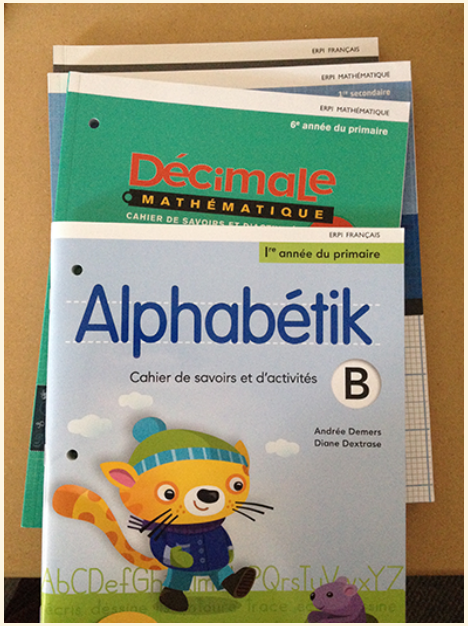
# HELLO

Front-End Web Developer at

CREATIVE SOAPBOX™

# FOR THE PAST 2 YEARS

Building an educational application for Pearson ERPI in Canada.




# DEMO

http://erpi.dev/alphabetik/

# OVERVIEW

- Demo
- What is Canvas
- What is Fabric.js
- Advantages of Fabric.js

# NATIVE CANVAS

*The canvas element provides scripts with a resolution-dependent bitmap canvas, which can be used for rendering graphs, game graphics, art, or other visual images on the fly.*

Canvas is a rectangle on the page where you can use Javascript to draw stuff.

# CANVAS EXPERIMENT

http://andrew-hoyer.com/experiments/cloth/

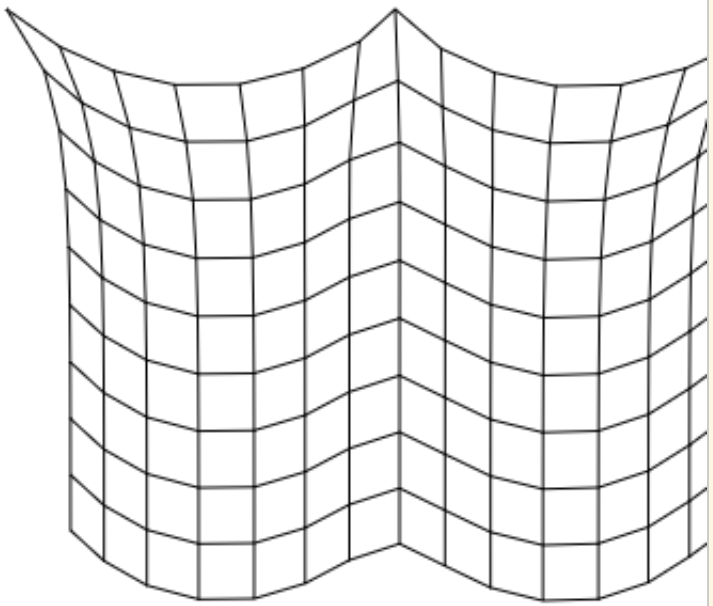
## The Cloth Simulation

Home/Experiments

Every line in the cloth simulation is technically called a constraint and every point is a point mass (an object with no dimension, just location and mass). All the constraints do is control the distance between each point mass. If two points move too far apart, it will pull them closer. If two points are too close together, it will push them apart. The cloth is really then just a collection of constraints and point masses in a never ending struggle.

Like 1.7k Tweet 550

This is a Chrome Experiment



Click and drag to move points. Hold down any key to p

Draw Lines Draw Points

# CANVAS CHARTS: CHARTJS.ORG

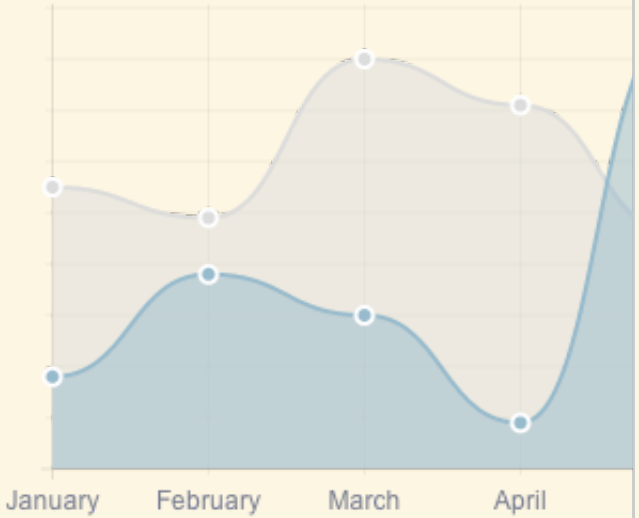
http://www.chartjs.org/

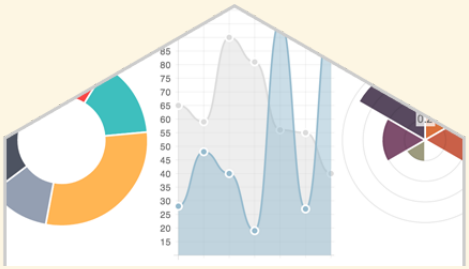
## Chart.js

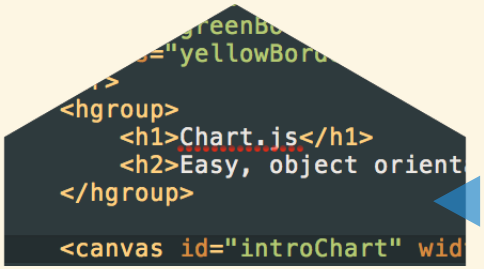
Easy, object oriented client side graphs for designers and developers

