

➤ Tutoriel sur SVG ou Canvas en D3.js

Work4Graph



CATI Sysmics

D3JS : D3 = Data Driven Documents

- Librairie Javascript OpenSource pour visualiser des données en utilisant le SVG, HTML, et CSS,
- V1 en 2011 => v6.2 Sept. 2020,
- Forte communauté, Reactivité forte,
- Enormement d'exemples, Visualisations sans limite



D3JS : Philosophie

- Même approche que JQuery
- Les données font le Graph! Approche pilotée par les données
- Manipulation du DOM (Document Objet Model), et donc du SVG
- Gestion fine des évènements
- Utilisable en html et/ou avec npm/node



D3JS : Comment ça marche?

- Installation :
 - `<script src="https://d3js.org/d3.v6.js"></script>`
- Utilisation de D3 pour manipuler le DOM
 - <https://liveweave.com/> : pour éditer notre code en ligne



D3JS & le DOM

HTML CODE

```
<html>
  <head>
    <meta charset="utf-8">
    <script src="https://d3js.org/d3.v6.js"></script>
  </head>

  <body>
  </body>
</html>
```

JAVASCRIPT CODE

```
// Recuperation du champ Body du DOM
var myBody = d3.select("body") ;

// Ajout dans le DOM avec D3
myBody.append("span").text("Hello World");
```

D3JS & le DOM

HTML CODE

```
<html>
  <head>
    <meta charset="utf-8">
    <script src="https://d3js.org/d3.v6.js"></script>
  </head>

  <body>

    <span>Hello World</span>

  </body>
</html>
```

JAVASCRIPT CODE

```
// Recuperation du champ Body du DOM
var myBody = d3.select("body") ;

// Ajout dans le DOM avec D3
myBody.append("span").text("Hello World");
```

OUTPUT

Hello World



D3JS & le DOM

HTML CODE

```
<html>
<head>
  <meta charset="utf-8">
  <script src="https://d3js.org/d3.v6.js"></script>
</head>
<body>
  <table border=1>
    <tr><td>col1</td><td>col2</td></tr>
    <tr><td>col1</td><td>col2</td></tr>
    <tr><td>col1</td><td>col2</td></tr>
  </table>
</body>
</html>
```

JAVASCRIPT CODE

```
// Recuperation du champ Body du DOM
var myBody = d3.select("body") ;

// Modification dans le DOM avec D3
d3.selectAll("td")
  .style("background-color", "red");
```

D3JS & le DOM

HTML CODE

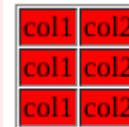
```
<html>
<head>
  <meta charset="utf-8">
  <script src="https://d3js.org/d3.v6.js"></script>
</head>
<body>
  <table border=1>
    <tr><td>col1</td><td>col2</td></tr>
    <tr><td>col1</td><td>col2</td></tr>
    <tr><td>col1</td><td>col2</td></tr>
  </table>
</body>
</html>
```

JAVASCRIPT CODE

```
// Recuperation du champ Body du DOM
var myBody = d3.select("body") ;

// Modification dans le DOM avec D3
d3.selectAll("td")
  .style("background-color", "red");
```

OUTPUT



col1	col2
col1	col2
col1	col2

D3JS : Let's go SVG

HTML CODE

```
<html>
<head>
  <meta charset="utf-8">
  <script src="https://d3js.org/d3.v6.js"></script>
</head>

<body>
</body>
</html>
```

JAVASCRIPT CODE

```
// Recuperation du champ Body du DOM
var myBody = d3.select("body")

var graphWidth = 300; // Largeur
var graphHeight = 200; // Hauteur

// definition de l'espace de dessin
// Ajout d'un champ svg au body
var mySVG = myBody.append("svg")
  .attr('width', graphWidth + 40) // Largeur du SVG
  .attr('height', graphHeight + 40) // Hauteur du SVG
  .style("background-color", '#ADD8E6') // Couleur
  .append("g") // Ajout d'un groupe d'element S
  .attr("transform", "translate(20,20)"); // On translate ce groupe
  // pour laisser une marge
```

