Intro to Flash Player 10

Ryan Taylor

2008.08.19

New Features

- Pixel Bender / Shader support
- Foundational 3D support
- Drawing API 2.0
- Improved FileReference
- Dynamic sound capabilities
- Vectors (typed arrays)
- Low-level text support

New Features Presentation Focus

- Pixel Bender / Shader support
- Foundational 3D support
- Drawing API 2.0
- Improved FileReference
- Dynamic sound capabilities
- Vectors (typed arrays)
- Low-level text support

Getting Started

- Grab a recent Flex SDK build (2008.05.15 or later)
- Add the SDK to Flex Builder 'sdks' directory
- In Flex Builder, create a new ActionScript project
- Click the 'Configure Flex SDKs...' link

Getting Started

- Add a listing for the new SDK
- Target the SDK for your project
    - In the build path window, toggle to 'Library path' tab
    - Remove the 'playerglobal.swc' listing under the SDK

Getting Started

- Click the 'Add SWC' button and browse to the new SDK
- Browse to 'frameworks/libs/player/10/' and select 'playerglobal.swc'
- Once added, click the toggle-arrow next to 'playerglobal.swc'

Getting Started

- Double-click 'Link Type' and set it to 'External'
- Finish setting up your new project

Publishing

- Make sure HTML wrapper requires FP 10.0.0
- Open SWF in hex editor; 4th byte should be '0A'

Code Assist

- Works for all new classes
- Pre-existing classes with changes have issues

Vector

- Typed Array
- Significantly better performance than Array
- Format: Vector.<Type>

Vector

- var verticies:Vector.<Number> = new Vector.<Number>();
- verticies.push(0, 0, 100, 0, 0, 100);
- var verticies:Vector.<Number> = Vector.<Number>([0, 0, 100, 0, 0, 100]);
- var verticies:Vector.<Number> = new Vector.<Number>(6);

Graphics API 2.0

- drawPath
- drawGraphicsData
- drawTriangles

Graphics API 2.0 drawGraphicsData

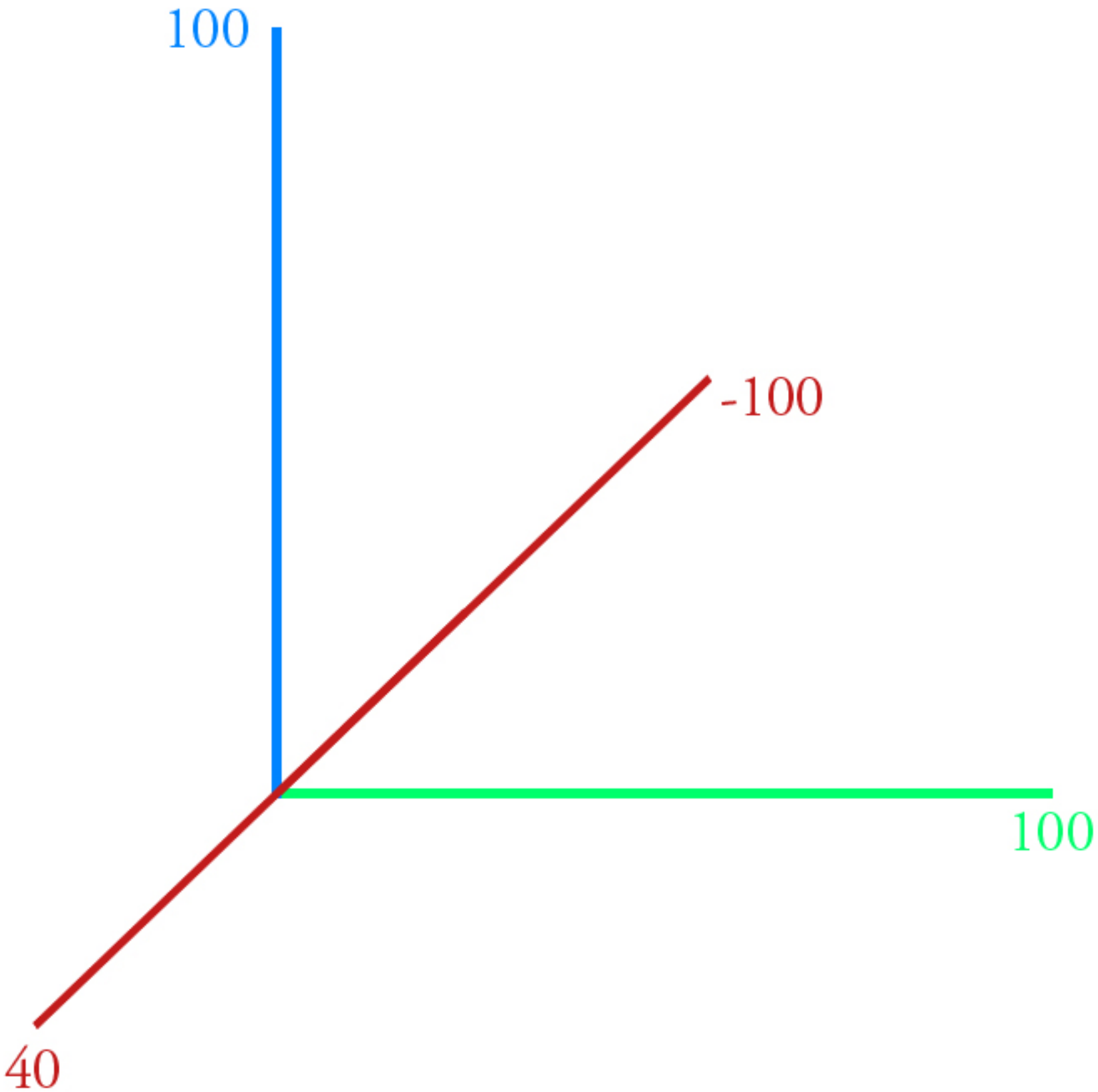
- fills - GraphicsSolidFill, GraphicsGradientFill, GraphicsBitmapFill, GraphicsShaderFill
- strokes - GraphicsStroke
- paths - GraphicsPath, GraphicsPathCommand