Documentation

Download



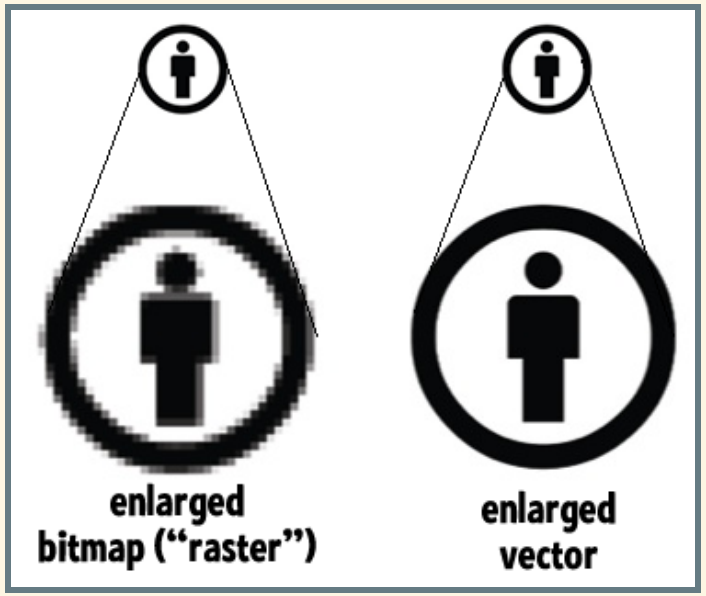





# CANVAS GRAPHICS EXAMPLE

http://hakim.se/experiments/html5/404/netmag.html

# BITMAP vs. VECTOR



# CANVAS API

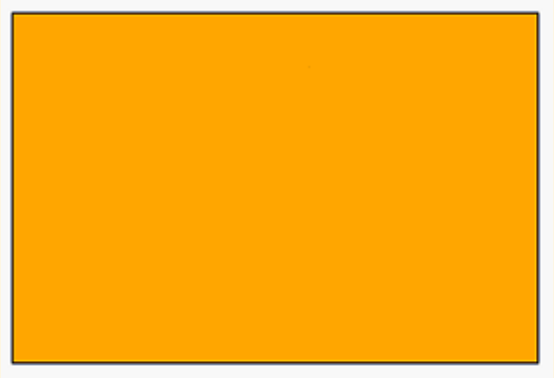
## SAMPLE ATTRIBUTES & METHODS

- strokeStyle
- fillStyle
- fillRect(x, y, width, height)
- strokeRect(x, y, width, height)
- clearRect(x, y, width, height)
- beginPath()
- closePath()
- lineTo(x,y)
- moveTo(x,y)
- save()
- restore()

# JUST THE CANVAS

## HTML

```
<canvas id="myCanvas" width="300" height="200" style="borde
r:1px solid black; background-color: orange;"></canvas>
```



# SIMPLEST CANVAS EXAMPLE

## HTML

```
<canvas id="myCanvas" width="300" height="200" style="borde
r:1px solid black; background-color: orange;"></canvas>
```

## JS

```
var myCanvas = document.getElementById("myCanvas");
var ctx = myCanvas.getContext("2d");
// (x, y, width, height)
ctx.fillRect(20, 30, 150, 75);
```

# NATIVE CANVAS

## SIMPLE RECTANGLE

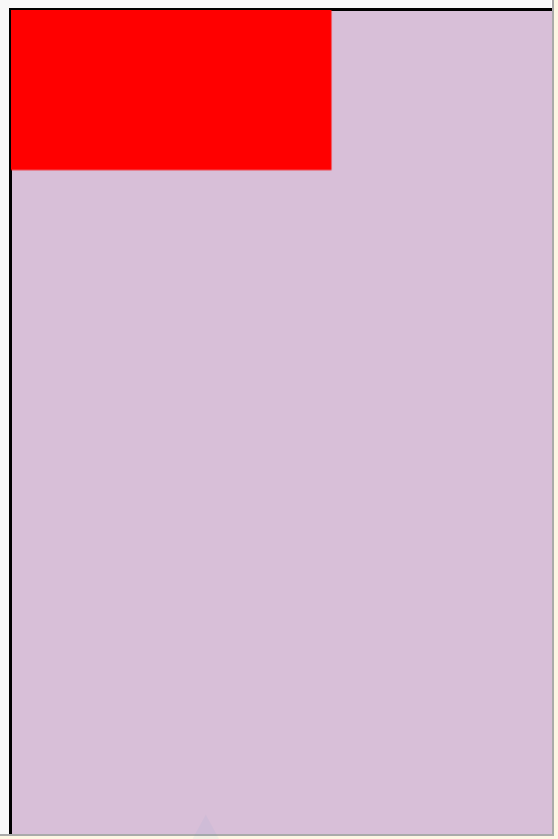
JS Bin Save HTML CSS JavaScript Console Output Help

HTML

```html
<!DOCTYPE html>
<html>
<head>
<meta name="description"
content="simple rectangle" />
  <meta charset="utf-8">
  <title>JS Bin</title>
</head>
<body>
  <canvas id="myCanvas"
width="300" height="400"
style="border:1px solid black;
background-color: thistle;">
</canvas>
</body>
</html>
```

JavaScript

```javascript
var
c=document.getElementById("myCa
nvas");
var ctx=c.getContext("2d");

ctx.fillStyle = "red";
ctx.fillRect(0,0,150,75);
```



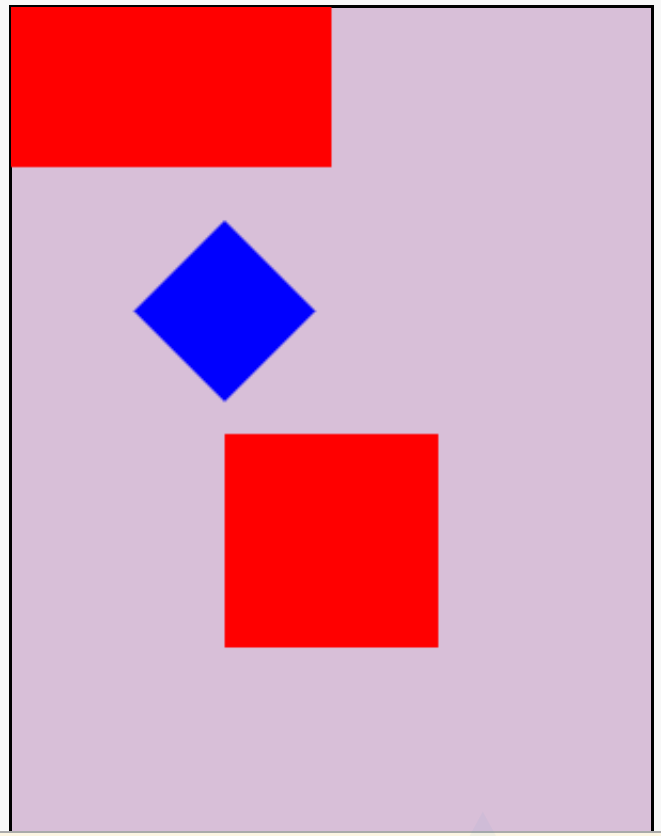
# NATIVE CANVAS

## SAVE STATE, ROTATE, RESTORE

JS Bin Save HTML CSS JavaScript Console Output Help

JavaScript

```
var c=document.getElementById("myCanvas");
var ctx=c.getContext("2d");

ctx.fillStyle = "red";
ctx.fillRect(0,0,150,75);

// save red style, no rotation
ctx.save();
// create yellow rectangle
ctx.fillStyle = "blue";
// move x, y
ctx.translate(100, 100);
// 1 radian = 57.2957795 degrees
// Math.PI / 180 * degrees
// rotate 45 degrees
ctx.rotate(Math.PI / 180 * 45);
// x: 100, y: 100 from translate line
ctx.fillRect(0, 0, 60, 60);
// restore to red, no rotation
ctx.restore();

ctx.fillRect(100, 200, 100, 100);
```