OUTPUT

D3JS : Let's go SVG

HTML CODE

```
<html>
<head>
  <meta charset="utf-8">
  <script src="https://d3js.org/d3.v6.js"></script>
</head>

<svg width="320" height="240"
      style="background-color: rgb(173, 216, 230);">
  <g transform="translate(20,20)">
  </g>
</svg>

  <body>
</body>
</html>
```

JAVASCRIPT CODE

```
// Recuperation du champ Body du DOM
var myBody = d3.select("body")

var graphWidth = 300;      // Largeur
var graphHeight = 200 ;   // Hauteur

// definition de l'espace de dessin
// Ajout d'un champ svg au body
var mySVG = myBody.append("svg")
  .attr('width',    graphWidth  + 40)    // Largeur du SVG
  .attr('height',  graphHeight + 40)    // Hauteur du SVG
  .style("background-color", '#ADD8E6') // Couleur
  .append("g") // Ajout d'un groupe d'element S
  .attr("transform", "translate(20,20)"); // On translate ce groupe
                                           // pour laisser une marge
```

OUTPUT



D3JS : Let's go SVG

HTML CODE

```
...  
<body>  
  <svg width="320" height="240"  
    style="background-color: rgb(173, 216, 230);">  
    <g transform="translate(20,20)">  
  
    </g>  
  </svg>  
</body>  
...
```

OUTPUT



en D3.js

JAVASCRIPT CODE

```
// Recuperation du champ Body du DOM  
var myBody = d3.select("body")  
  
var graphWidth = 300; // Largeur  
var graphHeight = 200 ; // Hauteur  
  
// definition de l'espace de dessin  
// Ajout d'un champ svg au body  
var mySVG = myBody.append("svg")  
  .attr('width', graphWidth + 40) // Largeur du SVG  
  .attr('height', graphHeight + 40) // Hauteur du SVG  
  .style("background-color", '#ADD8E6') // Couleur  
  .append("g") // Ajout d'un groupe d'element SVG  
  .attr("transform","translate(20,20)");// On translate ce groupe  
  // pour laisser une marge  
  
var myData = [ {x :0, y:8 } ,{x :1, y:2 } , {x :3, y:12 }];  
  
mySVG  
  .selectAll('circle')// Selection des cercles du SVG  
  .data(myData) // en selectionnant mes donnees  
  .enter() // Enter cree les rect qui n'ont  
  // pas encore ete cree a partir des data  
  .append("circle");// Ajouter des cercles SVG
```

D3JS : Let's go SVG

HTML CODE

```
...  
<body>  
  <svg width="320" height="240"  
    style="background-color: rgb(173, 216, 230);">  
    <g transform="translate(20,20)">  
      <circle></circle>  
      <circle></circle>  
      <circle></circle>  
    </g>  
  </svg>  
</body>  
...
```

OUTPUT



en D3.js

JAVASCRIPT CODE

```
// Recuperation du champ Body du DOM  
var myBody = d3.select("body")  
  
var graphWidth = 300; // Largeur  
var graphHeight = 200 ; // Hauteur  
  
// definition de l'espace de dessin  
// Ajout d'un champ svg au body  
var mySVG = myBody.append("svg")  
  .attr('width', graphWidth + 40) // Largeur du SVG  
  .attr('height', graphHeight + 40) // Hauteur du SVG  
  .style("background-color", '#ADD8E6') // Couleur  
  .append("g") // Ajout d'un groupe d'element SVG  
  .attr("transform","translate(20,20)");// On translate ce groupe  
  // pour laisser une marge  
  
var myData = [ {x :0, y:8 } ,{x :1, y:2 } , {x :3, y:12 }];  
  
mySVG  
  .selectAll('circle')// Selection des cercles du SVG  
  .data(myData) // en selectionnant mes donnees  
  .enter() // Enter cree les rect qui n'ont  
  // pas encore ete cree a partir des data  
  .append("circle");// Ajouter des cercles SVG
```