Graphics API 2.0
drawGraphicsData

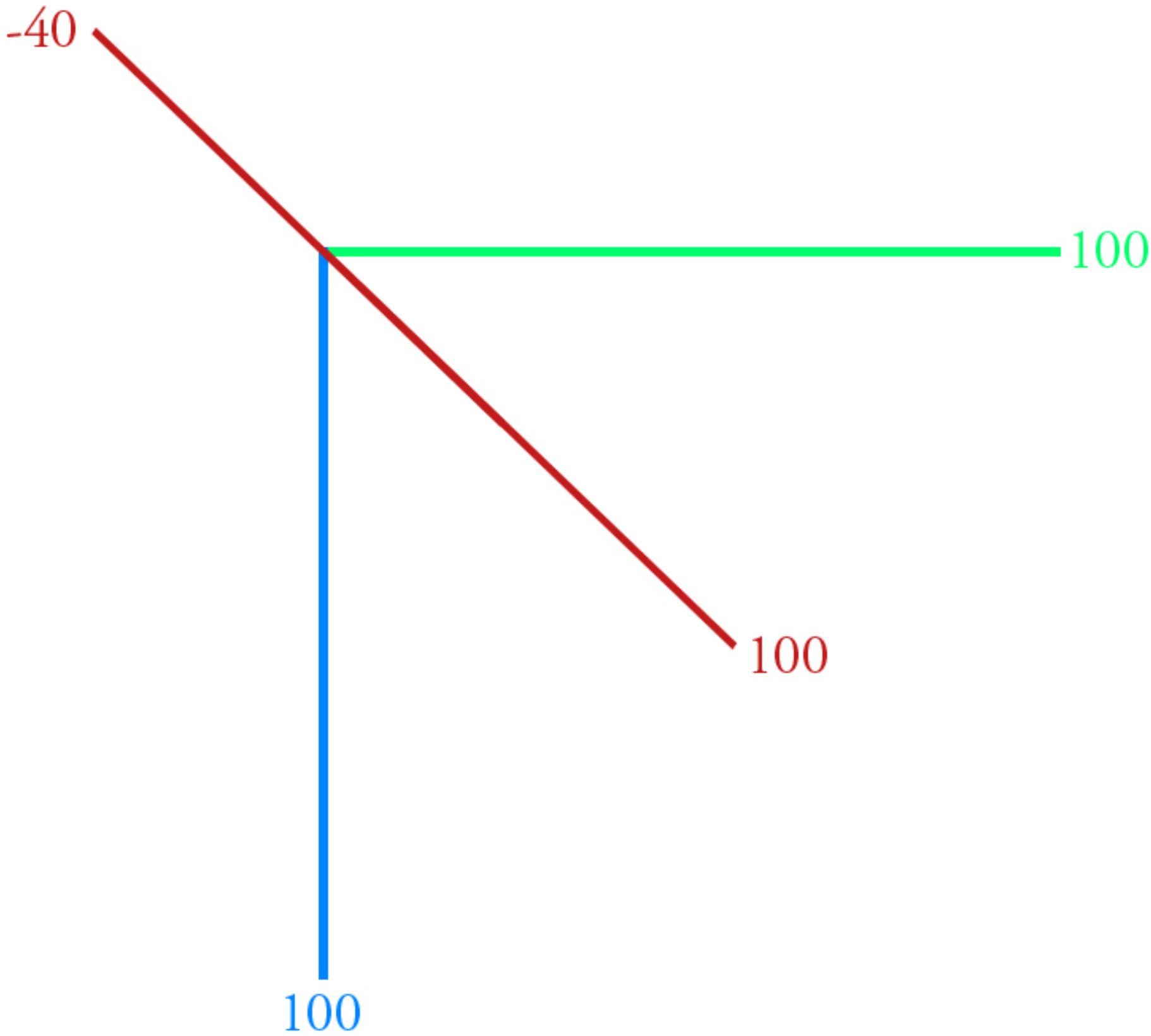
- var graphicsData:Vector.<IGraphicsData> = new Vector.<IGraphicsData>();
- graphicsData.push(fill, path);
- graphics.drawGraphicsData(graphicsData);

