

Visualization on the Web

SVG **SCALABLE VECTOR GRAPHICS**

Introducing SVG

- Descriptive tags for images
- Based on vector graphics
- D3.js can manage the creation and modification of tags

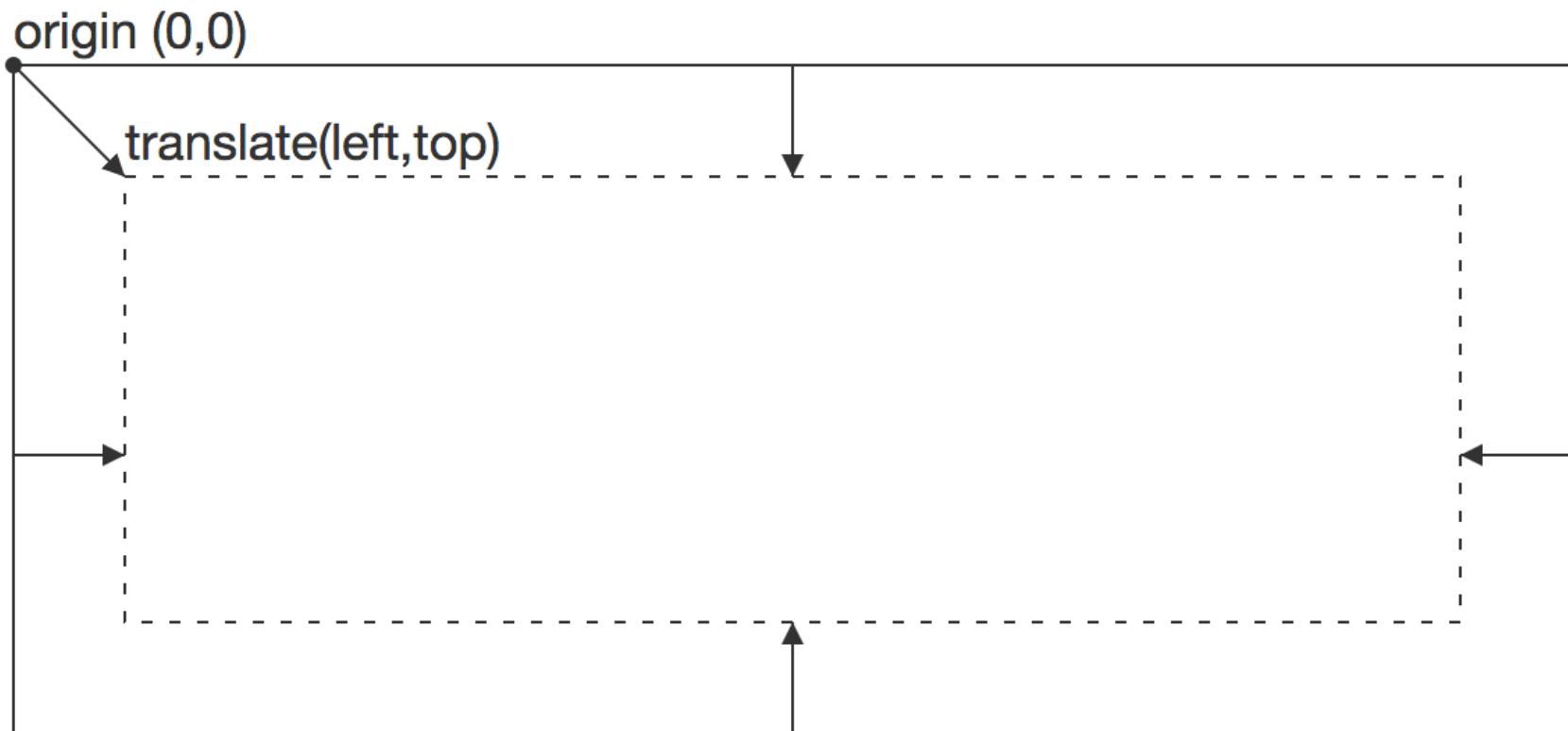
SVG External Resources

- SVG Specification
 - <http://www.w3.org/TR/SVG/>
- Mozilla Developer Network
 - <https://developer.mozilla.org/en/SVG>
- D3.js API Reference
 - <https://github.com/mbostock/d3/wiki/SVG-Shapes>