D3JS : It's a kind of magic

JAVASCRIPT CODE

```
// Recuperation du champ Body du DOM
var myBody = d3.select("body")

var graphWidth = 300;      // Largeur
var graphHeight = 200 ;   // Hauteur

// definition de l'espace de dessin
// Ajout d'un champ svg au body
var mySVG = myBody.append("svg")
    .attr('width',    graphWidth + 40) // Largeur du SVG
    .attr('height',  graphHeight + 40) // Hauteur du SVG
    .style("background-color", '#ADD8E6') // Couleur
    .append("g") // Ajout d'un groupe d'element SVG
    .attr("transform", "translate(20,20)"); // On translate ce groupe
                                           // pour laisser une marge

var myData = [ {x :0, y:8 } , {x :1, y:2 } , {x :3, y:12 }];

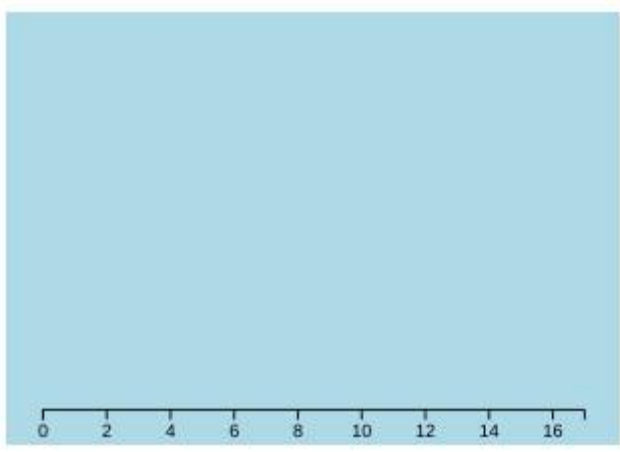
mySVG
    .selectAll('circle') // Selection des cercles du SVG
    .data(myData)        // en selectionnant mes donnees
    .enter()             // Enter cree les rect qui n'ont
                        // pas encore ete cree a partir des data
    .append("circle"); // Ajouter des cercles SVG

var minX = myData.reduce((min, b) => Math.min(min, b.x), myData[0].x);
var maxX = myData.reduce((max, b) => Math.max(max, b.x), myData[0].x);

var x = d3.scaleLinear() // creation de l'axe des X
    .domain([minX, maxX]) // Valeurs affichees
    .range([ 0, graphWidth ]); // Espace de travail

mySVG.append("g") // Ajout de l'axe au svg
    .attr("transform", "translate(0," + graphHeight + ")")
    .call(d3.axisBottom(x));
```

OUTPUT



D3JS : It's a kind of magic

JAVASCRIPT CODE 1/2

```
// Recuperation du champ Body du DOM
var myBody = d3.select("body")

var graphWidth = 300;    // Largeur
var graphHeight = 200 ; // Hauteur

// definition de l'espace de dessin
// Ajout d'un champ svg au body
var mySVG = myBody.append("svg")
  .attr('width',    graphWidth + 40) // Largeur du SVG
  .attr('height',  graphHeight + 40) // Hauteur du SVG
  .style("background-color", '#ADD8E6') // Couleur
  .append("g") // Ajout d'un groupe
  .attr("transform", "translate(20,20)"); // On translate ce g
  // pour laisser une marge
```

JAVASCRIPT CODE 2/2

```
var myData = [ {x :0, y:8 } ,{x :1, y:2 } , {x :3, y:12 }];

mySVG
  .selectAll('rect') // Selection des rectangles du SVG
  .data(myData)      // en selectionnant mes donnees
  .enter()           // Enter cree les rect qui n'ont
                    // pas encore ete cree a partir des data
  .append("circle");// Ajouter des cercles SVG

var minX = myData.reduce((min, b) => Math.min(min, b.x), myData[0].x);
var maxX = myData.reduce((max, b) => Math.max(max, b.x), myData[0].x);

var x = d3.scaleLinear() // creation de l'axe des X
  .domain([minX, maxX]) // Valeurs affichees
  .range([ 0, graphWidth ]); // Espace de travail

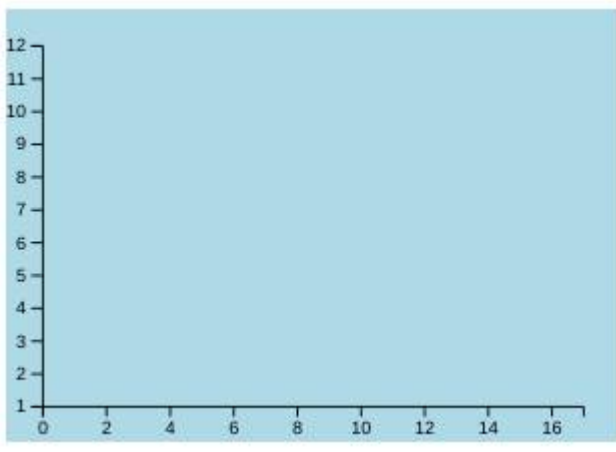
mySVG.append("g") // Ajout de l'axe au svg
  .attr("transform", "translate(0," + graphHeight + ")")
  .call(d3.axisBottom(x));

var minY = myData.reduce((min, b) => Math.min(min, b.y), myData[0].y);
var maxY = myData.reduce((max, b) => Math.max(max, b.y), myData[0].y);

var y = d3.scaleLinear() // creation de l'axe des Y
  .domain([minY, maxY]) // Valeurs affichees
  .range([ graphHeight, 0]); // Espace de travail

mySVG.append("g") // Ajout de l'axe au svg
  .call(d3.axisLeft(y));
```

