

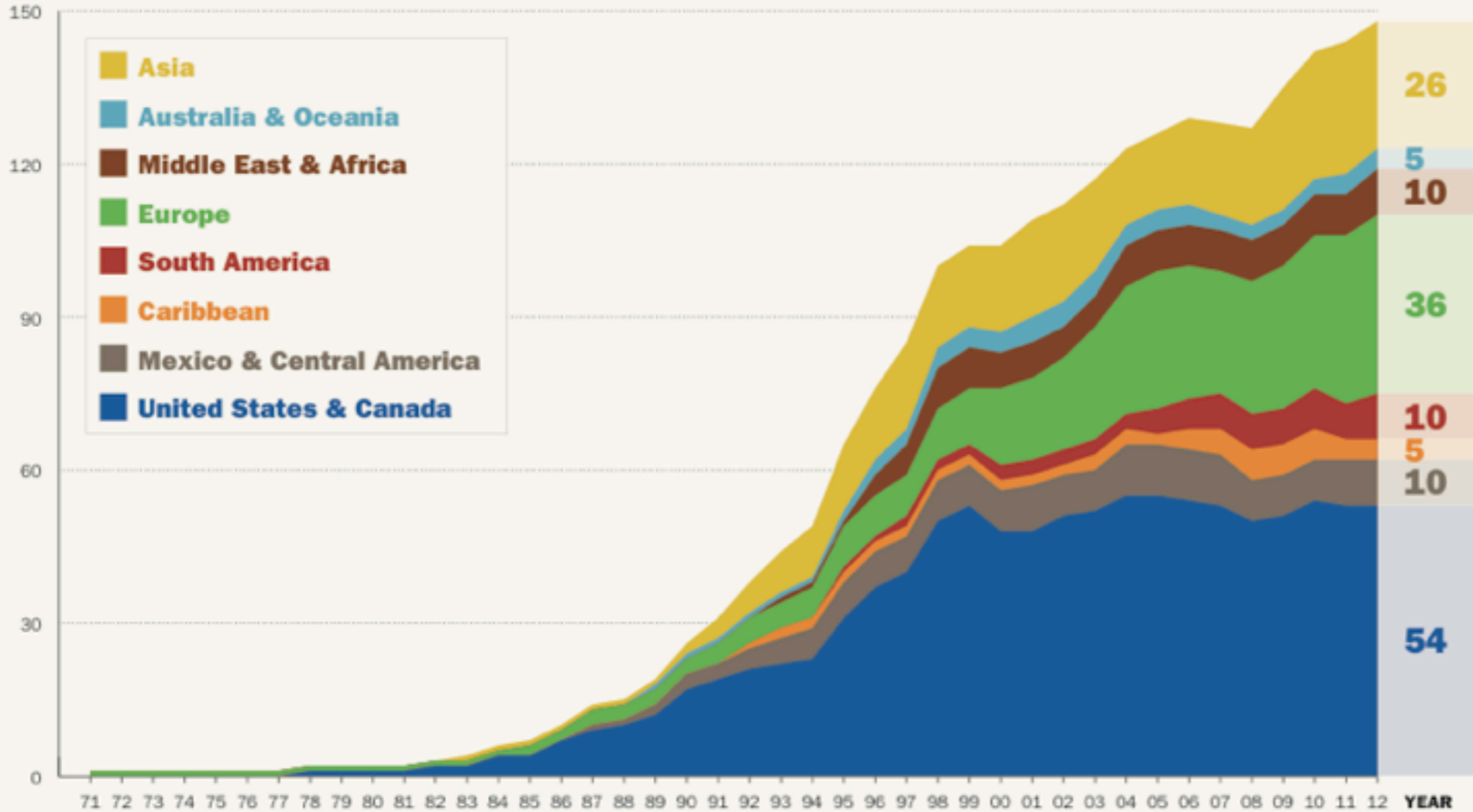
Lab 5

Raphaël.js

Assignment 2

Examples

NUMBER OF HARD ROCK CAFES



1971

The first Hard Rock Cafe opened in London, England

1982 & 1983

The first US and Asian locations opened in Los Angeles & Tokyo, respectively

1979

Hard Rock Cafe collected its first memorabilia: Eric Clapton's Red Fender Lead II guitar

1990

The first Signature Series T-shirt debuted, designed by Peter Max to benefit ECO

1985

The commemorative Hard Rock pins debuted

1999

Hard Rock's Pinktober breast cancer awareness initiative was launched

1996

The Hard Rock Cafes & Hotels were consolidated under one company

2007

Hard Rock Cafe International, Inc. was acquired by the Seminole Tribe of Florida

2011

40th Anniversary!



148
Locations

55
Countries

13
New locations planned

74,000+
Pieces of rock & roll memorabilia collected

77 mill.
Hard Rock Cafe visitors per year

London
Location of the first and oldest Hard Rock Cafe

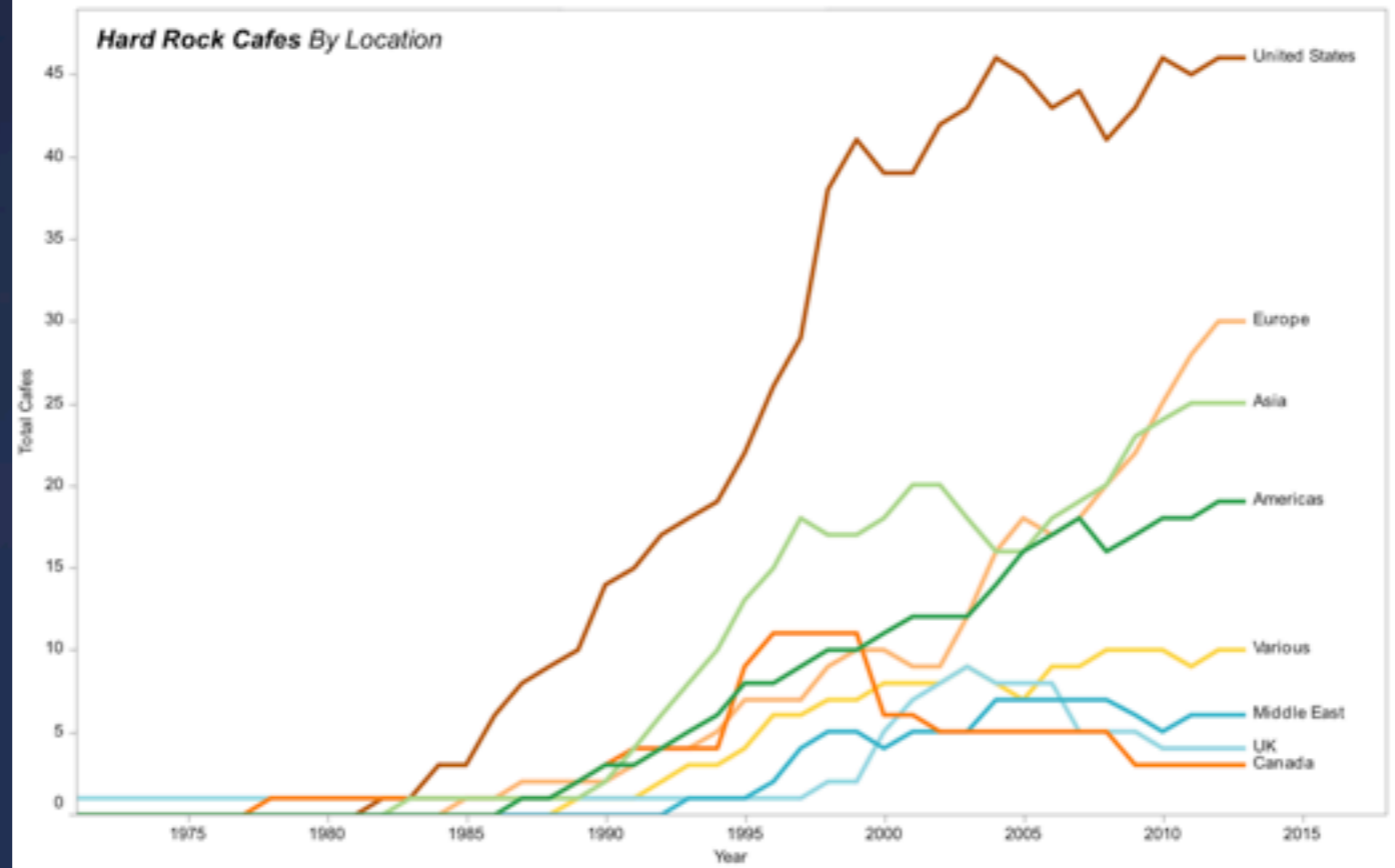
Orlando
Location of the Hard Rock headquarters and the largest Hard Rock Cafe

