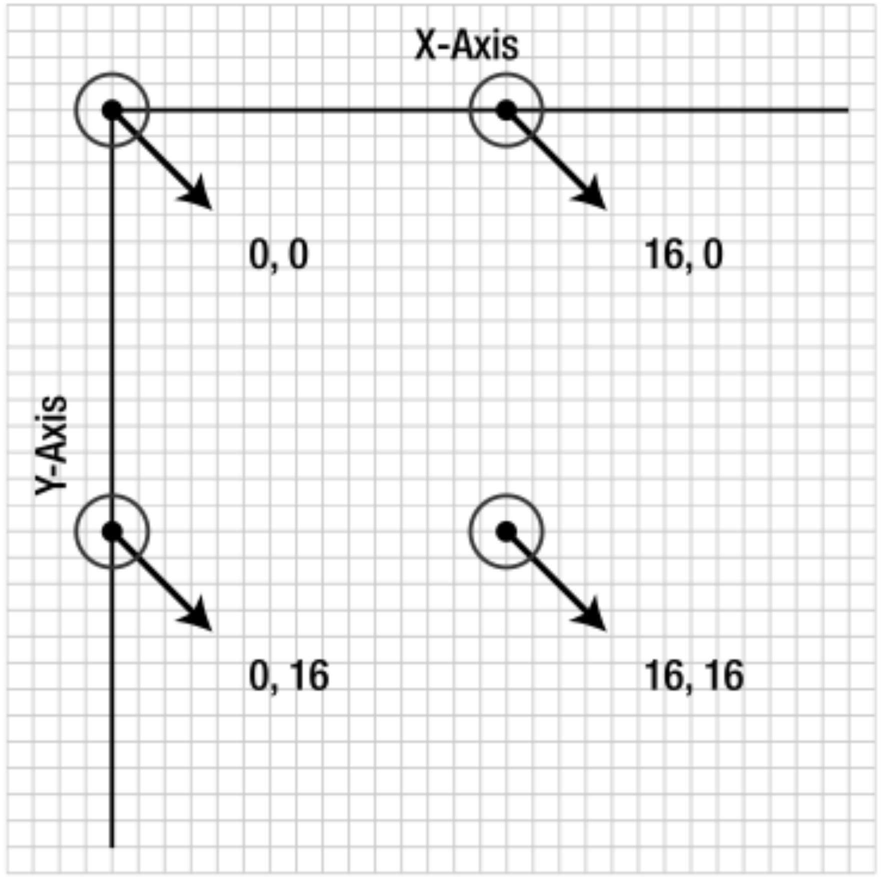
USTH MM2.1
Soft. Eng. for Interactive Media