Hello World Example

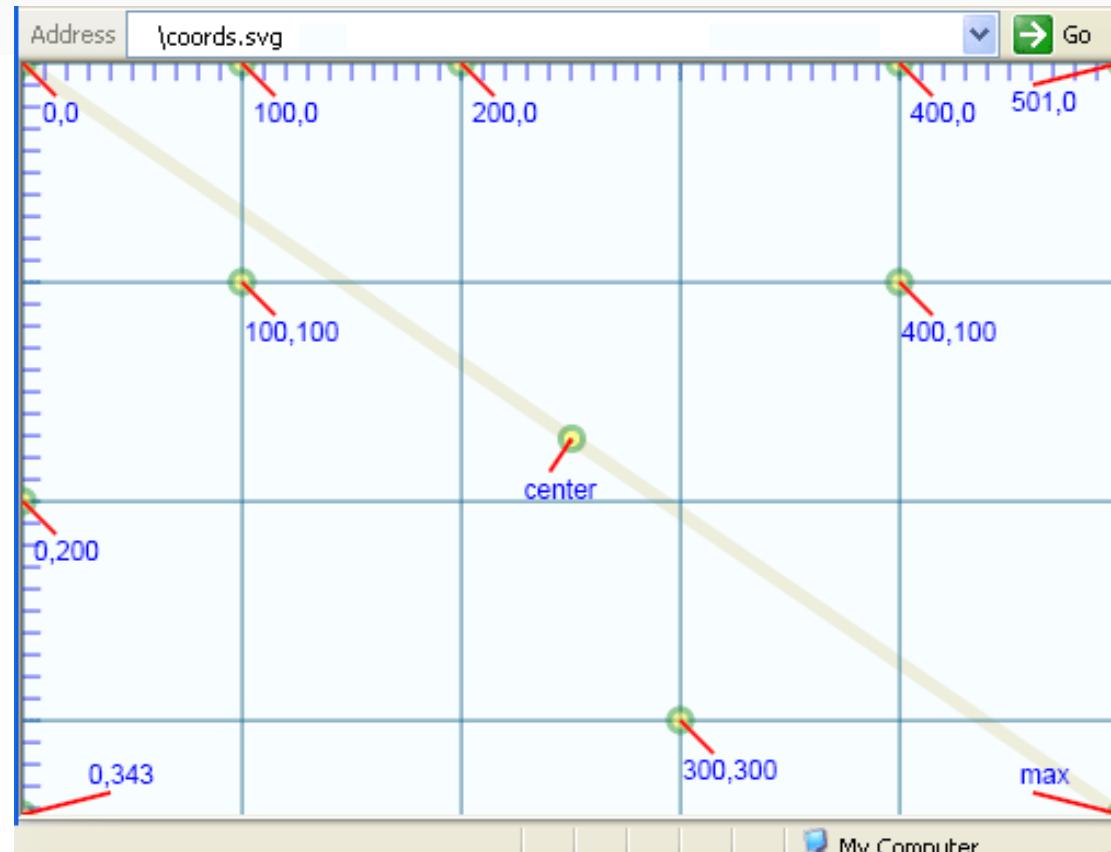
```
<!DOCTYPE html>
<meta charset="utf-8">
<svg width="960" height="500">
  <text y="12">
    Hello, world!
  </text>
</svg>
```

Coordinate System



SVG Viewport

```
<svg width="500" height="300">
  <circle cx="250" cy="150" r="30" fill="red" stroke="black" stroke-width="4px"/>
  <g transform="translate(50,50)">
    <circle r="50px" stroke="red" fill="pink"/>
    <text text-anchor="middle">Label</text>
  </g>
</svg>
```



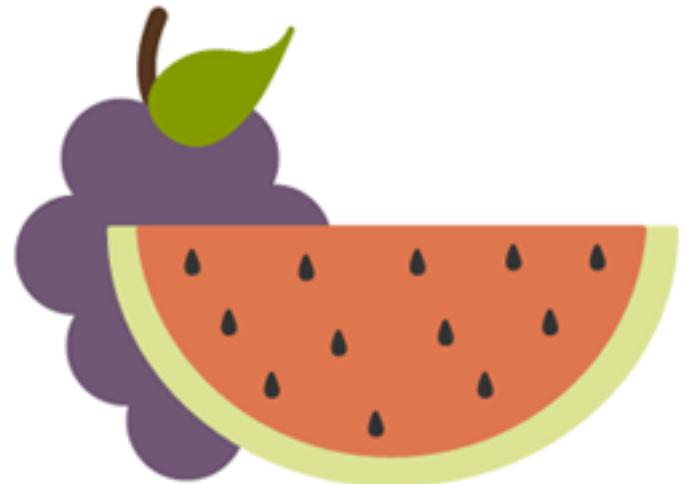
SVG – Construction and Margins

```
var width = 960;
var height = 600;
var margins = {left:10, right:10, top:10,
bottom:10}
var svg = d3.select("body").append("svg")
    .attr("width", width)
    .attr("height", height);

var g = svg.append("g")
    .attr("transform", "translate("
        + margins.left + ", "
        + margins.top + ")");
```