The Growth of Hard Rock Cafe (1971–2012)

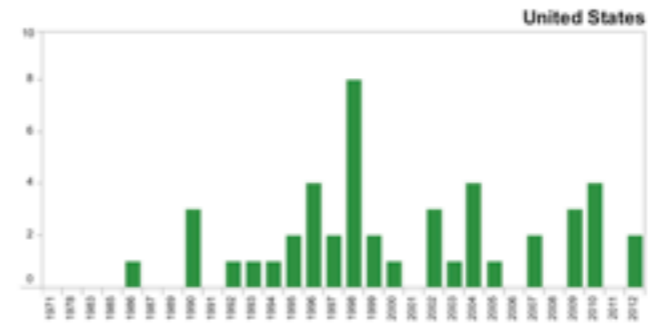
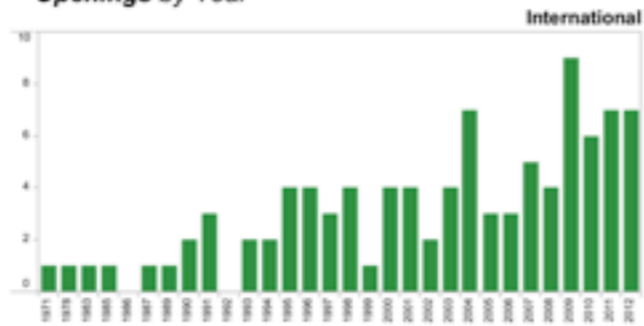
The chart above shows the growth of Hard Rock Cafe around the world over time, broken down by geographic regions. The data only takes Hard Rock Cafes into account—other Hard Rock venues (such as Hard Rock Live and Hard Rock Hotel & Casino) are excluded.

INFO 247 :: Assignment #2 :: Raymon Sutedjo-The

SOURCES
<http://www.hardrock.com/corporate/history.aspx>
<http://www.hardrock.com/locations.aspx>
http://en.wikipedia.org/wiki/Hard_Rock_Cafe