Foundational 3D Support Basics

- DisplayObjects now have a Z axis (z, rotationZ, etc)
- Right-handed coords, flipped 180 around X axis
- DisplayObject.transform.matrix3D
- DisplayObject.transform.perspectiveProjection
- z does not trump depth



Standard right-handed coordinates



Flash coordinates, right-handed rotated 180 degrees around X axis

Foundational 3D Support PerspectiveProjection

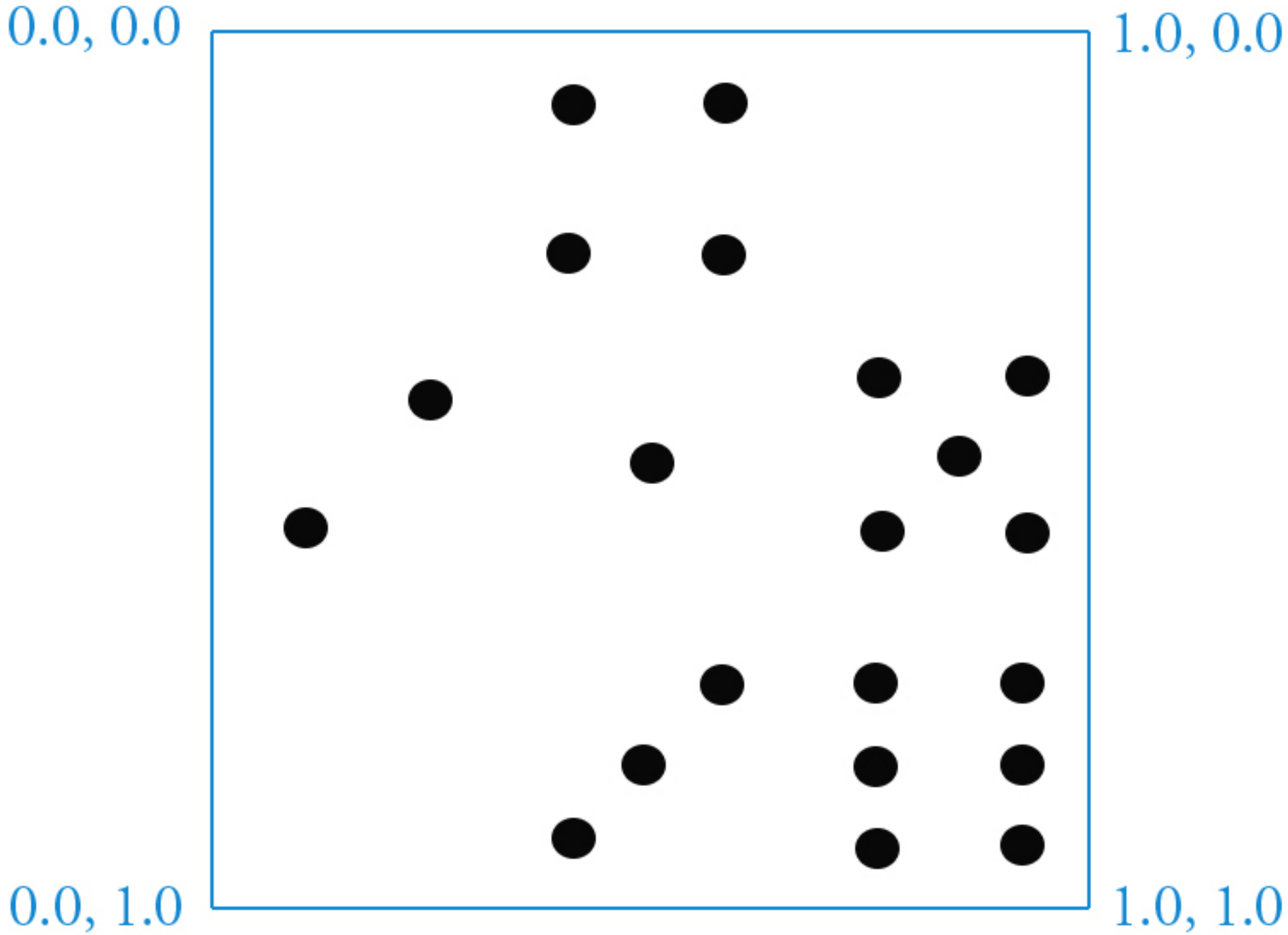
- Projection matrix
- Use 'fieldOfView' to set focalLength
- focalLength = centerY / Math.tan((fieldOfView * (Math.PI / 180)) * 0.5);