Stacking ordering

```
<svg>
  <g class="grapes">
    <!--<path <stem path> />-->
    <!--<path <grapes path> />-->
    <!--<path <leaf path> />-->
  </g>
  <g class="watermelon">
    <!--<path <outside path> />-->
    <!--<path <inside path> />-->
    <!--<path <seeds path> />-->
  </g>
</svg>
```



SVG – BASIC SHAPES

Rectangle

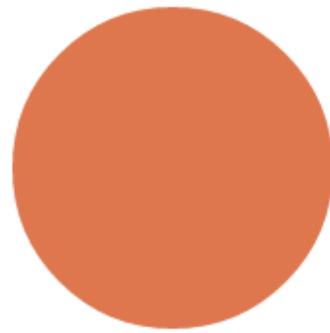
```
<svg>
  <rect width="200" height="100" fill="#BBC42A" />
</svg>
```



<http://codepen.io/jonitrythall/pen/525df2422b0ebc54c71a48d27534ea5e>

Circle

```
<svg>
  <circle cx="75" cy="75" r="75" fill="#ED6E46" />
</svg>
```



<http://codepen.io/jonitrythall/pen/088bbada7eed6739d09715666b945141>

Ellipse

```
<svg>
  <ellipse cx="100" cy="100" rx="100" ry="50"
fill="#7AA20D" />
</svg>
```



<http://codepen.io/jonitrythall/pen/8ec26dac6d5b64bc663c03f01c5d60e0>

Line

```
<svg>
  <line x1="5" y1="5" x2="100" y2="100"
stroke="#765373" stroke-width="8"/>
</svg>
```



Polyline

```
<svg>
  <polyline points="0,40 40,40 40,80 80,80 80,120 120,120 120,160"
fill="white" stroke="#BBC42A" stroke-width="6" />
</svg>
```



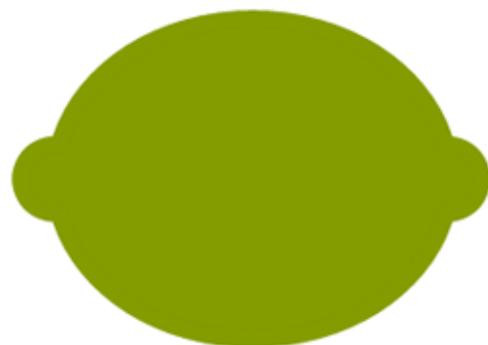
Polygon

```
<svg>
  <polygon points="50,5 100,5 125,30 125,80 100,105
50,105 25,80 25,30" fill="#ED6E46" />
</svg>
```



Path

```
<svg width="258px" height="184px">
  <path fill="#7AA20D" stroke="#7AA20D" stroke-width="9" stroke-
linejoin="round"
d="M248.761,92c0,9.801-7.93,17.731-17.71,17.731c-0.319,0-0.617,0-0.935-0
.021c-10.035,37.291-51.174,65.206-100.414,65.206
c-49.261,0-90.443-27.979-100.435-65.334c-0.765,0.106-1.531,0.149-2.317,0
.149c-9.78,0-17.71-7.93-17.71-17.731
c0-9.78,7.93-17.71,17.71-17.71c0.787,0,1.552,0.042,2.317,0.149C39.238,37
.084,80.419,9.083,129.702,9.083
c49.24,0,90.379,27.937,100.414,65.228h0.021c0.298-0.021,0.617-0.021,0.91
4-0.021C240.831,74.29,248.761,82.22,248.761,92z" />
</svg>
```



Path specifications

- Definition of a path is done within a path element
 - `<path d="{}{specifications of the path}" />`
- The specifications of path are instructions to move a virtual pen over the graphics
 - **Move to** (M or m). Go to coordinates lifting the pen, without a trace
 - **Line to** (L or l). Draw a line from the last point to the new coordinates
 - **Vertical or Horizontal lines** (H or h, V or v). Draw a line parallel to one of the axis
 - **Close path** (Z or z)