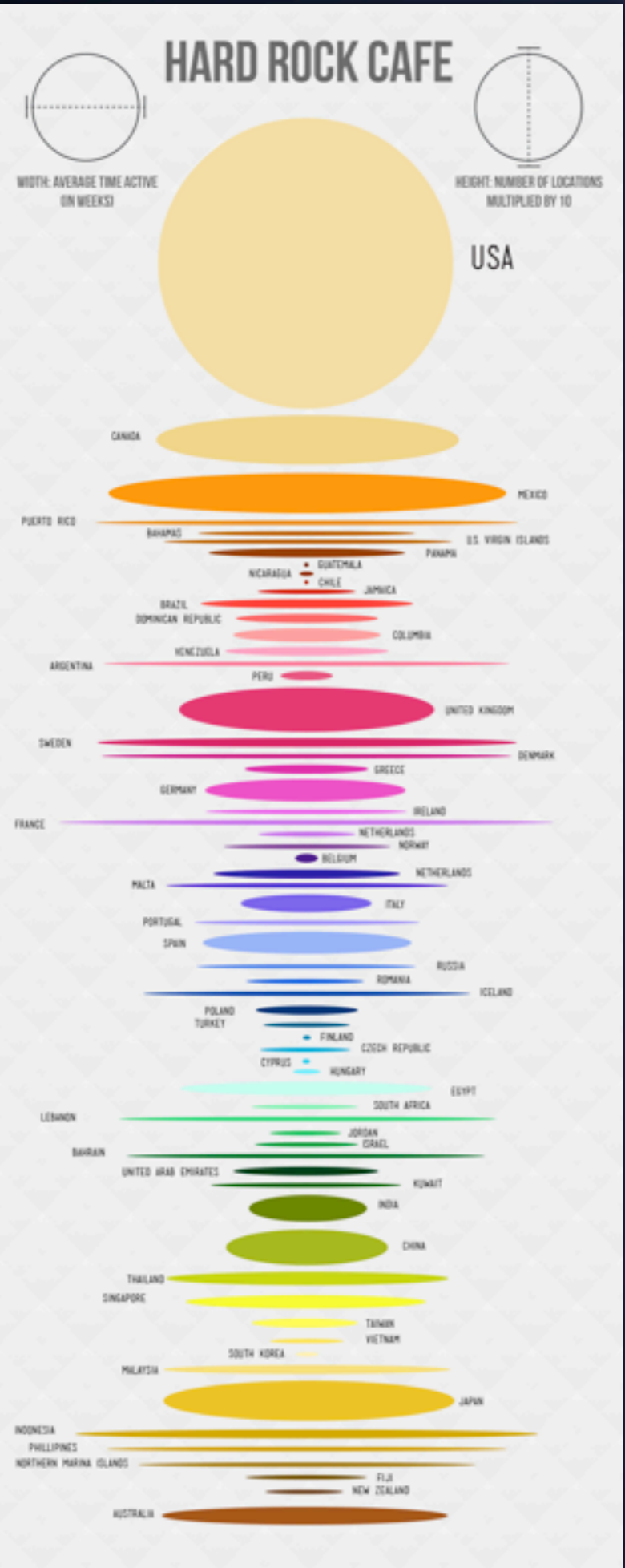


Openings by Year



Closings by Year





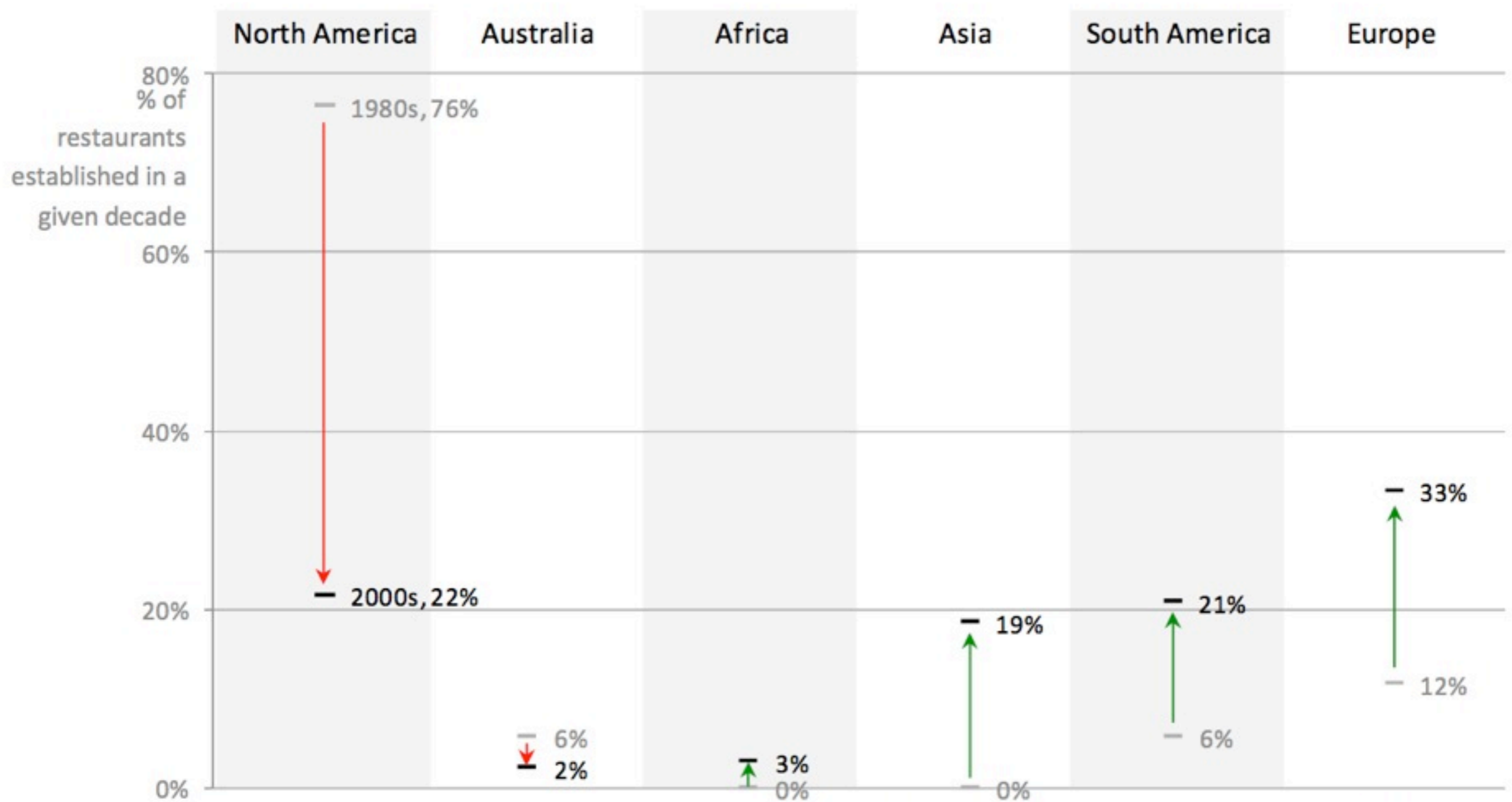


Figure 1. Change over time in proportion of all new Hard Rock Café establishments by geographic region.