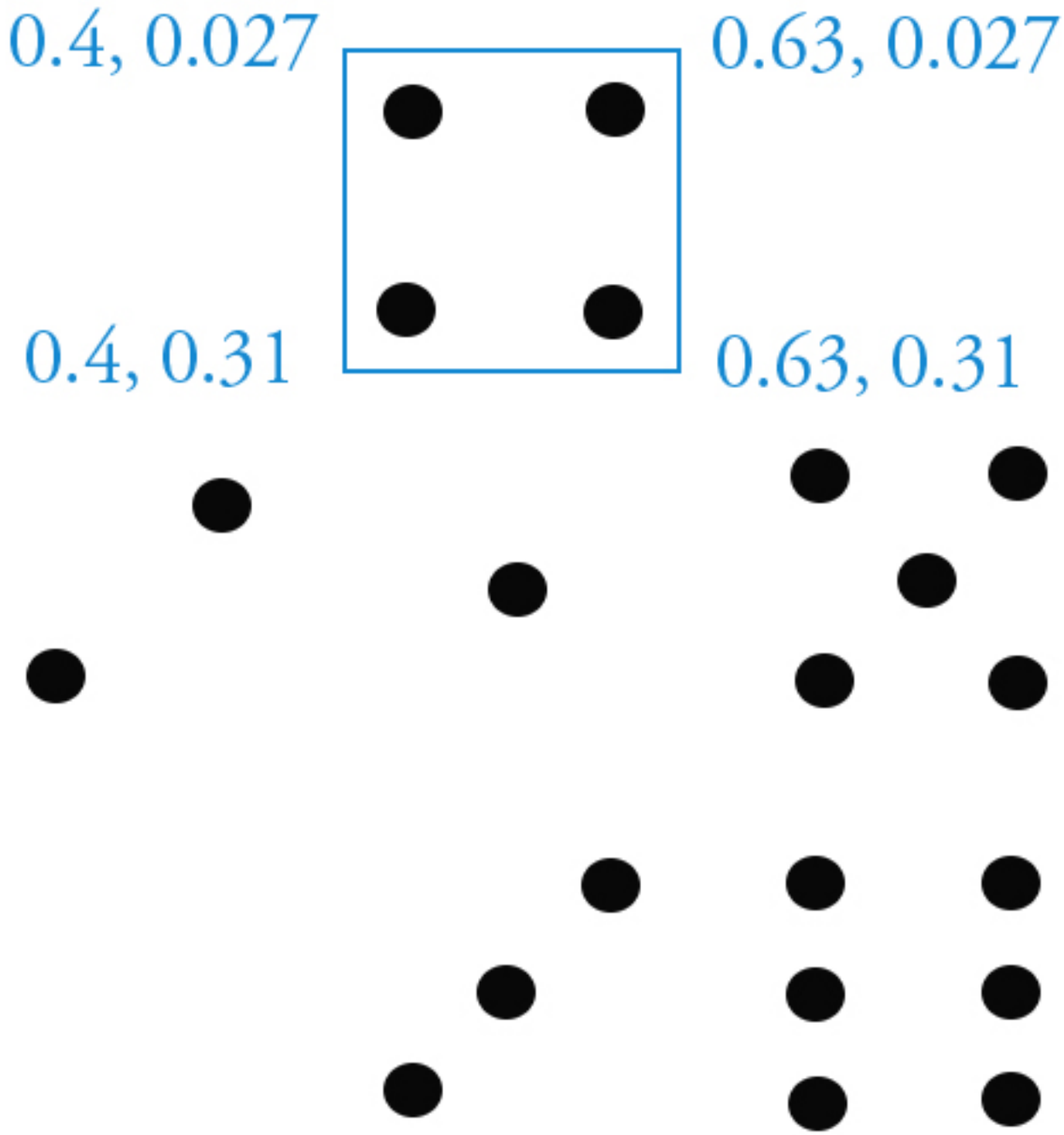
Foundational 3D Support Homogeneous Coordinates