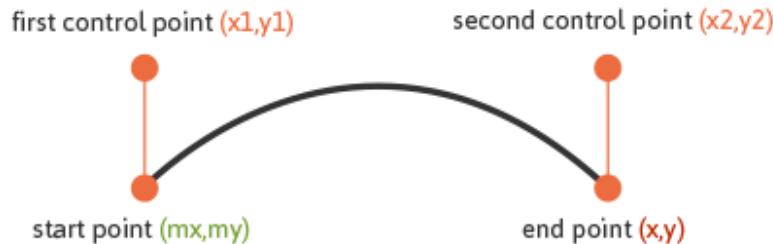
Path commands – Uppercase vs lowercase commands

- An uppercase letter indicates absolute coordinates will follow
- A lowercase letter indicates a relative coordinate

Path – Cubic Bezier

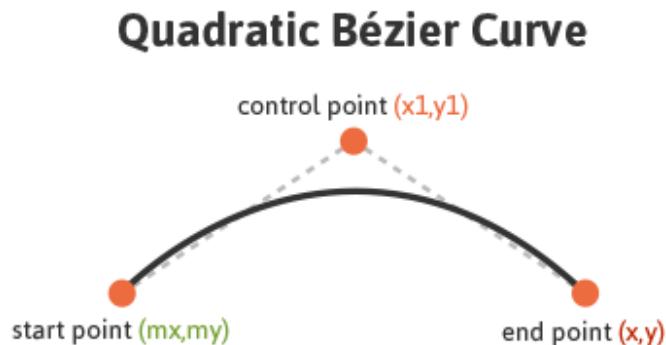
```
<svg>
    <path fill="none" stroke="#333333" stroke-width="3"
d="M10,55 C10,5 100,5 100,55" />
</svg>
```

Cubic Bézier Curve



Path – Quadratic bezier Curve

```
<svg>
    <path fill="none" stroke="#333333" stroke-width="3"
d="M20,50 Q40,5 100,50" />
</svg>
```



Example - Stairways with path

```
<!--  
  Stairways example using path  
-->  
  
<svg width="200" height="200">  
  <path d="M0,40 L40,40 L40,80 L80,80 L80,120 L120,120  
L120,160" fill="white" stroke="#BBC42A" stroke-width="6" />  
</svg>
```

- Live example at:
- <http://jsbin.com/xazajaw/2/edit?html,output>

Example - Stairways with path

```
<!--  
    Stairways example using path with H and V commands  
-->  
  
<svg width="200" height="200">  
    <path d="M0,40 H40 V80 H80 V120 H120 V160"  
fill="white" stroke="#BBC42A" stroke-width="6" />  
</svg>
```