Lecture #3.2 – Images, graphics and drawing on canvas

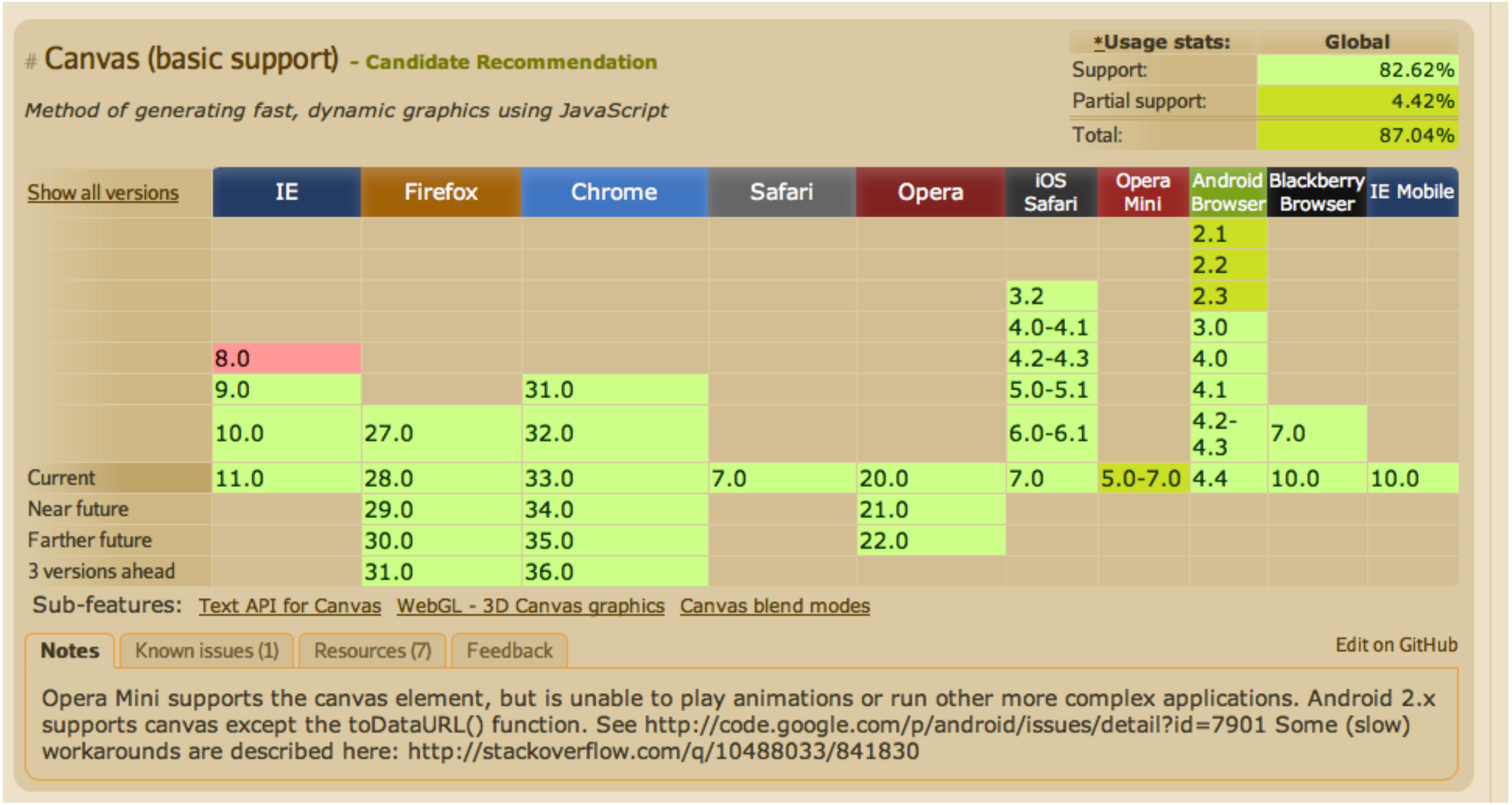
What is a canvas?

- To put it simply, it's a rectangular area in a web page.
- Once you have added a <canvas> element to your page, you can use JavaScript to manipulate it any way you want.
  - You can add graphics, lines, and text to it; you can draw on it; and you can even add advanced animations to it.
- The HTML5 Canvas API supports the same 2D drawing operations that most modern operating systems and frameworks support.
- To programmatically use a canvas, you have to first get its context.
  - You can then perform actions on the context and finally apply those actions to the context.

Canvas coordinates



Browser support



# Canvas (basic support) - **Candidate Recommendation**

Method of generating fast, dynamic graphics using JavaScript

*Usage stats:	Global
Support:	82.62%
Partial support:	4.42%
Total:	87.04%

Show all versions	IE	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Blackberry Browser	IE Mobile
								2.1		
								2.2		
						3.2		2.3		
						4.0-4.1		3.0		
	8.0					4.2-4.3		4.0		
	9.0		31.0			5.0-5.1		4.1		
	10.0	27.0	32.0			6.0-6.1		4.2-4.3	7.0	
Current	11.0	28.0	33.0	7.0	20.0	7.0	5.0-7.0	4.4	10.0	10.0
Near future		29.0	34.0		21.0					
Farther future		30.0	35.0		22.0					
3 versions ahead		31.0	36.0							