OUTPUT



travailler sur SVG ou canvas en D3.js

08/12/20 Franck Samsen - LaMME

D3JS : Add my points!

JAVASCRIPT CODE 1/2

```
// Recuperation du champ Body du DOM
var myBody = d3.select("body")

var graphWidth = 300;    // Largeur
var graphHeight = 200 ; // Hauteur

// definition de l'espace de dessin
// Ajout d'un champ svg au body
var mySVG = myBody.append("svg")
  .attr('width', graphWidth + 40) // Largeur du SVG
  .attr('height', graphHeight + 40) // Hauteur du SVG
  .style("background-color", '#ADD8E6') // Couleur
  .append("g") // Ajout d'un groupe
  .attr("transform", "translate(20,20)"); // On translate ce g
  // pour laisser une m
```

JAVASCRIPT CODE 2/2

```
var myData = [ {x :0, y:8 } , {x :1, y:2 } , {x :3, y:12 }];

var minX = myData.reduce((min, b) => Math.min(min, b.x), myData[0].x);
var maxX = myData.reduce((max, b) => Math.max(max, b.x), myData[0].x);

var x = d3.scaleLinear() // creation de l'axe des X
  .domain([minX, maxX]) // Valeurs affichees
  .range([ 0, graphWidth ]); // Espace de travail

mySVG.append("g") // Ajout de l'axe au svg
  .attr("transform", "translate(0," + graphHeight + ")")
  .call(d3.axisBottom(x));

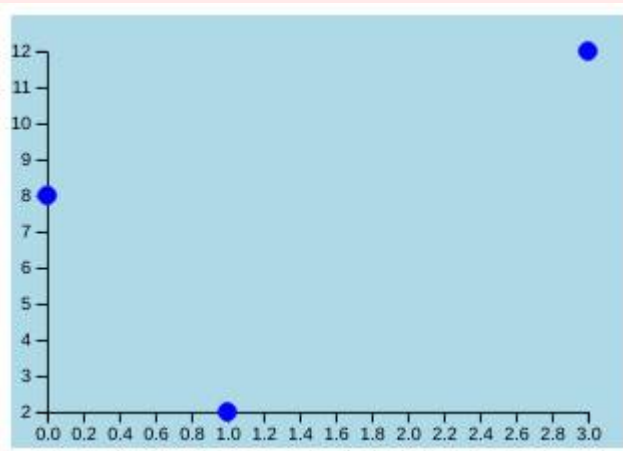
var minY = myData.reduce((min, b) => Math.min(min, b.y), myData[0].y);
var maxY = myData.reduce((max, b) => Math.max(max, b.y), myData[0].y);

var y = d3.scaleLinear() // creation de l'axe des Y
  .domain([minY, maxY]) // Valeurs affichees
  .range([ graphHeight, 0]); // Espace de travail

mySVG.append("g") // Ajout de l'axe au svg
  .call(d3.axisLeft(y));

// Creation des points du graph
mySVG
  .selectAll('circle') // Selection des cercles du SVG
  .data(myData) // en selectionnant mes donnees
  .enter() // Enter cree les rect qui n'ont
  // pas encore ete crees a partir des data
  .append("circle") // Ajouter des cercles SVG
    .attr("cx", function (d) { return x(d.x); } )
    .attr("cy", function (d) { return y(d.y); } )
    .attr("r", 5)
    .attr("stroke", "#2222ff")
    .attr("fill", "blue")
```