- Live example at:
- <http://jsbin.com/xazajaw/2/edit?html,output>

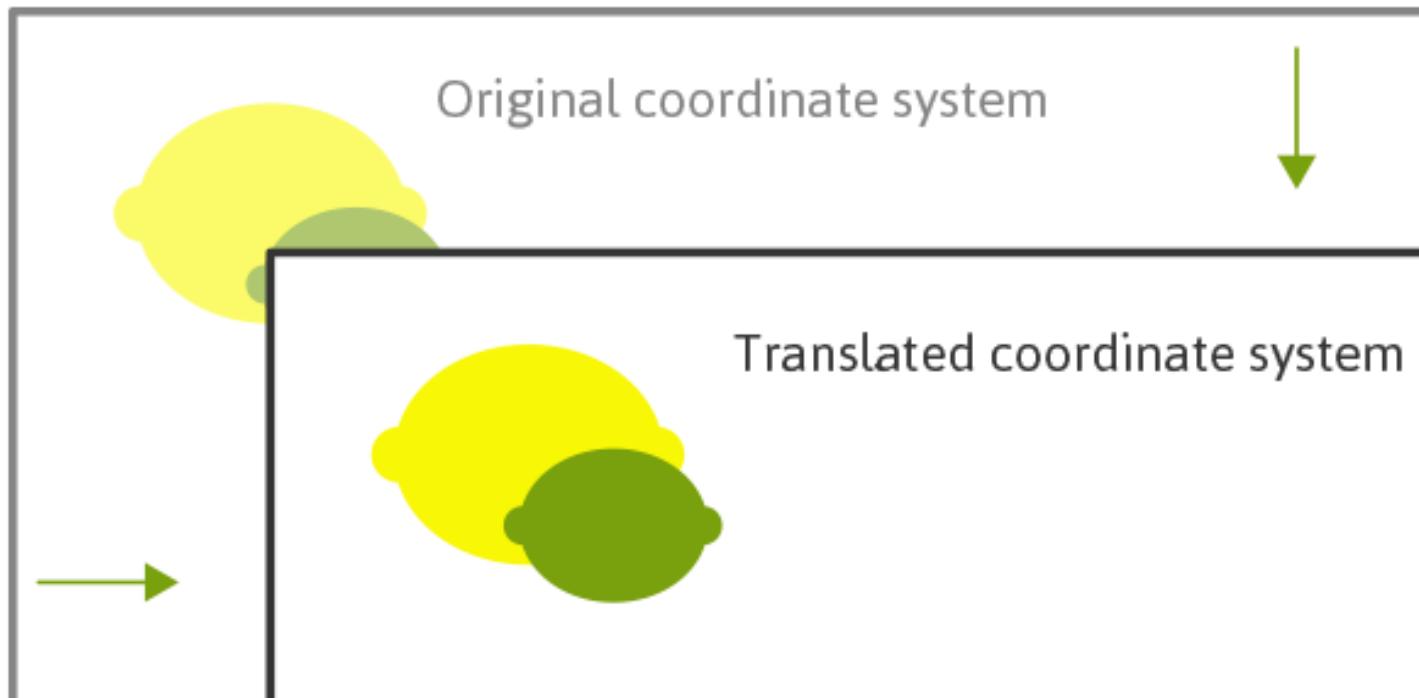
Example - Stairways with path

```
<!--  
    Stairways example using path with relative coordinates  
(h and v)  
-->  
<svg width="200" height="200">  
    <path d="M0,40 h40 v40 h40 v40 h40 v40" fill="white"  
stroke="#BBC42A" stroke-width="6" />  
</svg>
```

- Live example at:
- <http://jsbin.com/xazajaw/2/edit?html,output>

Coordinate System Transform

```
transform="translate(<tx>,<ty>) rotate(<rotation angle>)
```



Transformations

- Translate
 - `transform="translate(<tx>,<ty>)"`
- Rotate
 - `transform="rotate(<rotation angle>)"`
 - `transform=rotate(<rotation angle> [<cx>,<cy>])"`
- Scale
 - `transform="scale(<sx> [<sy>])"`
- Skew
 - `transform="skewX(20)"`

Circle example with translation

```
<!--  
    Draw a circle in the center of the element  
    using relative coordinates after a translation  
-->  
<svg width="200" height="100">  
  <g transform="translate(100,50)">  
    <circle r="50"/>  
    <circle r="20" style="fill:#fdbb84"/>  
  </g>  
</svg>
```