Sub-features: Text API for Canvas WebGL - 3D Canvas graphics Canvas blend modes

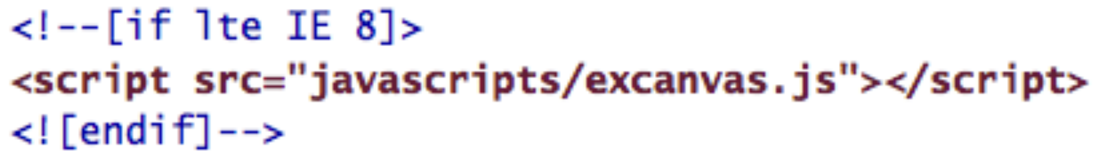
Notes Known issues (1) Resources (7) Feedback Edit on GitHub

Opera Mini supports the canvas element, but is unable to play animations or run other more complex applications. Android 2.x supports canvas except the toDataURL() function. See http://code.google.com/p/android/issues/detail?id=7901 Some (slow) workarounds are described here: http://stackoverflow.com/q/10488033/841830

Source: http://caniuse.com/#search=canvas

Browser support

- More current sources state that all latest versions of contemporary browsers (incl. IE9) support the <canvas> element.
- If you still need help making it work for earlier versions of IE, check http://code.google.com/p/explorercanvas/
- In summary, although the use of the <canvas> element is ever growing, consider it as a fall-back strategy.



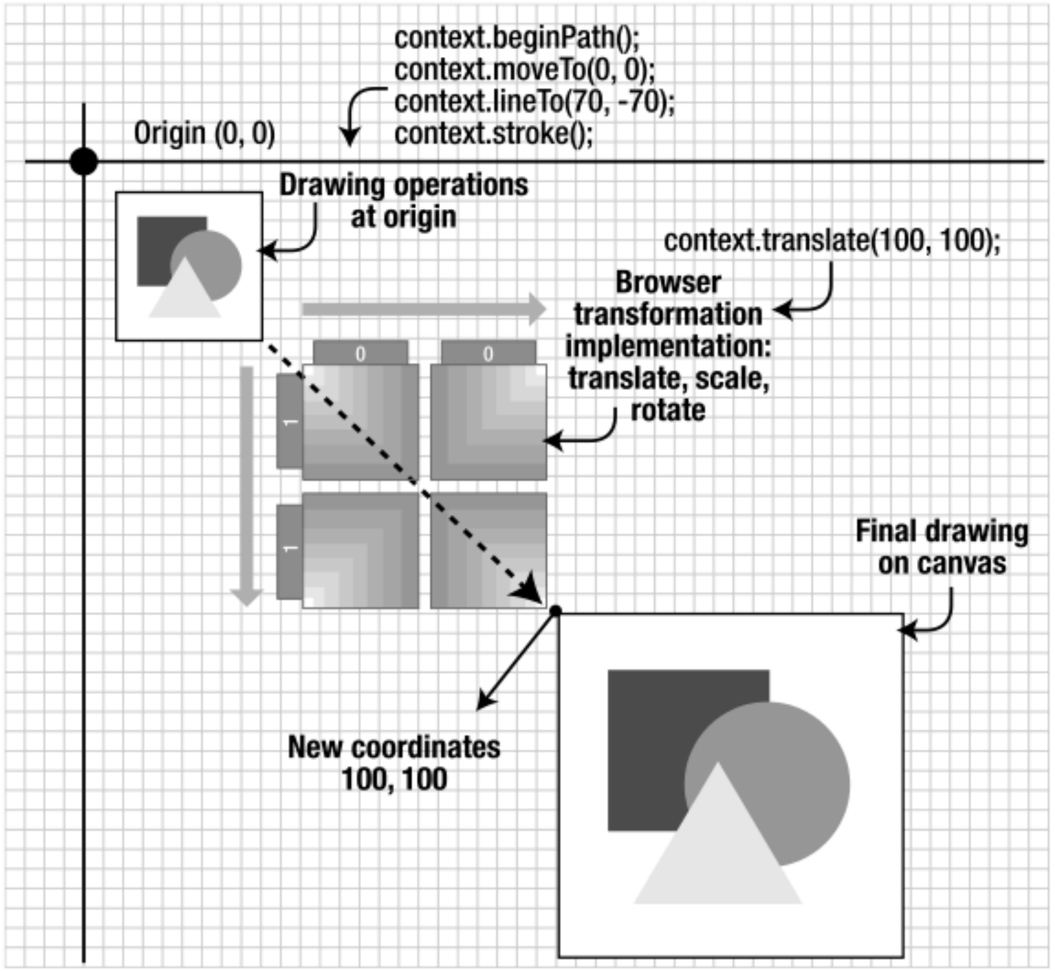
```
<!--[if lte IE 8]>
<script src="javascripts/excanvas.js"></script>
<![endif]-->
```