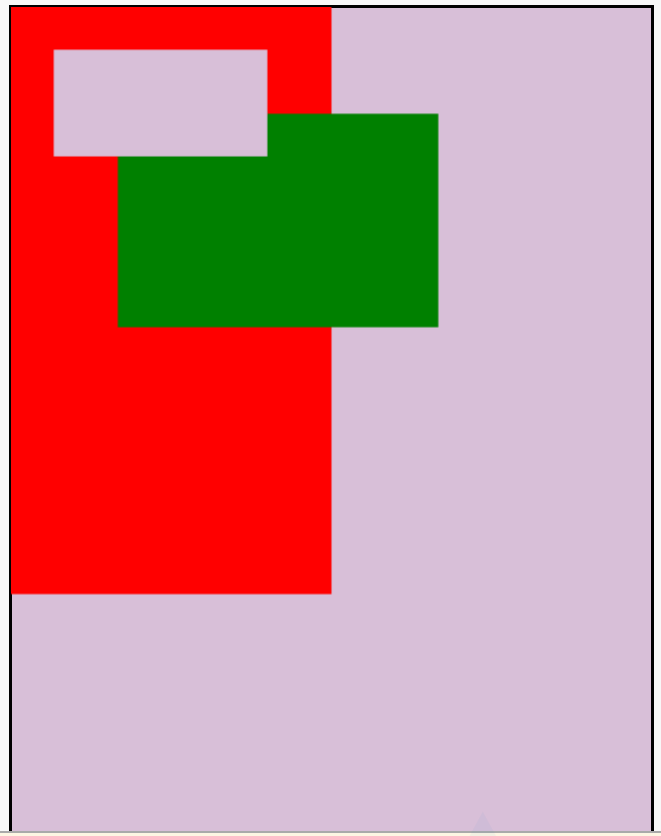
# NATIVE CANVAS

## CLEAR RECTANGLE

JS Bin Save HTML CSS JavaScript Console Output Help

JavaScript

```
var c=document.getElementById("myCanvas");
var ctx=c.getContext("2d");

ctx.fillStyle = "red";
ctx.fillRect(0,0,150,275);

ctx.fillStyle = "green";
ctx.fillRect(50,50,150,100);

ctx.clearRect(20,20,100,50);
```



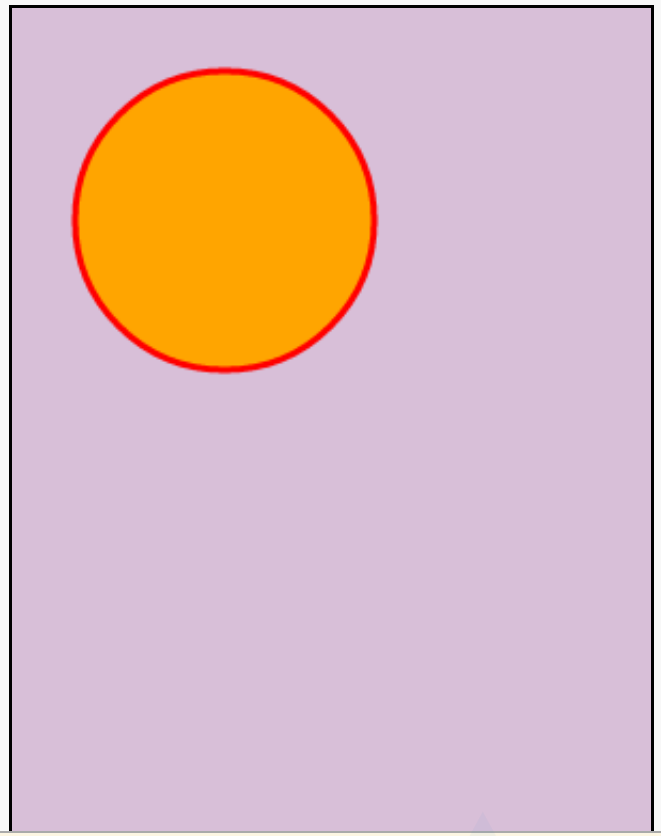
# NATIVE CANVAS

## CIRCLE

JS Bin Save HTML CSS JavaScript Console Output Help

JavaScript

```
var c=document.getElementById("myCanvas");
var ctx=c.getContext("2d");

ctx.beginPath();
// arc(x, y, radius, startAngle, endAngle,
counterClockwise);
ctx.arc(100, 100, 70, 0, 2 * Math.PI, false);
ctx.fillStyle = 'orange';
ctx.fill();
ctx.lineWidth = 3;
ctx.strokeStyle = 'red';
ctx.stroke();
```



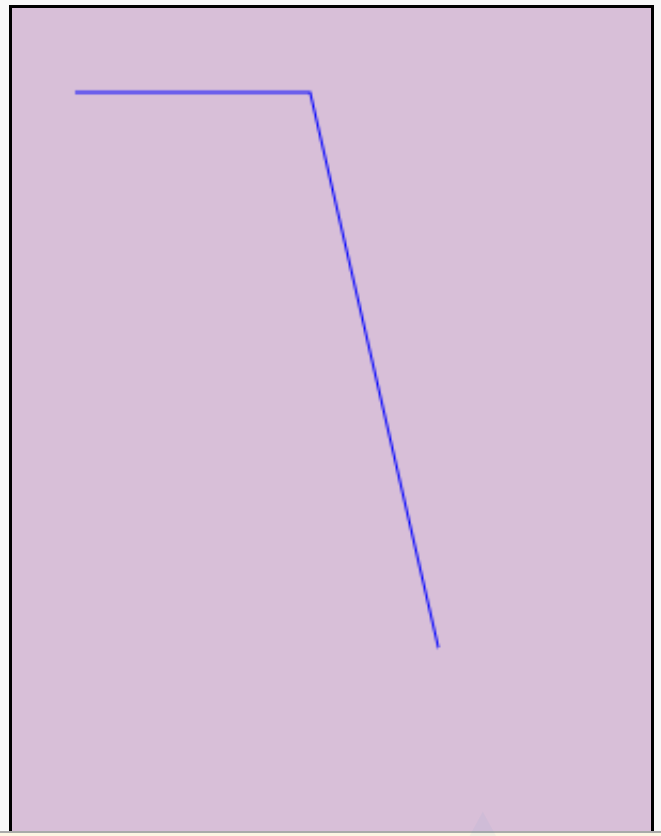
# NATIVE CANVAS

## LINE

JS Bin Save HTML CSS JavaScript Console Output Help

JavaScript

```
var c=document.getElementById("myCanvas");
var ctx=c.getContext("2d");

ctx.beginPath();
ctx.moveTo(30, 40);
ctx.lineTo(140, 40);
ctx.lineTo(200,300);
ctx.strokeStyle = 'blue';
ctx.stroke();
```