- Live example at:
- <http://jsbin.com/kiwukat/2/edit?html,output>

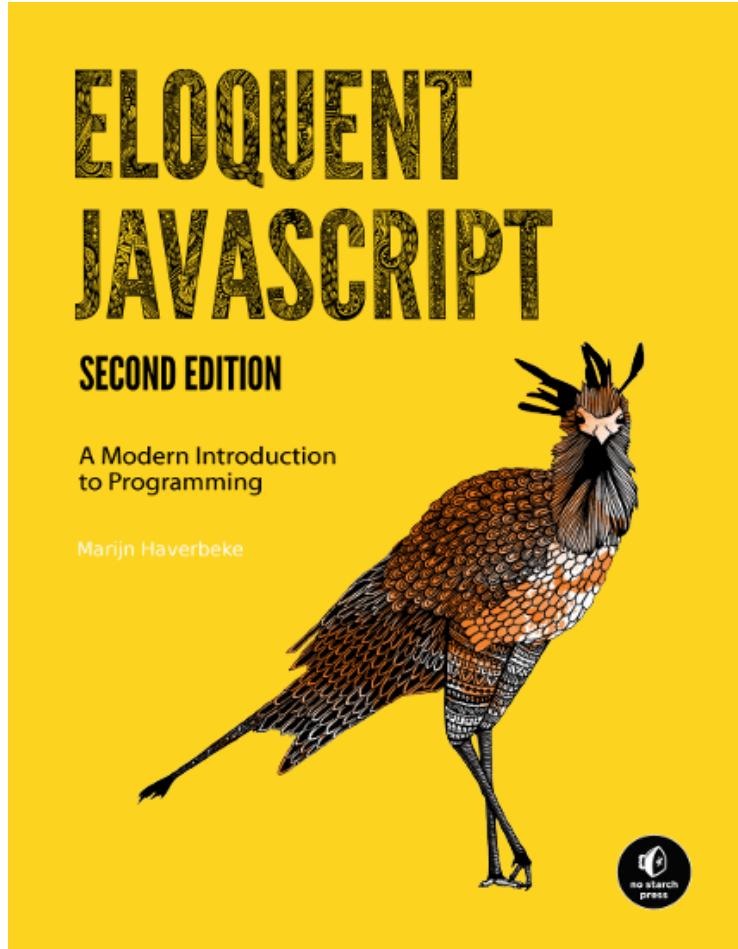
Pocket Guide to Writing SVG

<http://svgpocketguide.com/book/>



JAVASCRIPT

Javascript



Eloquent Javascript – Second Edition

Marijn Haverbeke

Licensed under CC license.

Available here: <http://eloquentjavascript.net/>

CANVAS ELEMENT

Canvas

- A **canvas** element is a container for raster graphics
- Within the canvas, a **context** provide the functions to draw visual elements
- Two different context types:
 - “2d”
 - “webgl”

Canvas - Example

```
<p>Before canvas.</p>
<canvas width="120" height="60"></canvas>
<p>After canvas.</p>
<script>
  var canvas =
document.querySelector("canvas");
  var context = canvas.getContext("2d");
  context.fillStyle = "red";
  context.fillRect(10, 10, 100, 50);
</script>
```

Canvas - Path

```
<canvas></canvas>
<script>
  var cx =
document.querySelector("canvas").getContext("2d");
  cx.beginPath();
  for (var y = 10; y < 100; y += 10) {
    cx.moveTo(10, y);
    cx.lineTo(90, y);
  }
  cx.stroke();
</script>
```

Canvas - Curves

```
<canvas></canvas>
<script>
  var cx =
document.querySelector("canvas").getContext("2d");
  cx.beginPath();
  cx.moveTo(10, 90);
  // control=(60,10) goal=(90,90)
  cx.quadraticCurveTo(60, 10, 90, 90);
  cx.lineTo(60, 10);
  cx.closePath();
  cx.stroke();
</script>
```

Canvas - Curves

```
<canvas></canvas>
<script>
  var cx =
document.querySelector("canvas").getContext("2d");
  cx.beginPath();
  cx.moveTo(10, 90);
  // control1=(10,10) control2=(90,10) goal=(50,90)
  cx.bezierCurveTo(10, 10, 90, 10, 50, 90);
  cx.lineTo(90, 10);
  cx.lineTo(10, 10);
  cx.closePath();
  cx.stroke();
</script>
```

VISUALIZATION ON THE WEB

tableau.com



DATA ANALYSIS SOFTWARE

[START YOUR FREE TRIAL](#)

Full-version trial. No credit card required.



Kibana GA



Kibi

Screenshot of the Kibi Relational Search and Analytics interface.

Header: Discover, Visualize, Dashboard, Settings, Last 15 minutes.

Left sidebar:

- Molecules Molecules
- Activities (13520737)
- Assays (1148941)
- Targets (10776)
- Papers (59610)

Top 10 molecule_type:

molecule_type	Count
Small molecule	1,437,508
Protein	19,405
Unknown	5,379
Antibody	718
Enzyme	88
Oligonucleotide	88
Oligosaccharide	60
Cell	22
Unclassified	6

Top 500 indication_class:

indication_class	Count
Antibacterial	319
Antineoplastic	187
Antidepressant	99
Antihypertensive	97
Anti-Inflammatory	89
Analgesic	81
Antipsychotic	60
Radioactive Agent	73

Molecules search results:

pref_name	molecule_type	availability_type	synonyms	chirality
(2S,4S,5R,6R)-6-acetamido-6-((1R,2R)-3-azahexyl)mercury	Small molecule	-1	-	-1
(1R,9S,12S)-12-Methoxy-9-methyl-10,11,11-trimethyl-stannane	Small molecule	-1	-	-1
(1R,9S,12S)-12-Methoxy-9-methyl-10,11,11-triphenyl-stannane	Small molecule	-1	-	-1
(+)-NEOMENTHOL	Small molecule	-1	(+)-Neomenthol	-1
(+)-RR-dichloro-[1,2-bis(4-hydroxyphenyl)]ethanum (B)	Small molecule	-1	-	-1
(+)-(8S)-PARASORBIC ACID	Small molecule	-1	-	-1
(+)-11-DEMETHYL CALANOLIDE A	Small molecule	-1	(+)-11-demethyl calanolide A	-1
(+)-11-DEMETHYL CORDATOLIDE A	Small molecule	-1	(+)-11-demethyl cordatolide A	-1
1-(3,5-DICFTYL-1-THIOL ACTONE	Small molecule	-1	-	-1