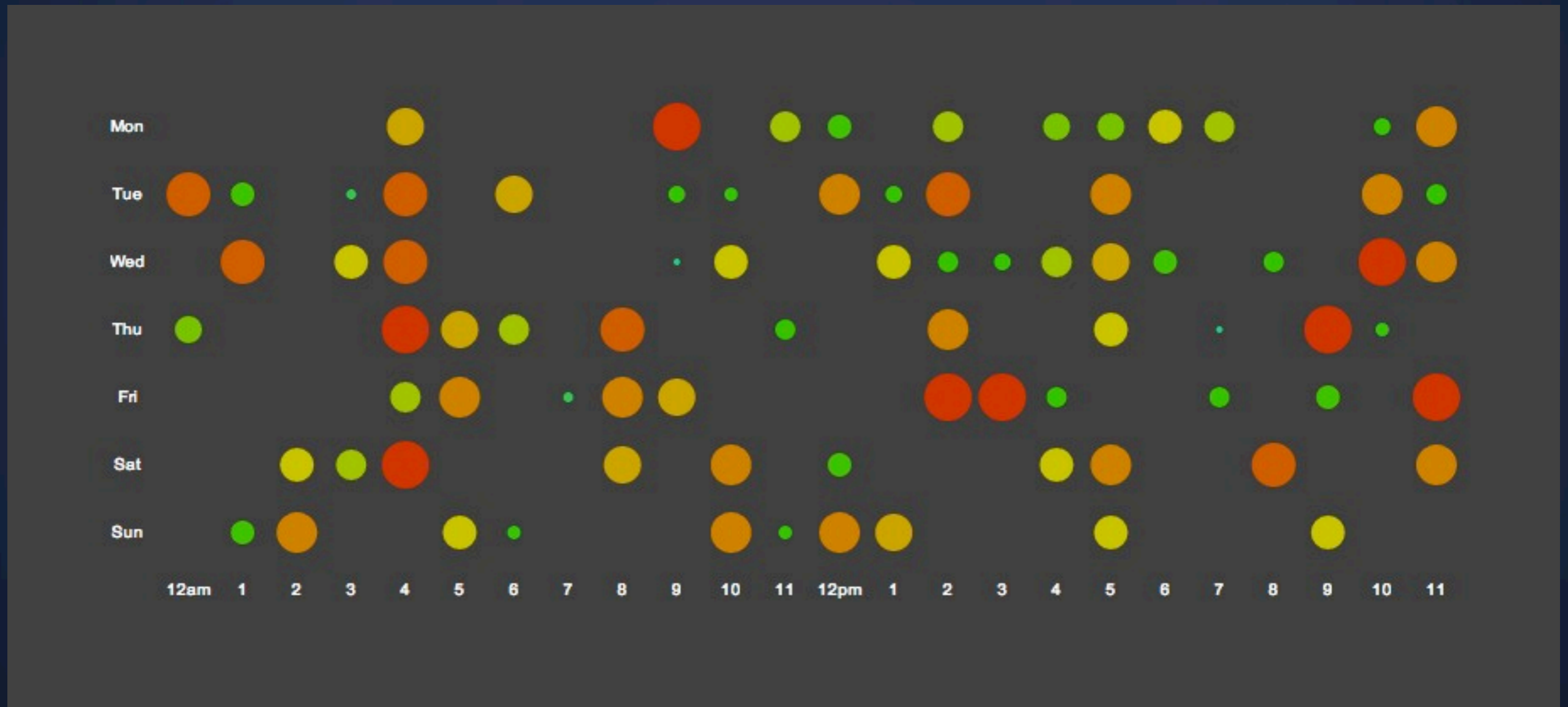
Raphaël.js



<http://raphaeljs.com/>

Raphaël.js

What it's used for



Raphaël.js

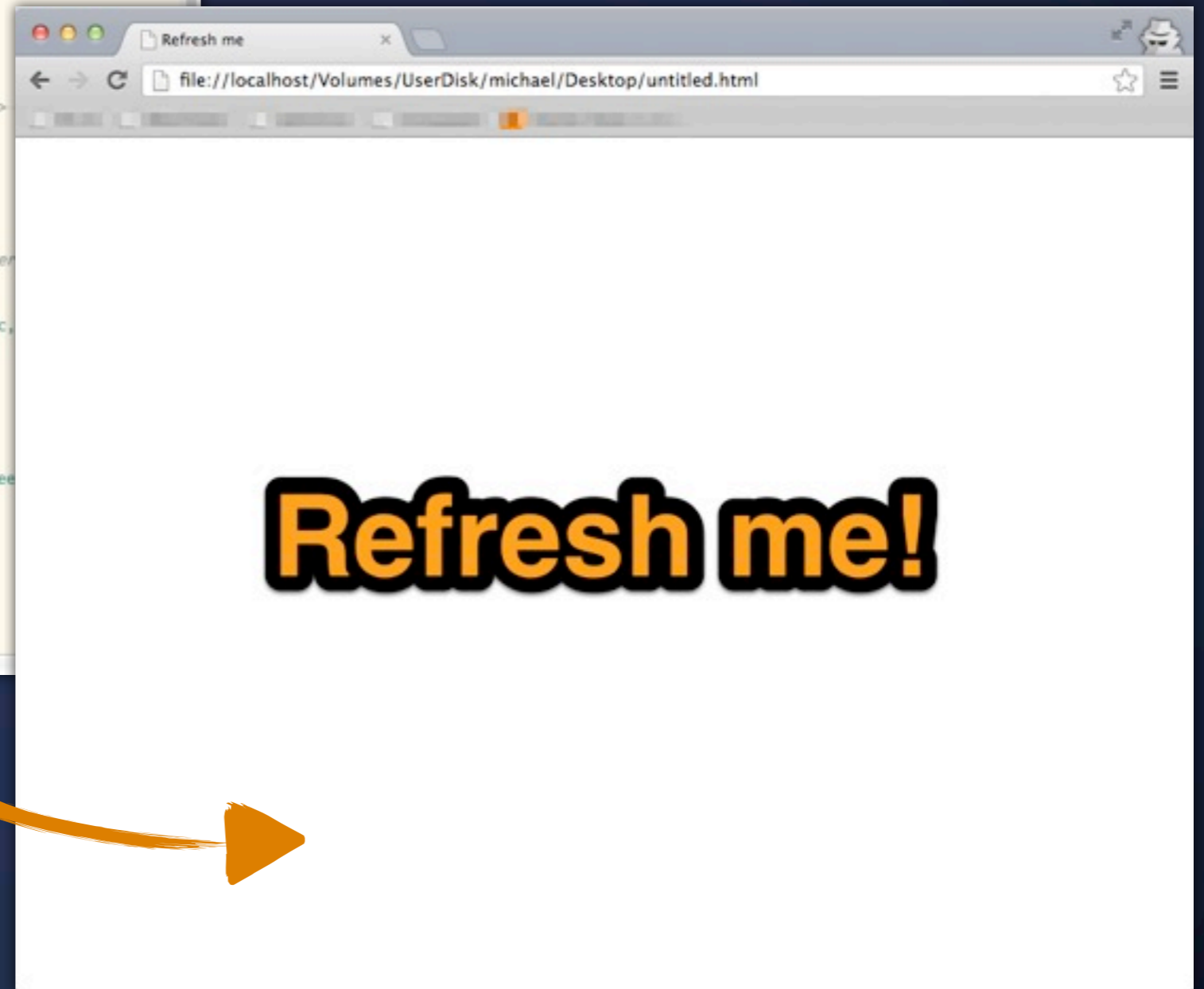
Reference / Examples / Documentation

<http://raphaeljs.com/reference.html>

Traditional Development

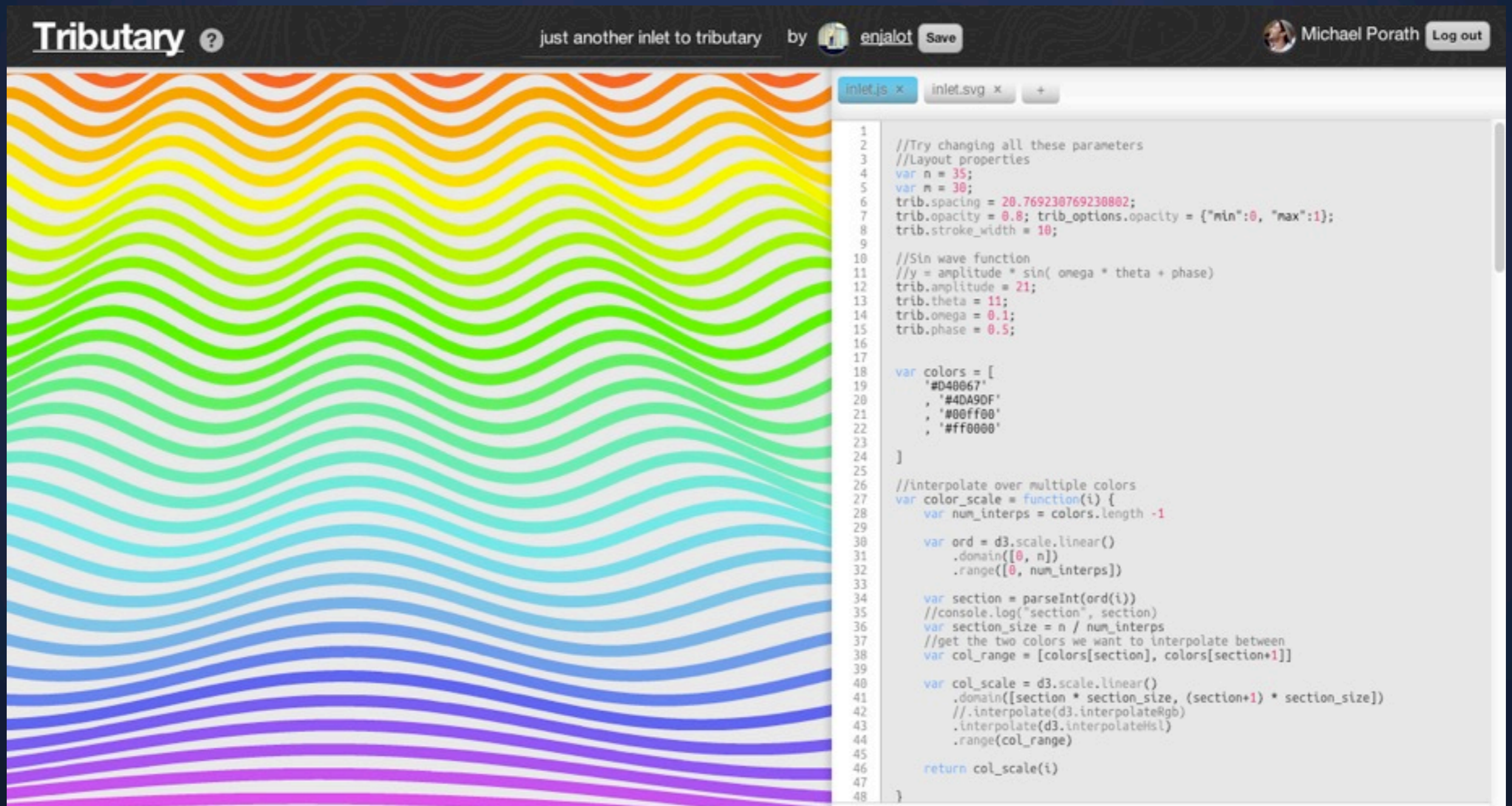
Text Editor and Browser

```
untitled
1 <html lang="en"><head>
2   <meta charset="utf-8">
3   <title>Tributary | just another inlet to tributary</title>
4   <!-- Place favicon.ico and apple-touch-icon.png in the root of your domain and delete these
5   references
6   -->
7   <link rel="icon" type="image/png" href="/static/img/favicon.32.png">
8   <link rel="shortcut icon" href="/static/img/favicon.ico">
9
10  <!--[if lt IE 9]>
11    <script src="http://html5shim.googlecode.com/svn/trunk/html5.js"></script>
12  <![endif]>
13  <link rel="stylesheet" media="all" href="">
14  <meta name="viewport" content="width=device-width, initial-scale=1">
15  <!-- Adding "maximum-scale=1" fixes the Mobile Safari auto-zoom bug:
16  http://filamentgroup.com/examples/iosScaleBug/ -->
17
18  <!-- TODO: separate out tributary ui styling from header styling? some of it over
19  <link rel="stylesheet" href="/static/css/trib.css">
20  <link rel="stylesheet" href="/static/css/animation.css">
21  <link href="http://fonts.googleapis.com/css?family=Ubuntu+Mono:400,700,400italic,
22  rel="stylesheet" type="text/css">
23  <link rel="stylesheet" href="/static/css/tipsy.css">
24  <link rel="stylesheet" type="text/css"
25  href="http://yui.yahooapis.com/2.9.0/build/reset/reset-min.css">
26
27  <!-- And the main styles -->
28  <link rel="stylesheet" href="/static/css/header.css" type="text/css" media="screen"
29  title="Primary Stylesheet" charset="utf-8">
30
31  <!-- Add jQuery -->
32  <script type="text/javascript" async=""
33  src="http://www.google-analytics.com/ga.js"></script><script></script>
34  </head></html>
```



Tributary

In-Browser Editor



The screenshot displays the Tributary In-Browser Editor interface. The top navigation bar includes the "Tributary" logo, a help icon, the text "just another inlet to tributary by enjalot", a "Save" button, and a user profile for "Michael Porath" with a "Log out" button. The main area is split into two panels. The left panel shows a vibrant, wavy pattern of lines in shades of orange, yellow, green, cyan, blue, and purple. The right panel is a code editor with two tabs: "inlet.js" and "inlet.svg". The "inlet.js" tab is active, showing the following JavaScript code:

```
1
2 //Try changing all these parameters
3 //Layout properties
4 var n = 35;
5 var m = 30;
6 trib.spacing = 20.769230769230802;
7 trib.opacity = 0.8; trib_options.opacity = {"min":0, "max":1};
8 trib.stroke_width = 10;
9
10 //Sin wave function
11 //y = amplitude * sin( omega * theta + phase)
12 trib.amplitude = 21;
13 trib.theta = 11;
14 trib.omega = 0.1;
15 trib.phase = 0.5;
16
17
18 var colors = [
19   '#D40067'
20   , '#4DA9DF'
21   , '#00ff00'
22   , '#ff0000'
23 ]
24
25
26 //interpolate over multiple colors
27 var color_scale = function(i) {
28   var num_interps = colors.length - 1
29
30   var ord = d3.scale.linear()
31     .domain([0, n])
32     .range([0, num_interps])
33
34   var section = parseInt(ord(i))
35   //console.log("section", section)
36   var section_size = n / num_interps
37   //get the two colors we want to interpolate between
38   var col_range = [colors[section], colors[section+1]]
39
40   var col_scale = d3.scale.linear()
41     .domain([section * section_size, (section+1) * section_size])
42     //.interpolate(d3.interpolateRgb)
43     .interpolate(d3.interpolateHsl)
44     .range(col_range)
45
46   return col_scale(i)
47
48 }
```