Transformations

- Transformations are considered best practice for more complex canvas operations; they are critical to understanding the HTML5 Canvas API's complex capabilities.
- You can think of the transformation system as a modification layer that sits between the commands you issue and the output on the canvas display.
  - This modification layer is always present, even if you choose not to interact with it.
- Transformations can be applied sequentially, combined, and modified at will.
  - Every drawing operation is passed through the modification layer to be modified before it appears on the canvas.

Transformations

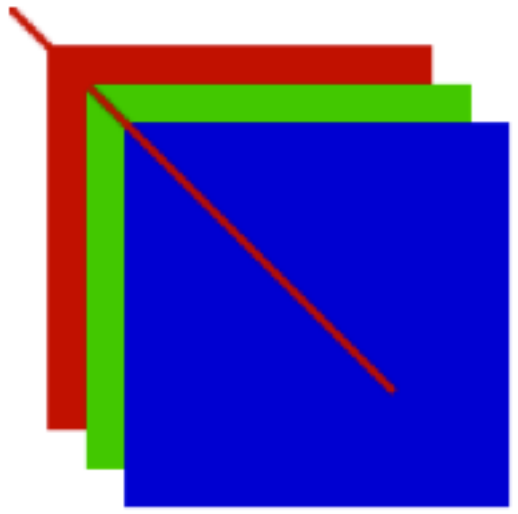
- A key recommendation for reusable code is that you usually want to draw at the origin (coordinate 0,0) and apply transformations – scale, translate, rotate, and so forth – to modify your drawing code into its final appearance, as shown on the right.



A simple example

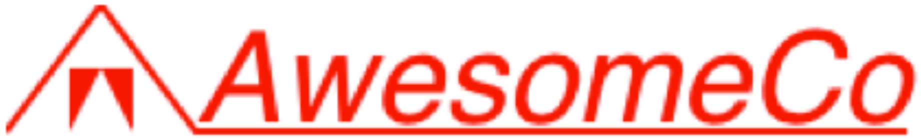
- Drawing lines and rectangles
- Live at http://media.pragprog.com/titles/bhh5/code/html5canvasgraph/canvas_simple_drawing.html

```
<canvas id="my_canvas" width="150" height="150">
  Fallback content here
</canvas>

<script type="text/javascript" charset="utf-8">
  var canvas = document.getElementById('my_canvas');
  if (canvas.getContext){
    var context = canvas.getContext('2d');
       context.fillStyle = "rgb(200,0,0)";
       context.fillRect (10, 10, 100, 100);
       context.fillStyle = "rgb(0,200,0)";
       context.fillRect (20, 20, 100, 100);
       context.fillStyle = "rgb(0,0,200)";
       context.fillRect (30, 30, 100, 100);
       context.strokeStyle = "rgb(200,0,0)";
       context.lineWidth = 2;
       context.beginPath();
       context.moveTo(0, 0);
       context.lineTo(100, 100);
       context.stroke();
       context.closePath();
  }else{
    // do something to show the canvas' hidden contents
    // or let the browser display the text within the <canvas> element.
  }
</script>
```