OUTPUT



D3JS : Add Events

JAVASCRIPT CODE 1/2

```
// Recuperation du champ Body du DOM
var myBody = d3.select("body")

var graphWidth = 300;    // Largeur
var graphHeight = 200 ; // Hauteur

// definition de l'espace de dessin
// Ajout d'un champ svg au body
var mySVG = myBody.append("svg")
  .attr('width', graphWidth + 40) // Largeur du SVG
  .attr('height', graphHeight + 40) // Hauteur du SVG
  .style("background-color", '#ADD8E6') // Couleur
  .append("g") // Ajout d'un groupe
  .attr("transform", "translate(20,20)"); // On translate ce g
  // pour laisser une marge
```

JAVASCRIPT CODE 2/2

```
var myData = [ {x :0, y:8 } , {x :1, y:2 } , {x :3, y:12 }];

var minX = myData.reduce((min, b) => Math.min(min, b.x), myData[0].x);
var maxX = myData.reduce((max, b) => Math.max(max, b.x), myData[0].x);

var x = d3.scaleLinear() // creation de l'axe des X
  .domain([minX, maxX]) // Valeurs affichees
  .range([ 0, graphWidth ]); // Espace de travail

mySVG.append("g") // Ajout de l'axe au svg
  .attr("transform", "translate(0," + graphHeight + ")")
  .call(d3.axisBottom(x));

var minY = myData.reduce((min, b) => Math.min(min, b.y), myData[0].y);
var maxY = myData.reduce((max, b) => Math.max(max, b.y), myData[0].y);

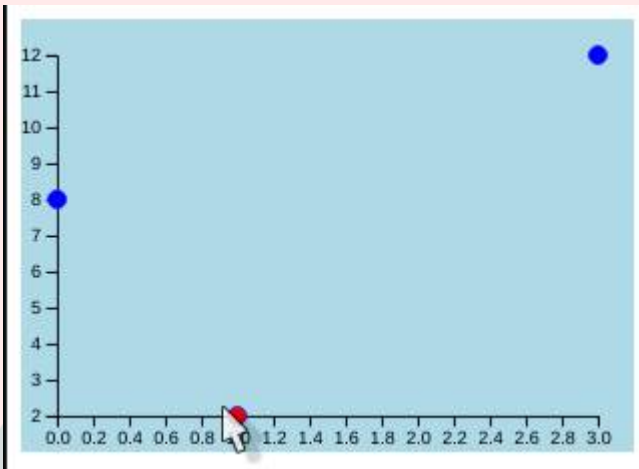
var y = d3.scaleLinear() // creation de l'axe des Y
  .domain([minY, maxY]) // Valeurs affichees
  .range([ graphHeight, 0]); // Espace de travail

mySVG.append("g") // Ajout de l'axe au svg
  .call(d3.axisLeft(y));

// Creation des points du graph
mySVG
  .selectAll('circle') // Selection des cercles du SVG
  .data(myData) // en selectionnant mes donnees
  .enter() // Enter cree les rect qui n'ont
  // pas encore ete cree a partir des data
  .append("circle") // Ajouter des cercles SVG
  .attr("cx", function (d) { return x(d.x); } )
  .attr("cy", function (d) { return y(d.y); } )
  .attr("r", 5)

  .attr("stroke", "#2222ff")
  .attr("fill", "blue")
  .on("mouseover",
    function () { d3.select(this).attr('fill', 'red'); })
  .on("mouseout",
    function () { d3.select(this).attr('fill', 'blue');})
```

OUTPUT



D3JS : Add Events

JAVASCRIPT CODE 1/2

```
// Recuperation du champ Body du DOM
var myBody = d3.select("body")

var graphWidth = 300;    // Largeur
var graphHeight = 200 ; // Hauteur

// definition de l'espace de dessin
// Ajout d'un champ svg au body
var mySVG = myBody.append("svg")
  .attr('width', graphWidth + 40) // Largeur du SVG
  .attr('height', graphHeight + 40) // Hauteur du SVG
  .style("background-color", '#ADD8E6') // Couleur
  .append("g") // Ajout d'un groupe
  .attr("transform", "translate(20,20)"); // On translate ce g
  // pour laisser une marge
```