<http://tributary.io>

Tributary

In-Browser Editor

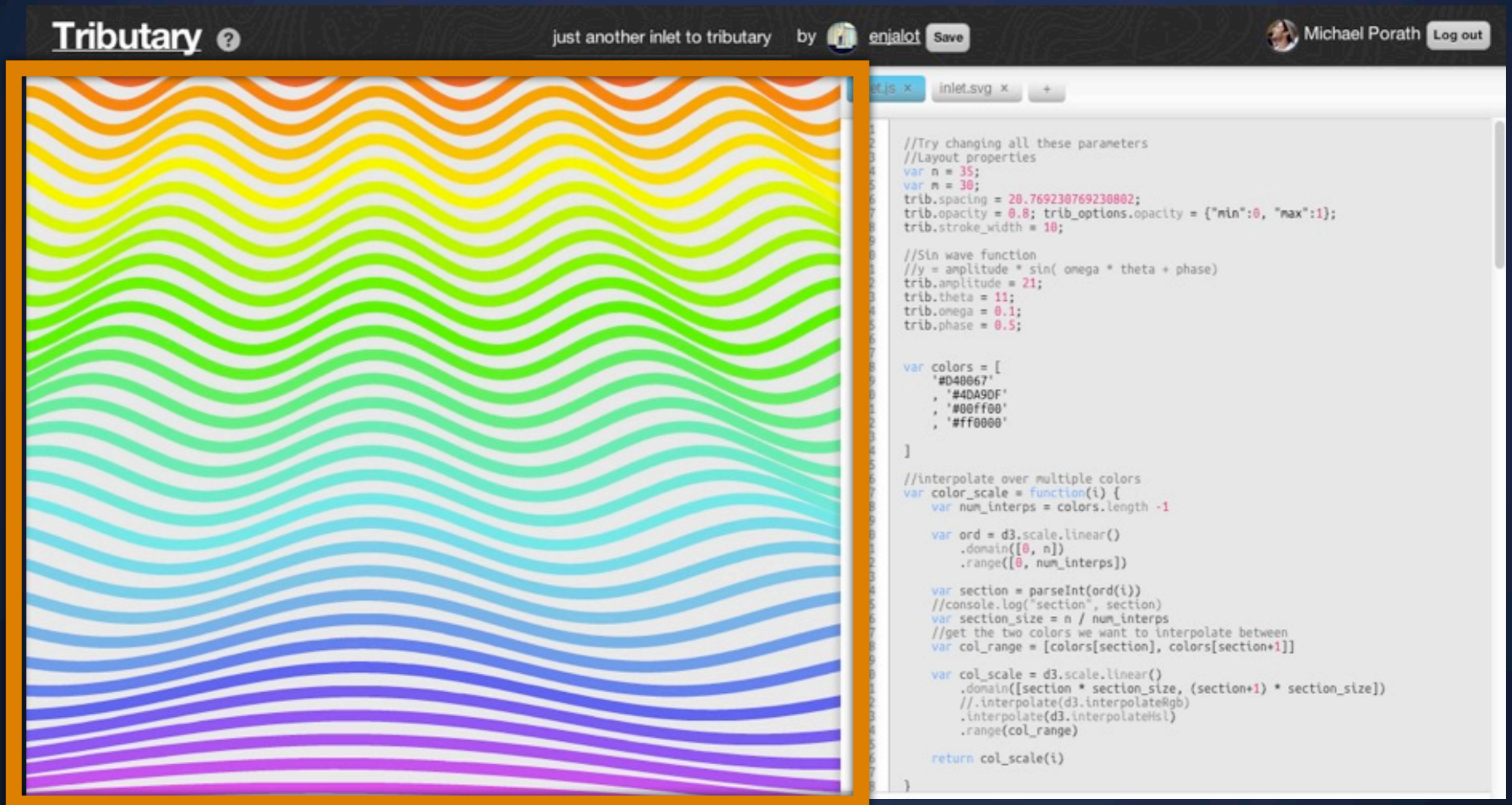
inlet.js x inlet.svg x +

```
1
2 //Try changing all these parameters
3 //Layout properties
4 var n = 35;
5 var m = 30;
6 trib.spacing = 20.769230769230802;
7 trib.opacity = 0.8; trib_options.opacity = {"min":0, "max":1};
8 trib.stroke_width = 10;
9
10 //Sin wave function
11 //y = amplitude * sin( omega * theta + phase)
12 trib.amplitude = 21;
13 trib.theta = 11;
14 trib.omega = 0.1;
15 trib.phase = 0.5;
16
17
18 var colors = [
19   '#D40067'
20   , '#4DA9DF'
21   , '#00ff00'
22   , '#ff0000'
23 ]
24
25
26 //interpolate over multiple colors
27 var color_scale = function(i) {
28   var num_interps = colors.length - 1
29
30   var ord = d3.scale.linear()
31     .domain([0, n])
32     .range([0, num_interps])
33
34   var section = parseInt(ord(i))
35   //console.log("section", section)
36   var section_size = n / num_interps
37   //get the two colors we want to interpolate between
38   var col_range = [colors[section], colors[section+1]]
39
40   var col_scale = d3.scale.linear()
41     .domain([section * section_size, (section+1) * section_size])
42     //.interpolate(d3.interpolateRgb)
43     .interpolate(d3.interpolateHsl)
44     .range(col_range)
45
46   return col_scale(i)
47
48 }
```