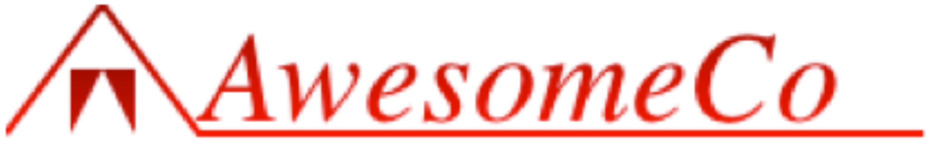


Another example

- Drawing a simple logo
- Live at http://media.pragprog.com/titles/bhh5/code/html5canvasgraph/logo.html



  – Variant (with a gradient): http://media.pragprog.com/titles/bhh5/code/html5canvasgraph/logo_gradient.html

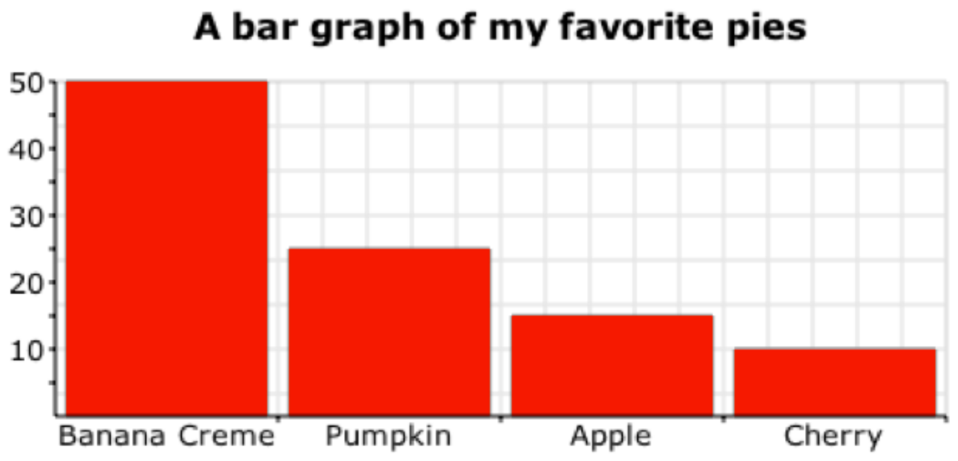


Getting help from libraries

- Drawing logos line by line, shape by shape, isn't exactly fun (or effective).
- For serious HTML5 canvas graphics, consider using libraries such as:
  - RGraph: http://www.rgraph.net/
  - MooTools: http://forvar.de/js/mcl/index.html
  - jCanvaScript: http://jcscript.com/

Example

- Bar graph example using Rgraph
- Live at: http://media.pragprog.com/titles/bhh5/code/html5canvasgraph/rgraph_bar_example.html