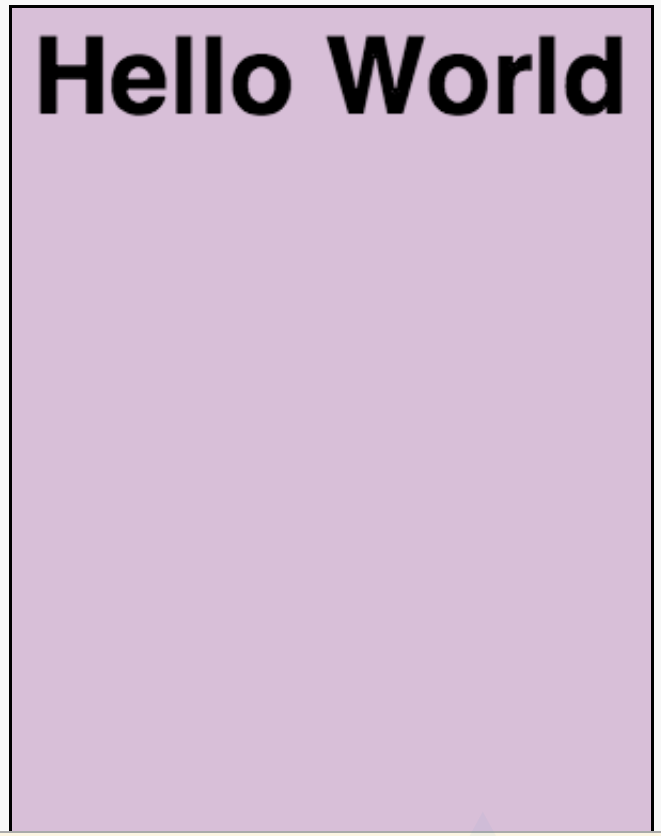
# NATIVE CANVAS

## TEXT

JS Bin Save HTML CSS JavaScript Console Output Help

JavaScript

```
var c=document.getElementById("myCanvas");
var ctx=c.getContext("2d");

ctx.font = "bold 50px Helvetica";
// (textToDraw, x, y)
ctx.fillText("Hello World", 10, 50);
```



# NATIVE CANVAS

## IMAGE

JS Bin Save HTML CSS JavaScript Console Output Help

JavaScript

```
var c=document.getElementById("myCanvas");
var ctx=c.getContext("2d");

var imageObj = new Image();
imageObj.onload = function() {
  ctx.drawImage(imageObj, 10, 10);
};
imageObj.src =
'https://c2.staticflickr.com/6/5541/11041656793
_e29178d393_n.jpg';
```

# FABRIC.JS

Fabric.js is a powerful and simple Javascript HTML5 canvas library.

Fabric provides interactive object model on top of canvas element.

# SIMPLEST FABRIC.JS EXAMPLE

## ADD FABRIC SCRIPT IN HTML

```
<script src="js/fabric.min.js"></script>
```

## HTML

```
<canvas id="myCanvas" width="300" height="200" style="borde
r:1px solid black;"></canvas>
```

## JS

```
var canvas = new fabric.Element("myCanvas");
var rect = new fabric.Rect({
  top: 100,
  left: 10,
  fill: "red",
  width: 180,
  height: 100
});
canvas.add(rect);
```

# FABRIC.JS

## SIMPLE RECTANGLE

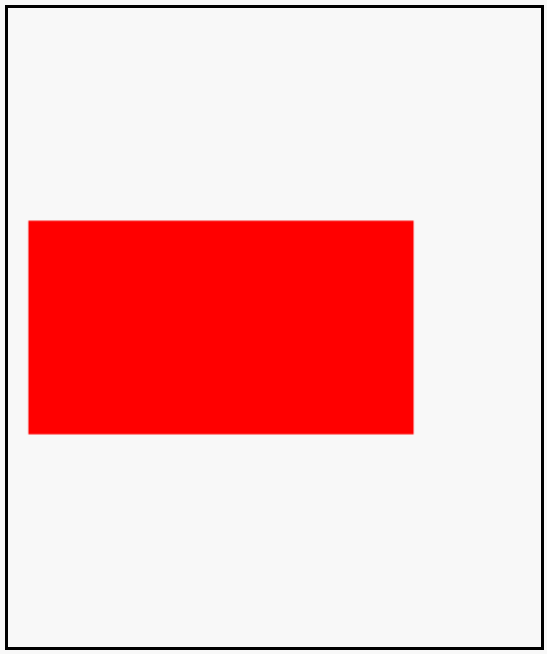
JS Bin Save HTML CSS JavaScript Console Output Help

HTML

```html
<!DOCTYPE html>
<html>
<head>
<script
src="http://127.0.0.1:8000/js/f
abric.min.js"></script>
<meta name="description"
content="fabric - simple
rectangle" />
  <meta charset="utf-8">
  <title>JS Bin</title>
</head>
<body>
  <canvas id="myCanvas"
width="250" height="300"
style="border:1px solid black">
</canvas>
</body>
</html>
```

JavaScript

```javascript
var canvas = new
fabric.Element("myCanvas");
var rect = new fabric.Rect({
  top: 100,
  left: 10,
  fill: "red",
  width: 180,
  height: 100
});
canvas.add(rect);
```



# FABRIC CREATES 2 CANVASES

## INITIAL HTML

```
<canvas id="myCanvas" width="300" height="200"></canvas>
```

## BECOMES 2 ABSOLUTELY-POSITIONED, OVERLAYING CANVASES

```
<div class="canvas-container">
  <!-- Group selection -->
  <canvas id="myCanvas" width="250" height="300" class="lower-canvas"></canvas>
  <!-- Rendering -->
  <canvas class="upper-canvas" width="250" height="300"></canvas>
</div>
```

This keeps group selection fast no matter how many objects are currently rendered on canvas.