<http://tributary.io>

Tributary

In-Browser Editor



The image shows a screenshot of the Tributary in-browser editor. The top navigation bar includes the "Tributary" logo, a help icon, the text "just another inlet to tributary by enjalot", a "Save" button, a user profile for "Michael Porath", and a "Log out" button. The main content area is split into two panes. The left pane, highlighted with an orange border, displays a vibrant, multi-colored wavy pattern that transitions from orange at the top to purple at the bottom. The right pane shows the JavaScript code that generates this pattern. The code includes comments and variables for layout properties, a sine wave function, a color array, and a function for interpolating colors.

```
//Try changing all these parameters
//Layout properties
var n = 35;
var m = 30;
trib.spacing = 20.769230769230802;
trib.opacity = 0.8; trib_options.opacity = {"min":0, "max":1};
trib.stroke_width = 10;

//Sin wave function
//y = amplitude * sin( omega * theta + phase)
trib.amplitude = 21;
trib.theta = 11;
trib.omega = 0.1;
trib.phase = 0.5;

var colors = [
  '#D40067',
  '#4DA9DF',
  '#00FF00',
  '#FF0000'
]

//interpolate over multiple colors
var color_scale = function(i) {
  var num_interps = colors.length - 1

  var ord = d3.scale.linear()
    .domain([0, n])
    .range([0, num_interps])

  var section = parseInt(ord(i))
  //console.log("section", section)
  var section_size = n / num_interps
  //get the two colors we want to interpolate between
  var col_range = [colors[section], colors[section+1]]

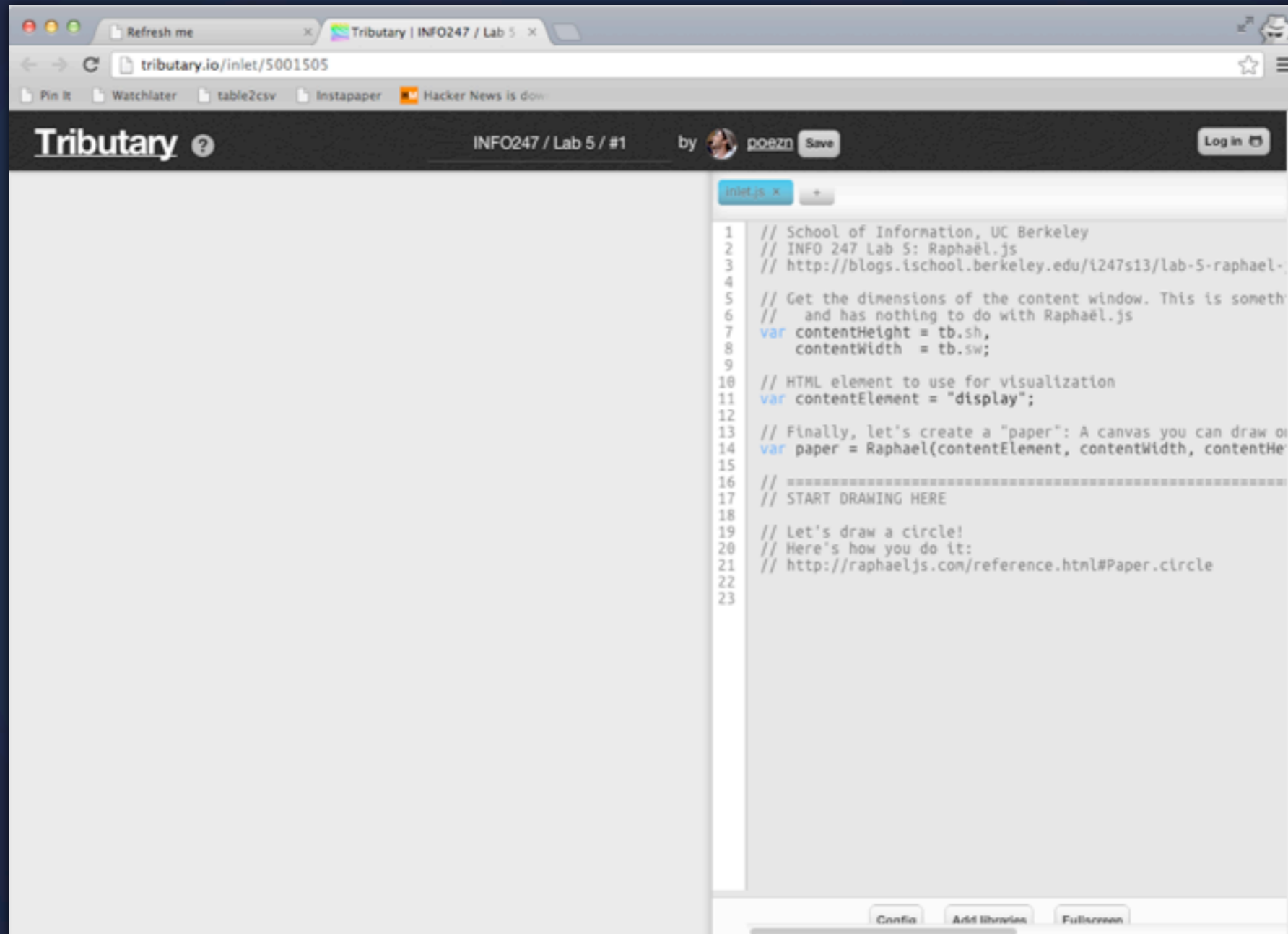
  var col_scale = d3.scale.linear()
    .domain([section * section_size, (section+1) * section_size])
    //.interpolate(d3.interpolateRgb)
    .interpolate(d3.interpolateHsl)
    .range(col_range)

  return col_scale(i)
}
```

<http://tributary.io>

Your Turn - Exercise 1

Blank Slate



The screenshot shows a web browser window with the URL `tributary.io/inlet/5001505`. The page title is "Tributary" and the content is "INFO247 / Lab 5 / #1" by user "p0020". The main area is a large, empty white space. On the right side, there is a code editor window titled "inlet.js" containing the following JavaScript code:

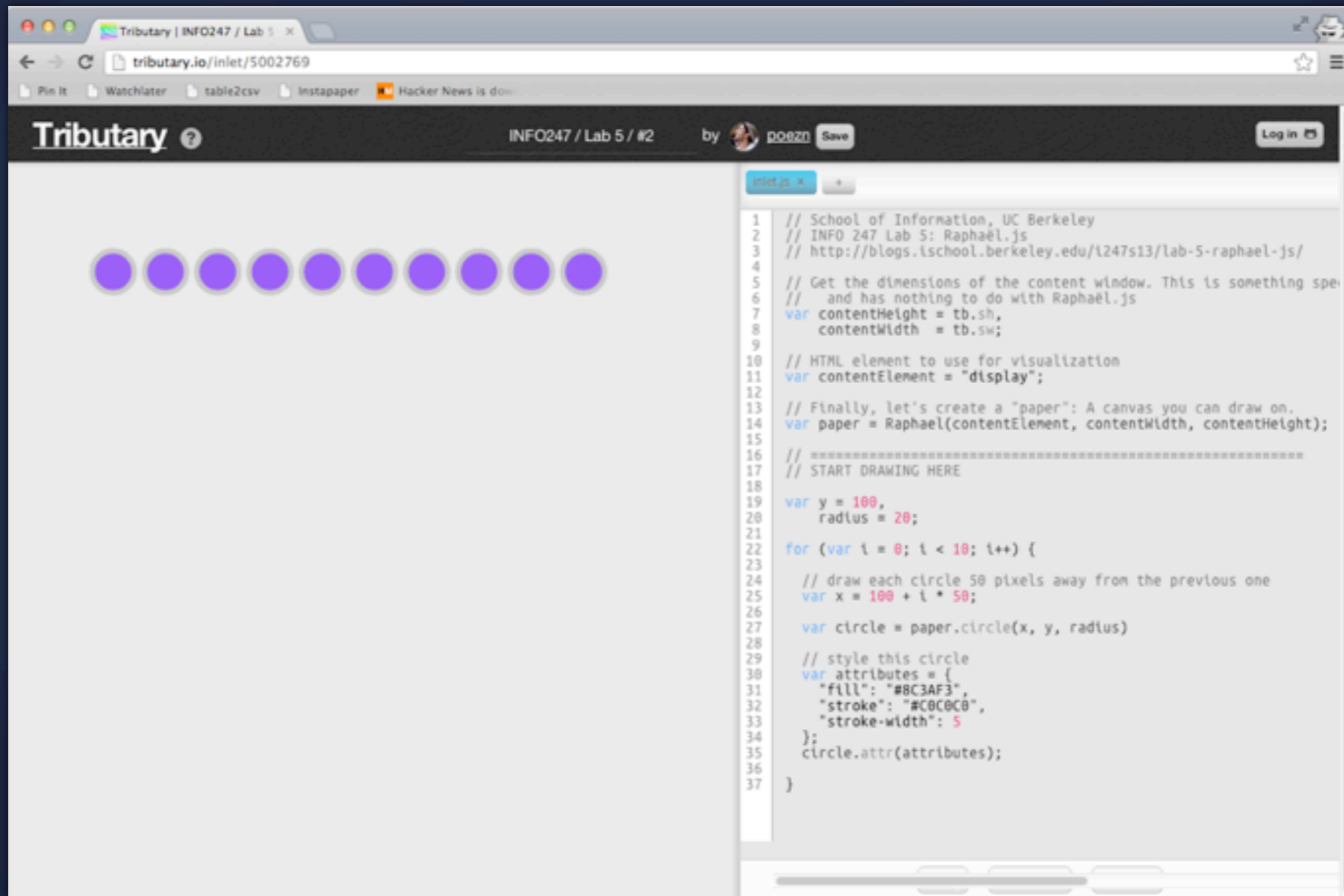
```
1 // School of Information, UC Berkeley
2 // INFO 247 Lab 5: Raphaël.js
3 // http://blogs.ischool.berkeley.edu/t247s13/lab-5-raphael-
4
5 // Get the dimensions of the content window. This is someth
6 // and has nothing to do with Raphaël.js
7 var contentHeight = tb.sh,
8     contentWidth = tb.sw;
9
10 // HTML element to use for visualization
11 var contentElement = "display";
12
13 // Finally, let's create a "paper": A canvas you can draw on
14 var paper = Raphael(contentElement, contentWidth, contentHe
15
16 // =====
17 // START DRAWING HERE
18
19 // Let's draw a circle!
20 // Here's how you do it:
21 // http://raphaeljs.com/reference.html#Paper.circle
22
23
```

At the bottom of the code editor, there are buttons for "Config", "Add libraries", and "Fullscreen".