JAVASCRIPT CODE 2/2

```
var myData = [ {x :0, y:8 } , {x :1, y:2 } , {x :3, y:12 }];

var minX = myData.reduce((min, b) => Math.min(min, b.x), myData[0].x);
var maxX = myData.reduce((max, b) => Math.max(max, b.x), myData[0].x);

var x = d3.scaleLinear() // creation de l'axe des X
  .domain([minX, maxX]) // Valeurs affichees
  .range([ 0, graphWidth ]); // Espace de travail

mySVG.append("g") // Ajout de l'axe au svg
  .attr("transform", "translate(0," + graphHeight + ")")
  .call(d3.axisBottom(x));

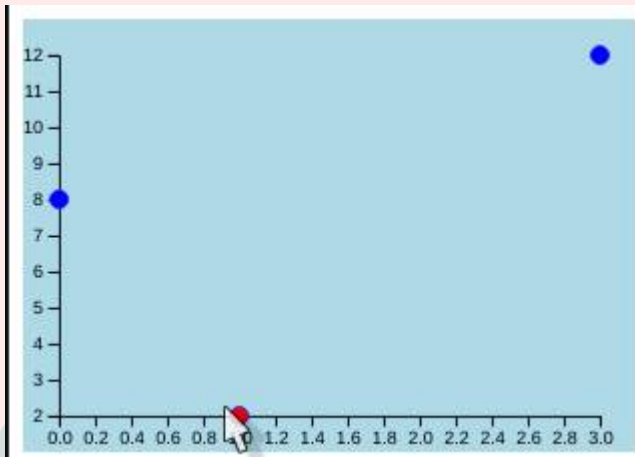
var minY = myData.reduce((min, b) => Math.min(min, b.y), myData[0].y);
var maxY = myData.reduce((max, b) => Math.max(max, b.y), myData[0].y);

var y = d3.scaleLinear() // creation de l'axe des Y
  .domain([minY, maxY]) // Valeurs affichees
  .range([ graphHeight - 0, 0]); // Espace de travail

// Ajout de l'axe au svg
```

Lets go to the HTML CODE

OUTPUT



```
// Creation des points du graph
mySVG
  .selectAll('circle') // Selection des cercles du SVG
  .data(myData) // en selectionnant mes donnees
  .enter() // Enter cree les rect qui n'ont
  // pas encore ete cree a partir des data
  .append("circle") // Ajouter des cercles SVG
  .attr("cx", function (d) { return x(d.x); } )
  .attr("cy", function (d) { return y(d.y); } )
  .attr("r", 5)

  .attr("stroke", "#2222ff")
  .attr("fill", "blue")
  .on("mouseover",
    function () { d3.select(this).attr('fill', 'red'); })
  .on("mouseout",
    function () { d3.select(this).attr('fill', 'blue');})
```