Relational Button Activities:

- .. show related activities (13520737)

Therapeutic vs Non (Chirality):

Legend: -1 (Green), 1 (Blue), 2 (Purple), 0 (White)

The first chart (top) shows a large majority of molecules with chirality type -1 (green). The second chart (bottom) shows a mix of types, with 1 (blue) being the largest segment.

<https://siren.solutions/kibi/>

Superset



World's Bank Data



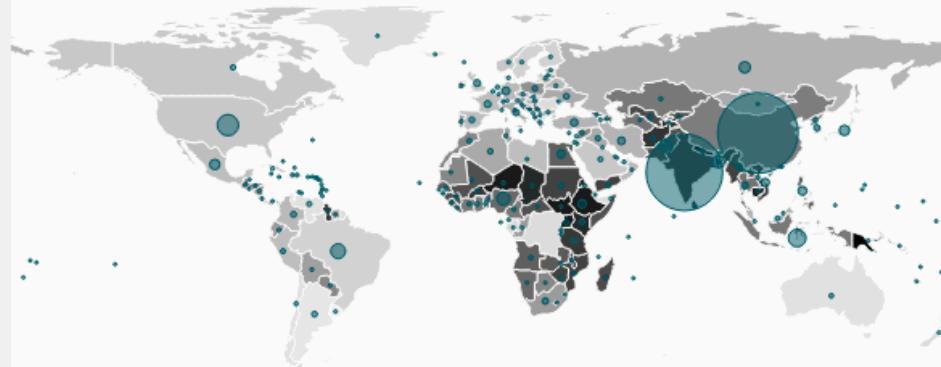
Region Filter

region

World's Population

7.24G
+12.9% over 10Y

% Rural



Most Populated Countries

country_name	sum_SP_POP_TOTL
China	1.36G
India	1.30G
United States	319M
Indonesia	254M
Brazil	206M
Pakistan	185M
Nigeria	177M
Bangladesh	159M
Russian Federation	144M
Japan	127M
Mexico	125M
Philippines	99.1M
Ethiopia	97.0M
Vietnam	90.7M
Egypt, Arab Rep.	89.6M
Germany	80.9M
Iran, Islamic Rep.	78.1M
Turkey	75.9M
Congo, Dem.	74.9M

NVD3.js

NVD3.js [Home](#) [Examples](#) [Live Code](#) [Source](#) [Blog](#) [Downloads: ZIP TAR.GZ](#)