<http://tributary.io/inlet/5001505>

Exercise 2

Data driven properties



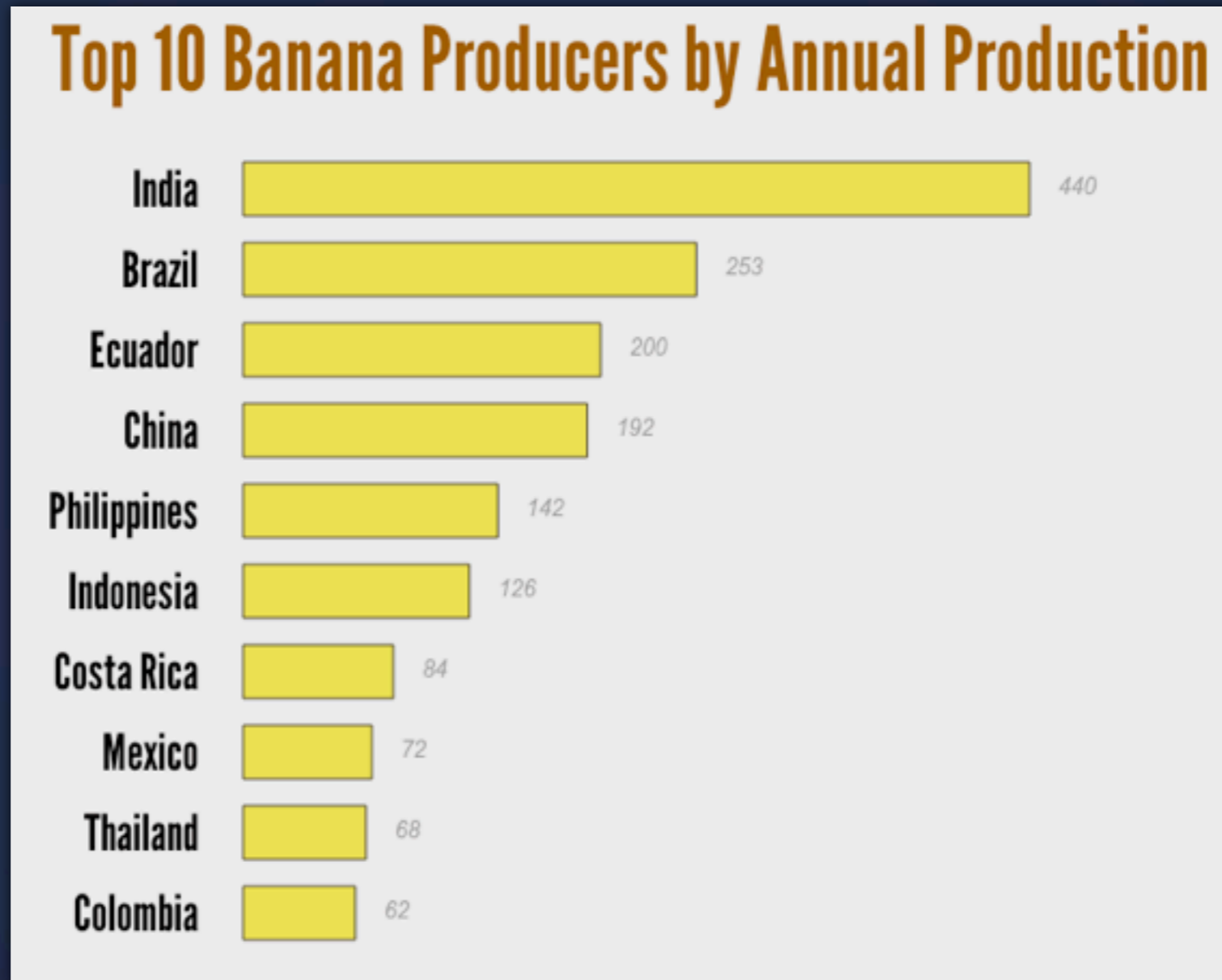
The screenshot shows a web browser window displaying a Tributary application. The browser's address bar shows the URL `tributary.io/inlet/5002769`. The page header includes the Tributary logo, the title "INFO247 / Lab 5 / #2", the author "by po00zi", and a "Save" button. The main content area features a horizontal row of 10 purple circles. To the right of the circles is a code editor with the following JavaScript code:

```
1 // School of Information, UC Berkeley
2 // INFO 247 Lab 5: Raphaël.js
3 // http://blogs.ischool.berkeley.edu/l247s13/lab-5-raphael-js/
4
5 // Get the dimensions of the content window. This is something spe
6 // and has nothing to do with Raphaël.js
7 var contentHeight = tb.sh,
8     contentWidth = tb.sw;
9
10 // HTML element to use for visualization
11 var contentElement = "display";
12
13 // Finally, let's create a "paper": A canvas you can draw on.
14 var paper = Raphael(contentElement, contentWidth, contentHeight);
15
16 // =====
17 // START DRAWING HERE
18
19 var y = 100,
20     radius = 20;
21
22 for (var i = 0; i < 10; i++) {
23
24     // draw each circle 50 pixels away from the previous one
25     var x = 100 + i * 50;
26
27     var circle = paper.circle(x, y, radius)
28
29     // style this circle
30     var attributes = {
31         "fill": "#BC3AF3",
32         "stroke": "#CBC0CB",
33         "stroke-width": 5
34     };
35     circle.attr(attributes);
36
37 }
```

<http://tributary.io/inlet/5002769>

Exercise 3

Bar Chart



<http://tributary.io/inlet/5002821>

Exercise 3

Bar Chart: Full Code

The screenshot shows a web browser window displaying a bar chart titled "Top 10 Banana Producers by Annual Production". The chart lists the following countries and their production in tonnes:

Country	Annual Production (tonnes)
India	440
Brazil	253
Ecuador	200
China	192
Philippines	142
Indonesia	126
Costa Rica	84
Mexico	72
Thailand	68
Colombia	62

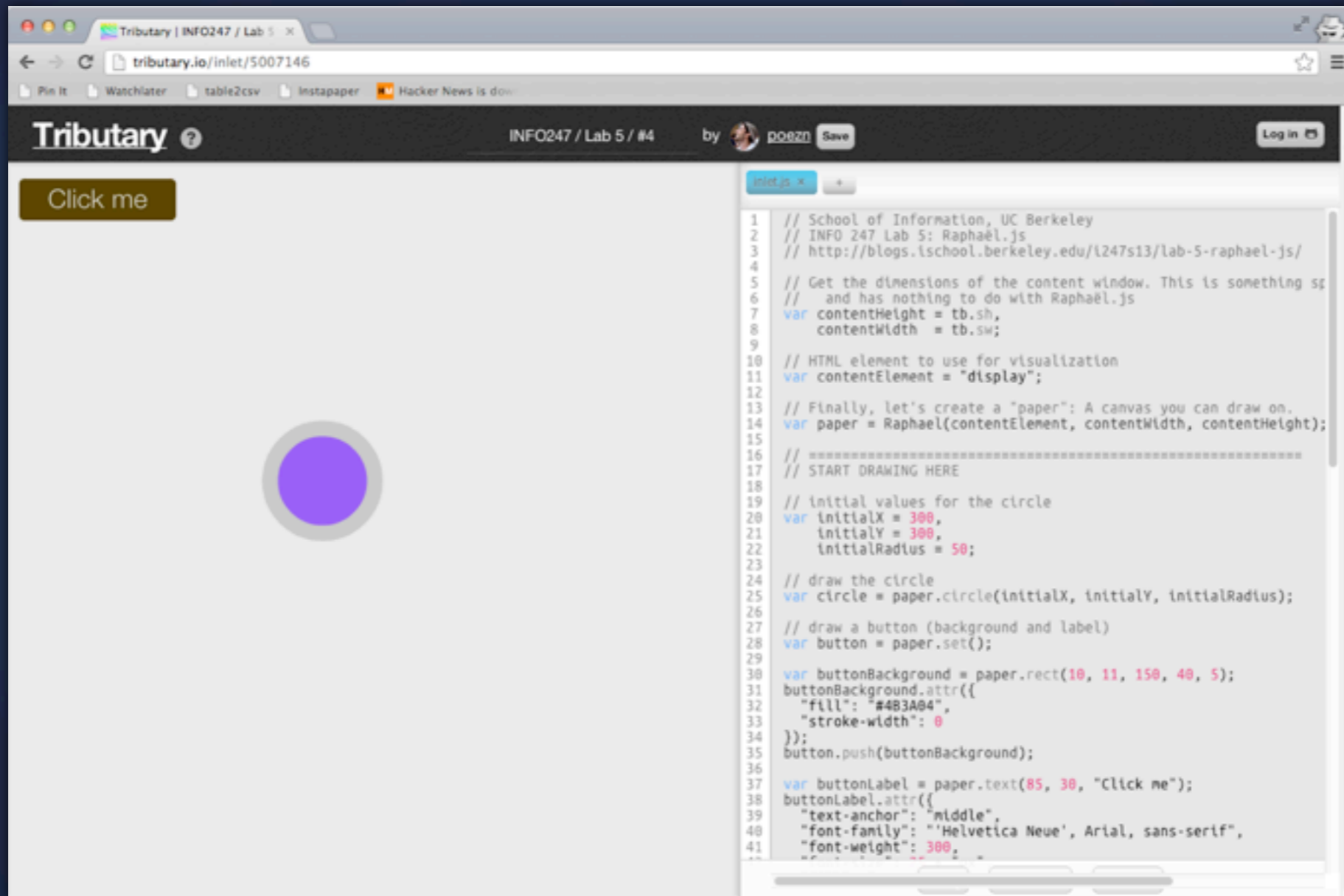
Below the chart, the JavaScript code for the visualization is shown in a code editor. The code includes comments and logic for sorting the data and drawing the bars.

```
1 // School of Information, UC Berkeley
2 // INFO 247 Lab 5: Raphaël.js
3 // http://blogs.ischool.berkeley.edu/l247s13/lab-5-raphael-js/
4
5 // Get the dimensions of the content window. This is something sp
6 // and has nothing to do with Raphaël.js
7 var contentHeight = tb.sh,
8     contentWidth = tb.sw;
9
10 // HTML element to use for visualization
11 var contentElement = "display";
12
13 // Finally, let's create a "paper": A canvas you can draw on.
14 var paper = Raphael(contentElement, contentWidth, contentHeight);
15
16 // Import the data. This is again something that is
17 // specific to Tributary. you can import any CSV or JSON
18 // file (shown in the yellow tab above)
19 var data = tb['data'];
20
21 // =====
22 // HELPER FUNCTIONS
23
24 // sort by production (descending)
25 var sortByProduction = function(a,b) {
26     return b.tonnes - a.tonnes;
27 };
28
29 data.sort(sortByProduction)
30
31 // =====
32 // START DRAWING HERE
33
34 for (var i = 0; i < 10; i++) {
35     var currentRecord = data[i];
36     var tonnes = currentRecord['tonnes'];
37     var width = tonnes / 25000;
38
39     // set the (base) vertical position for bars and labels
40     var y = 90 + (i * 45);
41
```