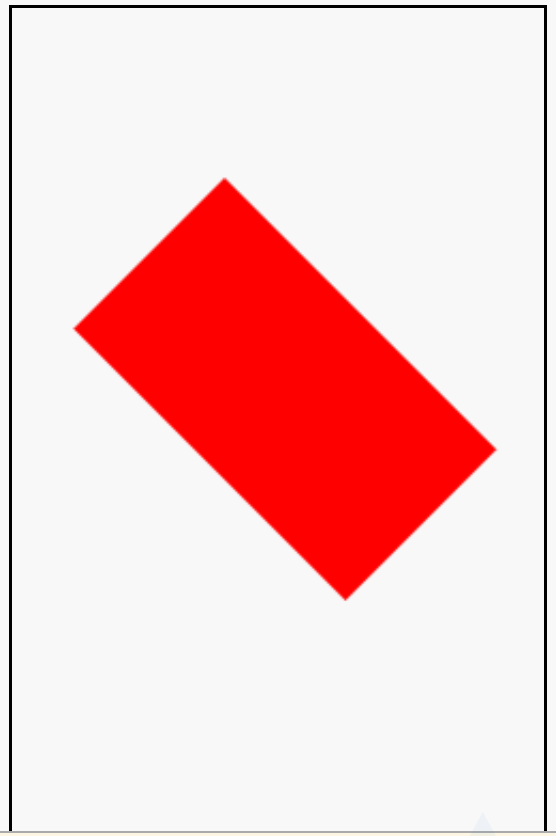
# FABRIC.JS

## SIMPLE RECTANGLE - ROTATED

JS Bin Save HTML CSS JavaScript Console Output Help

JavaScript

```
var canvas = new fabric.Element("myCanvas");
var rect = new fabric.Rect({
  top: 80,
  left: 100,
  fill: "red",
  width: 180,
  height: 100,
  // new line - in degrees
  angle: 45
});
canvas.add(rect);
```



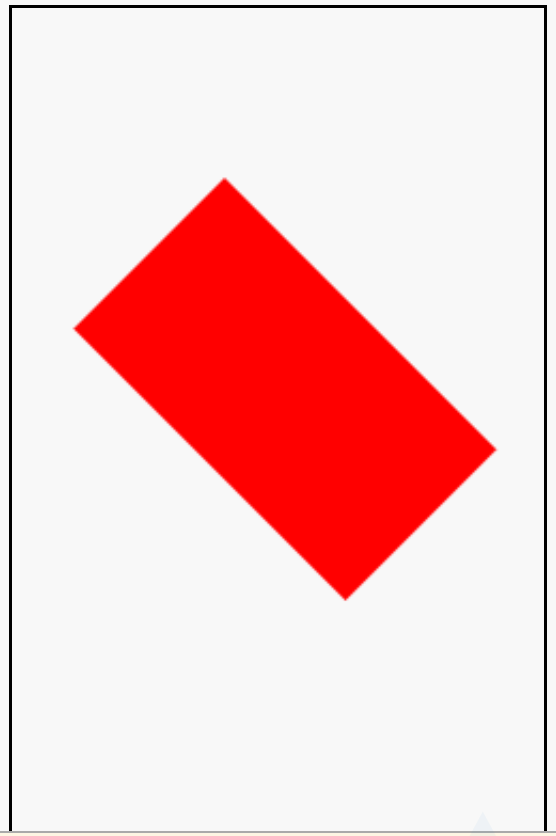
# FABRIC.JS

## SIMPLE RECTANGLE - REMOVED

JS Bin Save HTML CSS JavaScript Console Output Help

JavaScript

```
var canvas = new fabric.Element("myCanvas");
var rect = new fabric.Rect({
  top: 80,
  left: 100,
  fill: "red",
  width: 180,
  height: 100,
  // new line - in degrees
  angle: 45
});
canvas.add(rect);

//canvas.remove(rect);
```



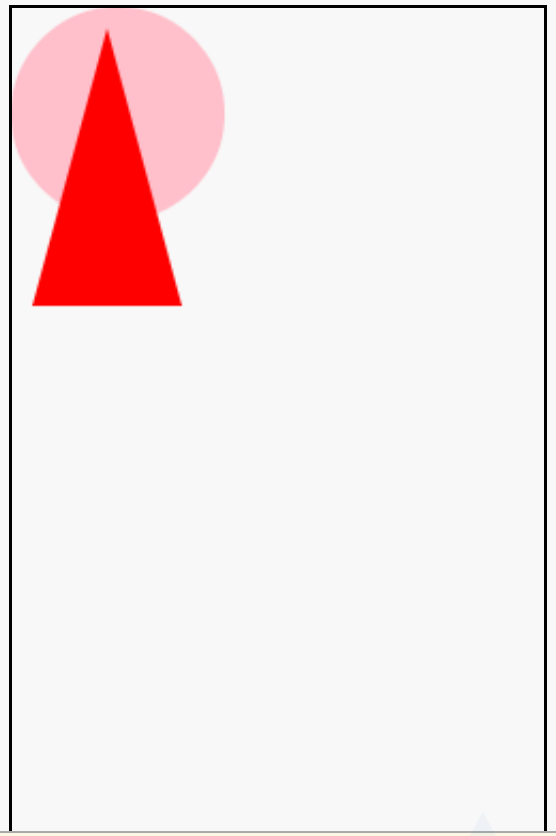
# FABRIC.JS

## SHAPES

JS Bin Save HTML CSS JavaScript Console Output Help

JavaScript

```
var canvas = new fabric.Element("myCanvas");
var circle = new fabric.Circle({
  radius: 50,
  fill: 'pink',
  left: 0,
  top: 0
});

var triangle = new fabric.Triangle({
  width: 70,
  height: 130,
  fill: 'red',
  left: 10,
  top: 10
});

canvas.add(circle, triangle);
```



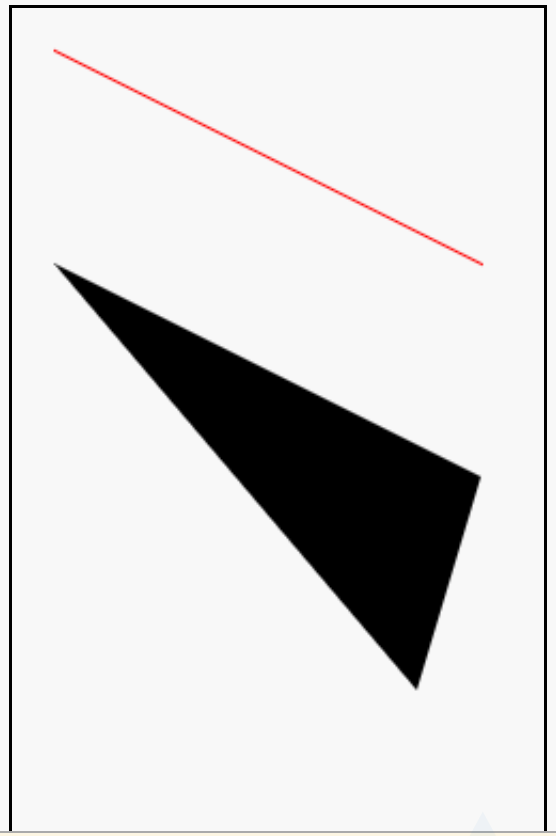
# FABRIC.JS

## CUSTOM SHAPES, LINES

JS Bin Save HTML CSS JavaScript Console Output Help

JavaScript

```
var canvas = new fabric.Element("myCanvas");
var customShape = new fabric.Path('M 0 0 L 200
100 L 170 200 z');
customShape.set({ left: 20, top: 120 });

var line = new fabric.Line([0, 0, 200, 100], {
  left: 20,
  top: 20,
  stroke: 'red'
});

canvas.add(customShape, line);
```