NVD3 Re-usable charts for d3.js

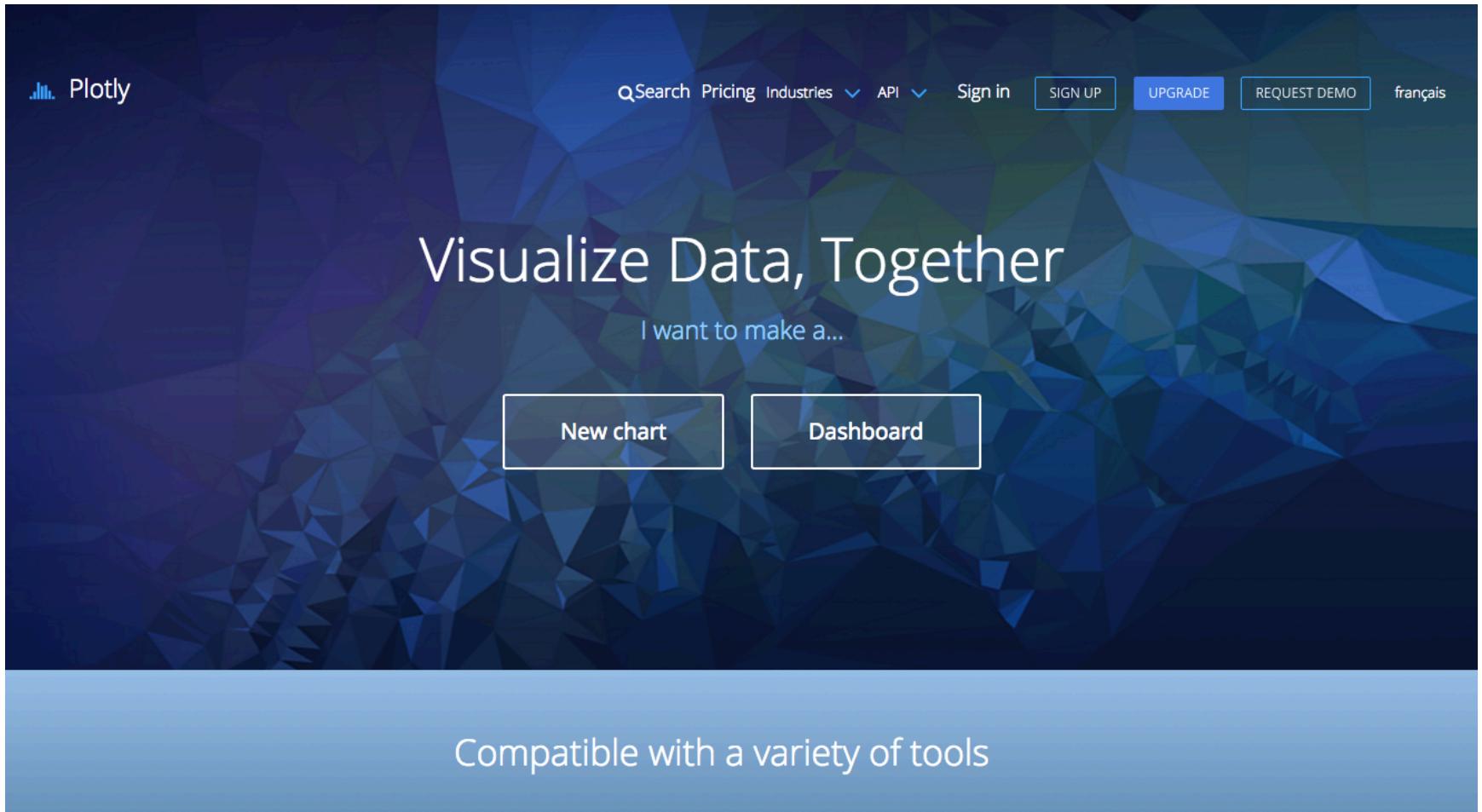
This project is an attempt to build re-usable charts and chart components for d3.js without taking away the power that d3.js gives you. This is a very young collection of components, with the goal of keeping these components very customizable, staying away from your standard cookie cutter solutions.

[View more examples »](#)

[GitHub Repo](#)

The page displays three charts illustrating the capabilities of NVD3.js:

- Grouped Bar Chart:** Shows three data streams (Stream0, Stream1, Stream2) as grouped bars. Stream0 is blue, Stream1 is light blue, and Stream2 is orange. The x-axis has ticks at 0, 7, 15, 23, and 31. The y-axis ranges from 0.0 to 3.4. Stream2 has peaks around x=12, 18, and 24.
- Stacked Area Chart:** Shows three stacked areas (Stream0, Stream1, Stream2). Stream0 is blue, Stream1 is light blue, and Stream2 is orange. The x-axis has ticks at 0, 10, 20, 30, and 36. The y-axis ranges from 0.0 to 4.5. The total height of the stacked areas fluctuates between 1.0 and 3.5.
- Line Chart:** Shows three data streams (Stream0, Stream1, Stream2) as lines. Stream0 is blue, Stream1 is light blue, and Stream2 is orange. The x-axis has ticks at 0, 10, 20, 30, and 36. The y-axis ranges from 0.1 to 3.4. Stream2 shows a sharp peak near x=12.



The image shows the homepage of Plotly. The background features a dark blue polygonal pattern. At the top left is the Plotly logo. The top navigation bar includes a search bar, links for Pricing, Industries, API, Sign in, SIGN UP (in a white box), UPGRADE (in a blue box), REQUEST DEMO (in a white box), and a French language link (français). The main headline reads "Visualize Data, Together" with a subtitle "I want to make a...". Below this are two buttons: "New chart" and "Dashboard". A footer banner at the bottom states "Compatible with a variety of tools".

Plotly

Search Pricing Industries API Sign in SIGN UP UPGRADE REQUEST DEMO français

Visualize Data, Together

I want to make a...

New chart Dashboard

Compatible with a variety of tools