<http://tributary.io/inlet/5002933>

Exercise 4

Animations



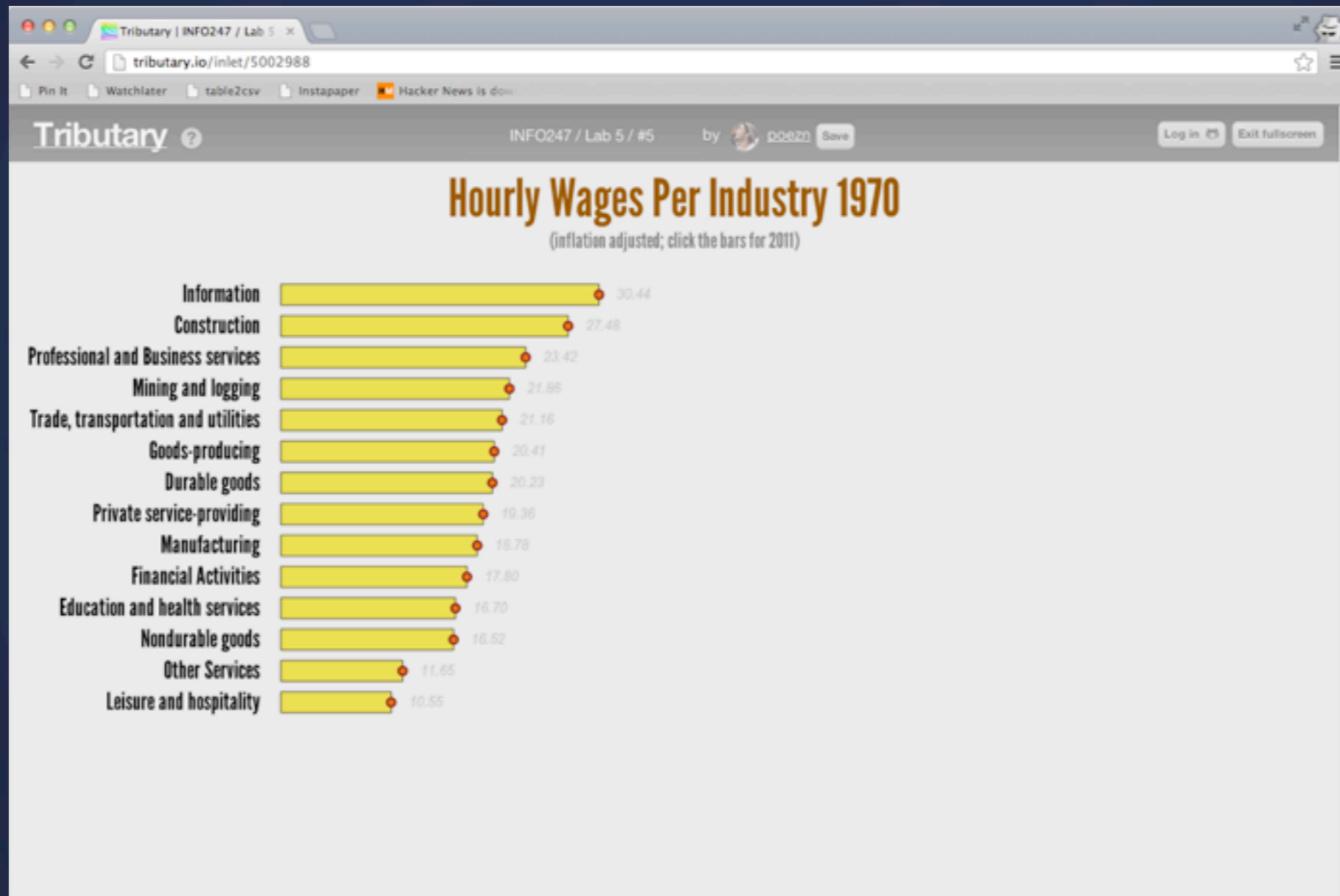
The screenshot shows a web browser window displaying a Tributary page. The page title is "Tributary | INFO247 / Lab 5" and the URL is "tributary.io/inlet/5007146". The page content includes a "Click me" button and a purple circle. The right side of the browser shows the JavaScript code for the animation.

```
1 // School of Information, UC Berkeley
2 // INFO 247 Lab 5: Raphaël.js
3 // http://blogs.ischool.berkeley.edu/l247s13/lab-5-raphael-js/
4
5 // Get the dimensions of the content window. This is something sp
6 // and has nothing to do with Raphaël.js
7 var contentHeight = tb.sh,
8     contentWidth = tb.sw;
9
10 // HTML element to use for visualization
11 var contentElement = "display";
12
13 // Finally, let's create a "paper": A canvas you can draw on.
14 var paper = Raphael(contentElement, contentWidth, contentHeight);
15
16 // =====
17 // START DRAWING HERE
18
19 // Initial values for the circle
20 var initialX = 300,
21     initialY = 300,
22     initialRadius = 50;
23
24 // draw the circle
25 var circle = paper.circle(initialX, initialY, initialRadius);
26
27 // draw a button (background and label)
28 var button = paper.set();
29
30 var buttonBackground = paper.rect(10, 11, 150, 40, 5);
31 buttonBackground.attr({
32   "fill": "#483A04",
33   "stroke-width": 0
34 });
35 button.push(buttonBackground);
36
37 var buttonLabel = paper.text(85, 30, "Click me");
38 buttonLabel.attr({
39   "text-anchor": "middle",
40   "font-family": "'Helvetica Neue', Arial, sans-serif",
41   "font-weight": 300,
```

<http://tributary.io/inlet/5007146>

Exercise 5

Bar chart with Animations



<http://tributary.io/inlet/5002988>

Tributary

Google Groups

<https://groups.google.com/forum/#!forum/tributary>

Next Lecture

Color