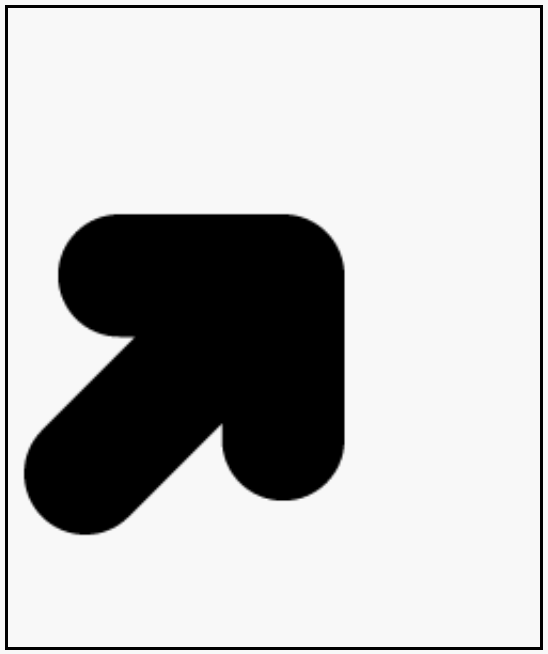
# FABRIC.JS

## SVG PARSING

JS Bin Save HTML CSS JavaScript Console Output Help

JavaScript

```
var canvas = new fabric.Element("myCanvas");
/*
fabric.loadSVGFromURL('http://localhost:3000/im
ages/awesome_tiger.svg', function(objects) {
    var group = new fabric.PathGroup(objects, {
      left: 165,
      top: 100,
      width: 295,
      height: 211
  });
  canvas.add(group);
  canvas.renderAll();
}); */

var path = new
fabric.Path('M121.32,0L44.58,0C36.67,0,29.5,3.2
2,24.31,8.41c-5.19,5.19-8.41,12.37-
8.41,20.28c0,15.82,12.87,28.69,28.69,28.69c0,0,
4.4,0,7.48,0C36.66,72.78,8.4,101.04,8.4,101.04C
2.98,106.45,0,113.66,0,121.32c0,7.66,2.98,14.87
,8.4,20.29l0,0c5.42,5.42,12.62,8.4,20.28,8.4c7.
66,0,14.87-2.98,20.29-8.4c0,0,28.26-
```



# FABRIC.JS

## TEXT

JS Bin Save HTML CSS JavaScript Console Output Help

JavaScript

```
var canvas = new fabric.Element("myCanvas");
var text = new fabric.Text('Hello World', {
  left: 10,
  top: 100,
  fontFamily: "Helvetica",
  fontWeight: "bold",
  shadow: 'rgba(0,0,0,0.3) 5px 5px 5px',
  angle: -20,
  backgroundColor: "yellow"
});
canvas.add(text);
```

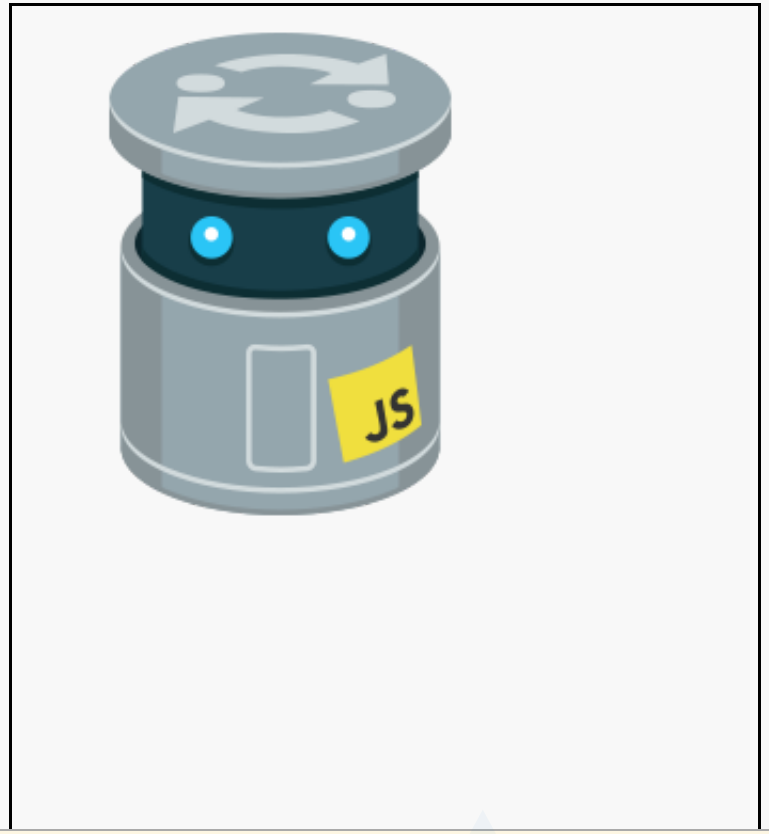
# FABRIC.JS

## IMAGE

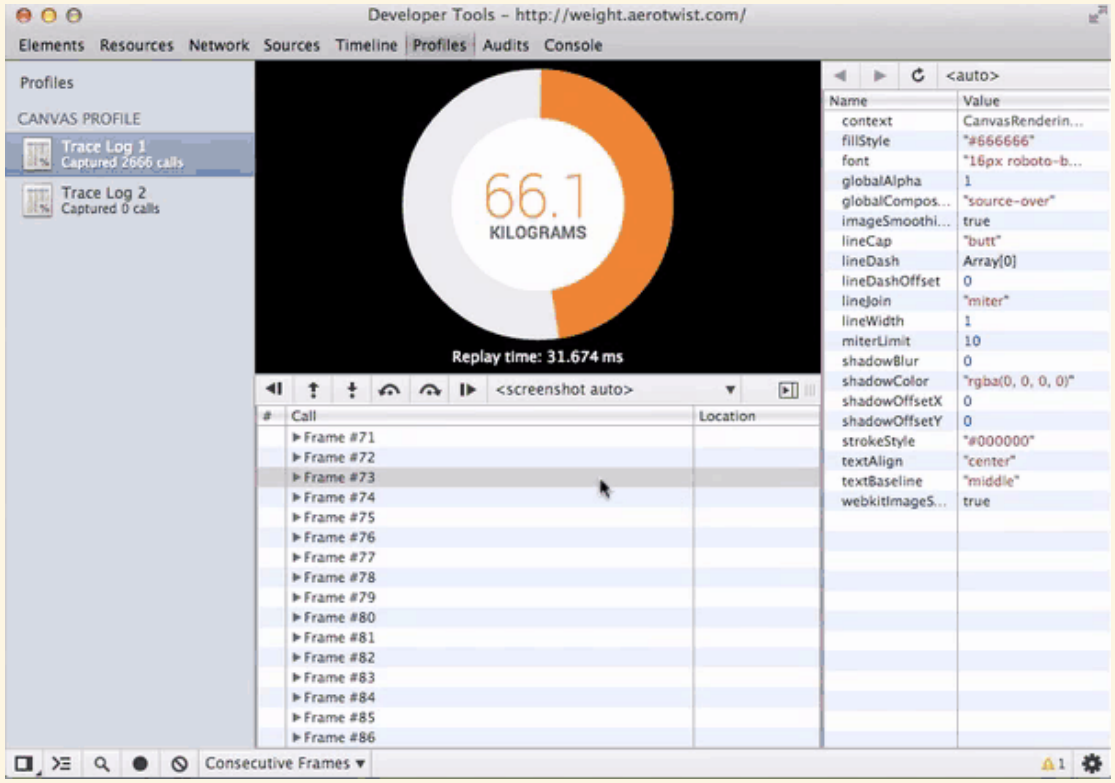
JS Bin Save HTML CSS JavaScript Console Output Help

