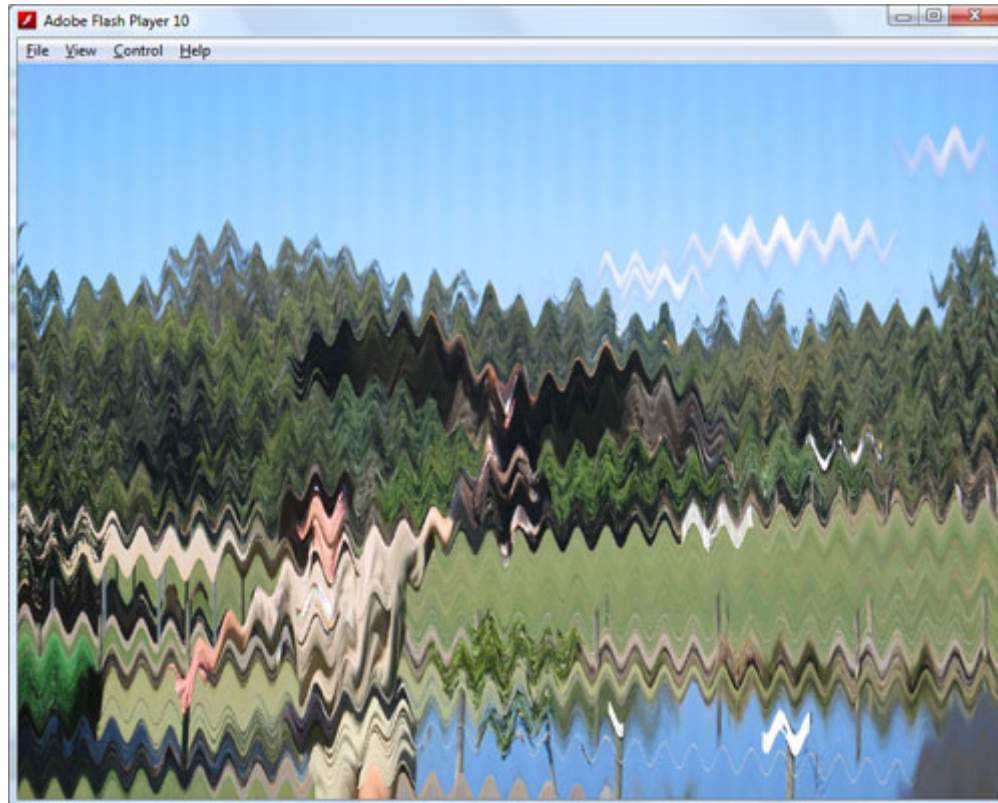
```
var outputString:String = colour.toString(16);
var leadingZeros:String = "";
for (var i:Number = 1; i <= 6 - outputString.length;
    i++) {
    leadingZeros += "0";
}
outputString = leadingZeros+outputString;
```

# Project 9 – the new stuff

- Pixel Bender
- Flash player 10



# New toys



- PixelBender lets you manipulate graphics pixel by pixel
- [Flash 10 examples](#)

# Project 10 - Questions



# More info

- Grant Skinner
- Keith Peters
- Andrea Michelle
- Flash and math
- gotoAndLearn
- Google 😊
- Wikipedia
  
- C# tutorials and photoshop tutorials