- For affine transformations; projective space
- Represented as (x, y, w) or (x, y, z, w)
- w is a scalar
- Applied as (x * w, y * w)

Foundational 3D Support Homogeneous Coordinates

- Vector3D has x, y, z, and w
- UV coords refer to w as t in Flash (u, v, t)
- w = focalLength / (focalLength + z);


0.0, 0.0
1.0, 0.0
0.0, 1.0
1.0, 1.0


0.4, 0.027
0.63, 0.027
0.4, 0.31
0.63, 0.31

Foundational 3D Support
3D Utilities

- Utils3D.pointTowards
- Utils3D.projectVector
- Utils3D.projectVectors
- Utils3D.projectVectors(projection.toMatrix3D(), verticies3D, verticies2D, uvtData);

Foundational 3D Support
Graphics.drawTriangles

- New in Drawing API 2.0
- Better performance and improved texture quality

Foundational 3D Support Graphics.drawTriangles

- plane.graphics.beginBitmapFill(texture.bitmapData);
- plane.graphics.drawTriangles(verticies, indices, uvtData, TriangleCulling.POSITIVE);
- plane.graphics.endFill();

Foundational 3D Support Graphics.drawTriangles

- Culling increases performance by drawing only what is visible to the camera
- Culling uses sign of dot product of camera (view plane) vector and surface normal of triangle
- Culling can be NONE, POSITIVE, or NEGATIVE

Pixel Bender

- 2D pixel shader language
- Similar to HLSL and GLSL, but only fragment (aka pixel) shading
- Easy to create custom filters (kernels) for Adobe products

Pixel Bender
Flash Player Limitations

- ShaderModel 3.0 targeted for Photoshop, After Effects
- ShaderModel 2.0 targeted for Flash Player
- Needed to support wider range of hardware
- May target ShaderModel 3.0 for FP11

Pixel Bender Shaders In Flash

- Can be used as:
  - Fill - Graphics.beginShaderFill
  - Filter - ShaderFilter with BitmapData.applyFilter or DisplayObject.filters
  - Blend mode - DisplayObject.blendShader
  - Generic number crunching - ShaderJob

Pixel Bender Shaders In Flash

- Can be loaded externally or embedded

- [Embed(source="/../assets/filters/MyFilter.pbj", mimeType="application/octet-stream")]
  protected var MyFilter:Class;

Pixel Bender Shaders In Flash

- var shader:Shader = new Shader(new MyFilter());
- shader.data.myInput.input
- shader.data.myParameter.value

Pixel Bender
Generic Number Crunching

- Useful for complex calculations; i.e. audio effect filter
- Use ShaderJob
- Processes shader on different thread than ActionScript
- Dramatically faster than optimized ActionScript

Pixel Bender Generic Number Crunching

- ShaderJob dispatches an event when the shader has completed
- Set Pixel Bender kernel input to image1
- Set Pixel Bender kernel output to pixel3 or pixel4
- NOTE: output should be pixel1 or pixel2, but a current limitation in Pixel Bender Toolkit prevents this

Ryan Taylor
rtaylor@schematic.com

More FP10 info at:
www.boostworthy.com/blog