JavaScript

```
var canvas = new fabric.Element("myCanvas");

fabric.Image.fromURL('http://localhost:3000/images/logo.png', function(img) {
  // add filter
  //img.filters.push(new fabric.Image.filters.Sepia2());

  // apply filters and re-render canvas when done

//img.applyFilters(canvas.renderAll.bind(canvas));

  // add image onto canvas
  canvas.add(img);
});
```



# CANVAS PROFILER



# PROTOTYPAL INHERITENCE

## FABRIC.OBJECT

- fabric.Line
- fabric.Circle
- fabric.Rect
- fabric.Group
- fabric.Text
- fabric.Ellipse
- fabric.Image
- fabric.Polyline
- fabric.Polygon
- fabric.Path

# PROPERTIES

- angle
- fill
- hasControls
- lockRotation
- opactiy

# METHODS

- .bringToFront()
- .clone()
- .getBoundingRect()
- .getStroke()
- .moveTo()

# GETTERS

```
rect.getWidth();

rect.getLeft();
rect.getTop();

rect.getFill(); // rgb(0,0,0)
rect.getStroke();

rect.getOpacity(); // 1
```

# SETTERS

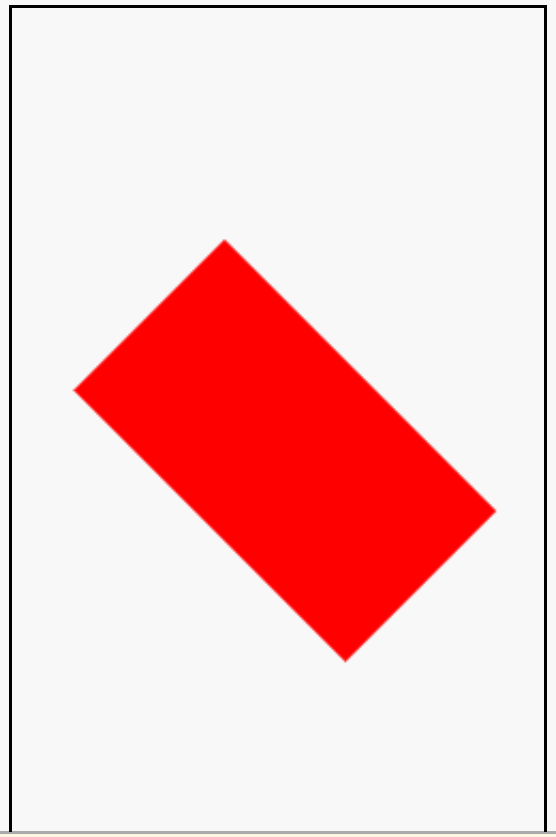
```
var rect = new fabric.Rect();
rect.set({ width: 10, height: 20, fill: '#f55', opacity: 0.
7 });
```

# ANIMATION

JS Bin Save HTML CSS JavaScript Console Output Help

JavaScript

```
var canvas = new fabric.Element("myCanvas");
var rect = new fabric.Rect({
  top: 0,
  left: 100,
  fill: "red",
  width: 180,
  height: 100,
  // new line - in degrees
  angle: 45
});
canvas.add(rect);

rect.animate('top', '+=202', {
  onChange: canvas.renderAll.bind(canvas),
  duration: 5000,
  easing: fabric.util.ease.easeOutBounce
});
```



# GROUPS

```
var text = new fabric.Text('hello world', {
  fontSize: 30
});
var circle = new fabric.Circle({
  radius: 100,
  fill: '#eef',
  scaleY: 0.5
});
var group = new fabric.Group([ text, circle ], {
  left: 150,
  top: 100,
  angle: -10
});
canvas.add(group);
```

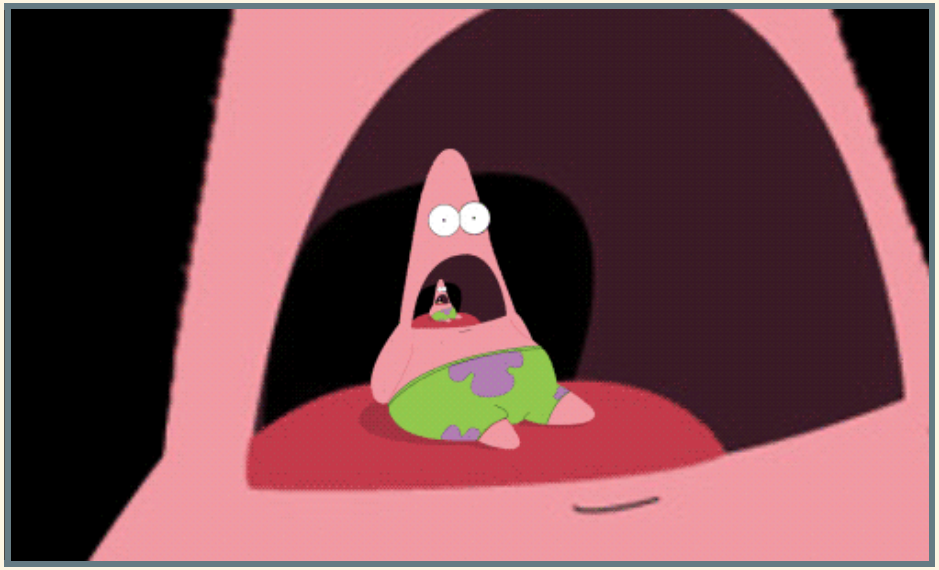
# EVENTS

```
canvas.on('object:moving', function(e) {
  var activeObj = e.target;
  console.log(activeObj.get('left'), activeObj.get('top'));
});

// attach an event to an object
rect.on("mousedown", closeTool);
```

# SUBCLASSING

```
var LabeledRect = fabric.util.createClass(fabric.Rect, {

  type: 'labeledRect',

  initialize: function(options) {
    options || (options = { });

    this.callSuper('initialize', options);
    this.set('label', options.label || '');
  },

  toObject: function() {
    return fabric.util.object.extend(this.callSuper('toObje
ct'), {
```