Example

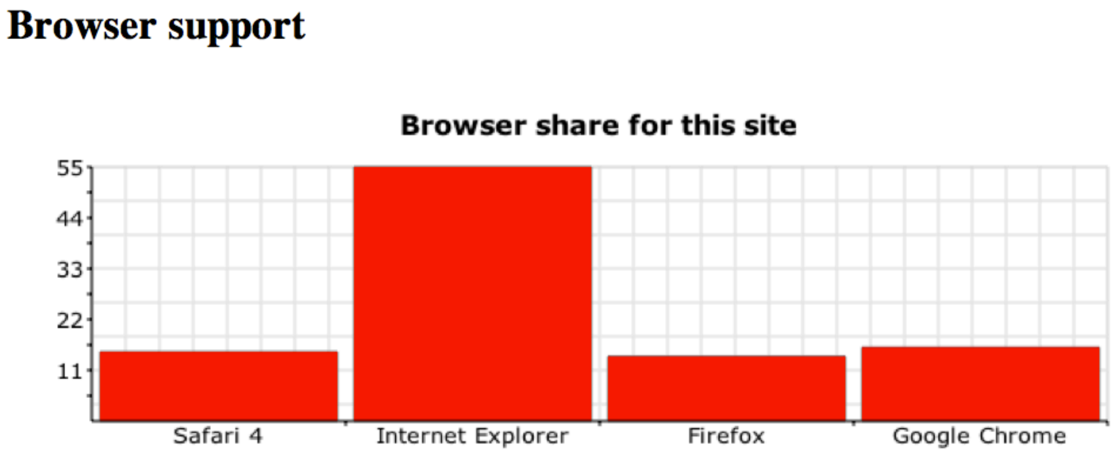
- Bar graph example using Rgraph
- Live at: http://media.pragprog.com/titles/bhh5/code/html5canvasgraph/rgraph_bar_example.html

```
<html>
  <head>
    <title>Graph</title>
    <script src="javascripts/RGraph.common.js" ></script>
    <script src="javascripts/RGraph.bar.js" ></script>
  </head>
  <body>
  <canvas width="500" height="250" id="test">[no canvas support]</canvas>

  <script type="text/javascript" charset="utf-8">
    var bar = new RGraph.Bar('test', [50,25,15,10]);
    bar.Set('chart.gutter', 50);
    bar.Set('chart.colors', ['red']);
    bar.Set('chart.title', "A bar graph of my favorite pies");
    bar.Set('chart.labels', ["Banana Creme", "Pumpkin", "Apple", "Cherry"]);
    bar.Draw();
  </script>

  </body>
</html>
```

Another example

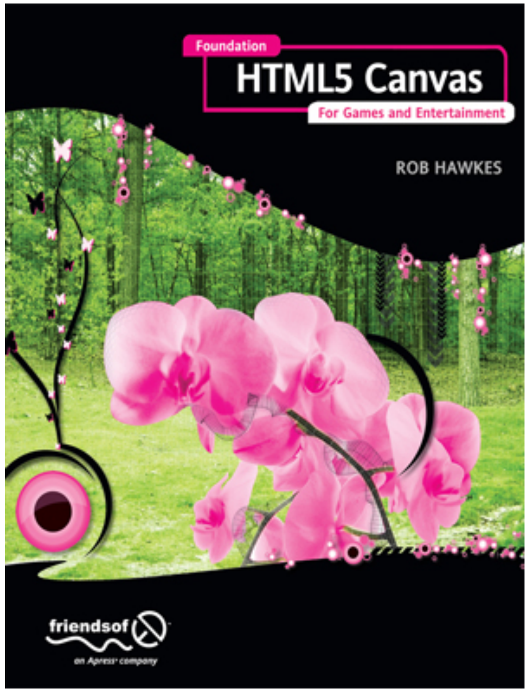
- Bar graph with data from HTML
- Live at: http://media.pragprog.com/titles/bhh5/code/html5canvasgraph/canvas_graph.html

- Includes hard-coded fallback function divGraph() for browsers that don't support the <canvas> element – and therefore cannot execute canvasGraph().



Learn more about it

- The HTML5 <canvas> element and Canvas API are rich enough to deserve an entire book…
- … covering their *foundations*.
- http://rawkes.com/foundationcanvas



Learn more about it

- Another book focusing primarily on the HTML5 <canvas> element and Canvas API
  - http://shop.oreilly.com/product/0636920013327.do

Native Interactivity and Animation for the Web
HTML5 Canvas
O'REILLY
Steve Fulton & Jeff Fulton