# OTHER AWESOMENESS



# FAST

Benchmarks
150 random objects:
Initialization: 151ms Rendering: 79ms Total time: 230ms

```
canvas.renderOnAddRemove = false
```

# TEST-DRIVEN

2400+ tests

Unit, functional, import/export

# BROWSER SUPPORT

- Firefox 2+
- Safari 3+
- Opera 9.64+
- Chrome (all versions should work)
- IE9, IE10, IE11

*If you are brave, there is some IE<9 support with excanvas*

# IN DEVELOPMENT

- Currently in development, new features all the time
- Great docs
- responsive community (Google Group)

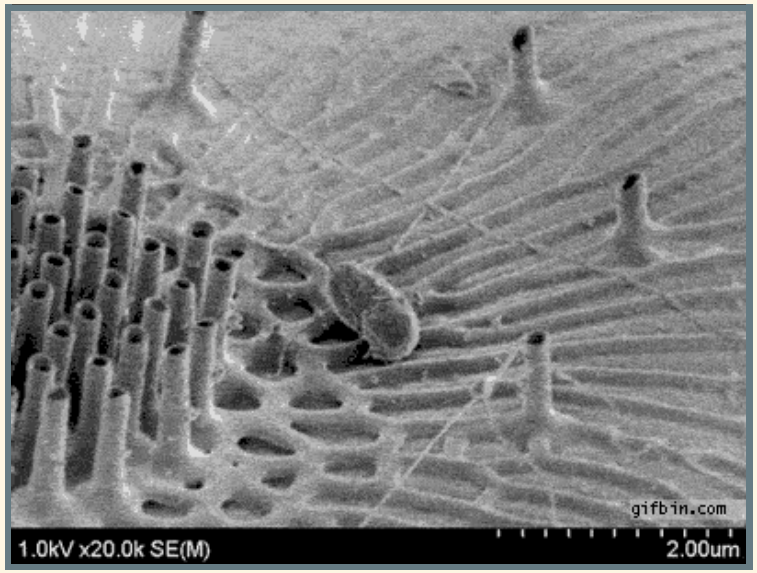
# SAVE/RESTORE 3.0

Canvas can be serialized to JSON or SVG and restored

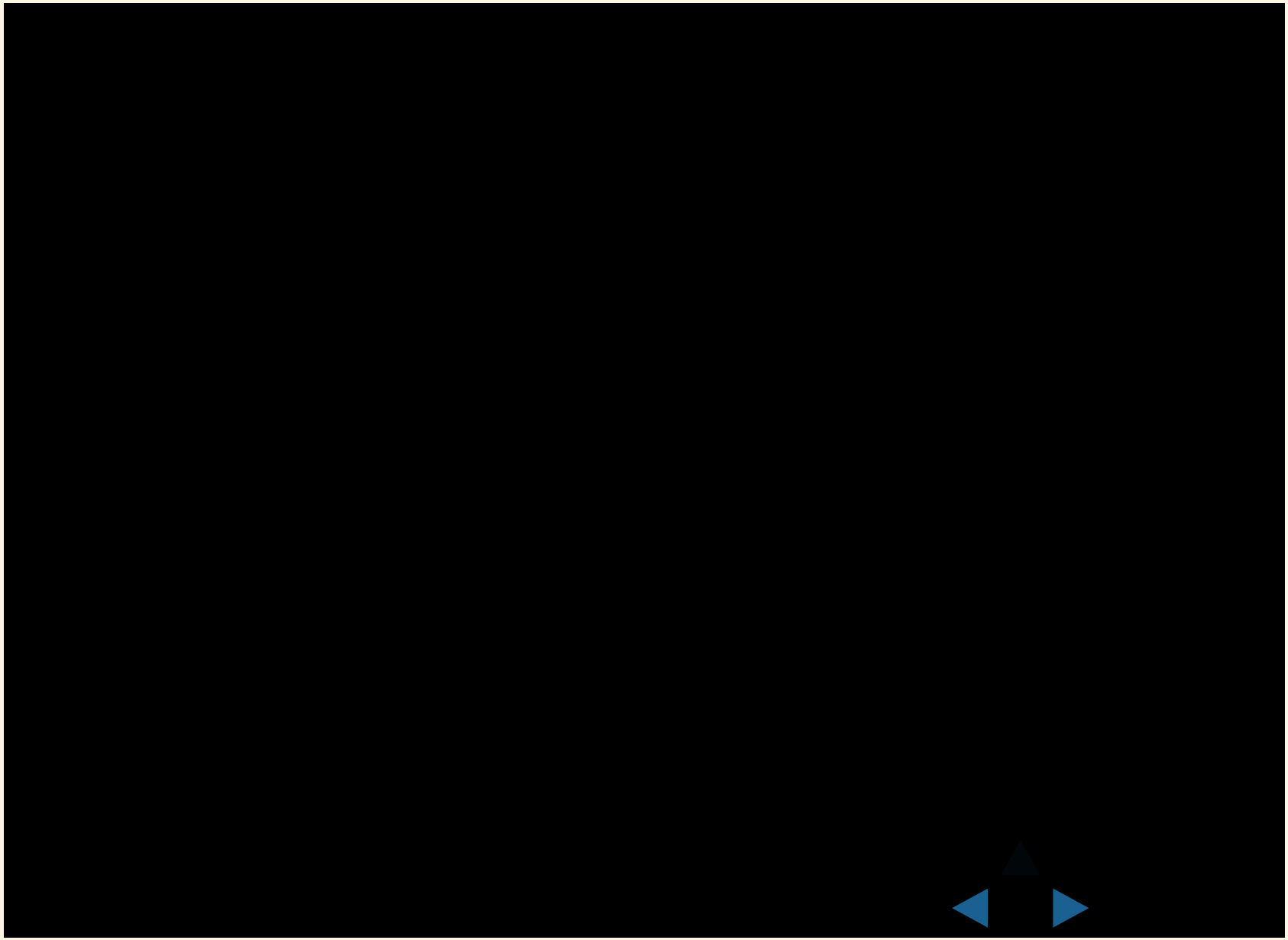
# MODULAR

## (60 SMALL CLASSES, MODULES, MIXINS)

```
node build.js modules=text,serialization,parser
```



https://www.youtube.com/watch?v=lMtjczxTSjU



# THANKS

## KELLY PACKER / @KELLYPACKER

## REFERENCES

- 4 part tutorial: http://fabricjs.com/articles/
- Docs: http://fabricjs.com/docs/
- Demos: http://fabricjs.com/demos/
- Introduction to Canvas: http://diveintohtml5.info/canvas.html
- Canvas Inspection using Chrome DevTools http://www.html5rocks.com/en/tutorials/canvas/inspection/