D3JS : Add Events

JAVASCRIPT CODE 1/2

JAVASCRIPT CODE 2/2

```
// Re  
var m  
  
var g  
var g  
  
// de  
// Aj  
var m  
.at  
.at  
.st  
.ap  
.at
```

```
<svg width="340" height="240" style="background-color: rgb(173, 216, 230);">  
<g transform="translate(20,20)">  
<g class="tick" opacity="1" transform="translate(20.5,0)">  
<path class="domain" stroke="currentColor" d="M0.5,6V0.5H300.5V6"></path>  
<g class="tick" opacity="1" transform="translate(0.5,0)">  
<line stroke="currentColor" y2="6"></line>  
<text fill="currentColor" y="9" dy="0.71em">0.0</text></g>  
<g class="tick" opacity="1" transform="translate(20.5,0)">  
<line stroke="currentColor" y2="6"></line><text fill="currentColor" y="9" dy="0.71em">0.2</text></g>  
<g class="tick" opacity="1" transform="translate(40.5,0)"><line stroke="currentColor" y2="6"></line>  
<text fill="currentColor" y="9" dy="0.71em">0.4</text></g>  
<g class="tick" opacity="1" transform="translate(60.49999999999999,0)"><line stroke="currentColor" y2="6"></line><text fill="currentColor" y="9" dy="0.71em">0.6</text></g>  
<g class="tick" opacity="1" transform="translate(80.5,0)"><line stroke="currentColor" y2="6"></line><text fill="currentColor" y="9" dy="0.71em">0.8</text></g>  
<g class="tick" opacity="1" transform="translate(100.5,0)"><line stroke="currentColor" y2="6"></line>  
<text fill="currentColor" y="9" dy="0.71em">1.0</text></g>  
<g class="tick" opacity="1" transform="translate(120.49999999999999,0)"><line stroke="currentColor" y2="6"></line>  
<text fill="currentColor" y="9" dy="0.71em">1.2</text></g><g class="tick" opacity="1" transform="translate(140.49999999999997,0)">  
<line stroke="currentColor" y2="6"></line><text fill="currentColor" y="9" dy="0.71em">1.4</text></g><g class="tick" opacity="1" transform="translate(160.5,0)">  
<line stroke="currentColor" y2="6"></line><text fill="currentColor" y="9" dy="0.71em">1.6</text></g><g class="tick" opacity="1" transform="translate(180.5,0)">  
<line stroke="currentColor" y2="6"></line><text fill="currentColor" y="9" dy="0.71em">1.8</text></g><g class="tick" opacity="1" transform="translate(200.5,0)">  
<line stroke="currentColor" y2="6"></line><text fill="currentColor" y="9" dy="0.71em">2.0</text></g><g class="tick" opacity="1" transform="translate(220.50000000000003,0)">  
<line stroke="currentColor" y2="6"></line><text fill="currentColor" y="9" dy="0.71em">2.2</text></g><g class="tick" opacity="1" transform="translate(240.49999999999997,0)">  
<line stroke="currentColor" y2="6"></line><text fill="currentColor" y="9" dy="0.71em">2.4</text></g><g class="tick" opacity="1" transform="translate(260.5,0)">  
<line stroke="currentColor" y2="6"></line><text fill="currentColor" y="9" dy="0.71em">2.6</text></g><g class="tick" opacity="1" transform="translate(280.49999999999994,0)">  
<line stroke="currentColor" y2="6"></line><text fill="currentColor" y="9" dy="0.71em">2.8</text></g><g class="tick" opacity="1" transform="translate(300.5,0)">  
<line stroke="currentColor" y2="6"></line><text fill="currentColor" y="9" dy="0.71em">3.0</text></g></g>  
<g fill="none" font-size="10" font-family="sans-serif" text-anchor="end"><path class="domain" stroke="currentColor" d="M-6,200.5H0.5V0.5H-6"></path>  
<g class="tick" opacity="1" transform="translate(0,200.5)"><line stroke="currentColor" x2="-6"></line><text fill="currentColor" x="-9" dy="0.32em">2</text></g>  
<g class="tick" opacity="1" transform="translate(0,180.5)"><line stroke="currentColor" x2="-6"></line><text fill="currentColor" x="-9" dy="0.32em">3</text></g>  
<g class="tick" opacity="1" transform="translate(0,160.5)"><line stroke="currentColor" x2="-6"></line><text fill="currentColor" x="-9" dy="0.32em">4</text></g>  
<g class="tick" opacity="1" transform="translate(0,140.5)"><line stroke="currentColor" x2="-6"></line><text fill="currentColor" x="-9" dy="0.32em">5</text></g>  
<g class="tick" opacity="1" transform="translate(0,120.5)"><line stroke="currentColor" x2="-6"></line><text fill="currentColor" x="-9" dy="0.32em">6</text></g>  
<g class="tick" opacity="1" transform="translate(0,100.5)"><line stroke="currentColor" x2="-6"></line><text fill="currentColor" x="-9" dy="0.32em">7</text></g>  
<g class="tick" opacity="1" transform="translate(0,80.5)"><line stroke="currentColor" x2="-6"></line><text fill="currentColor" x="-9" dy="0.32em">8</text></g>  
<g class="tick" opacity="1" transform="translate(0,60.500000000000001)"><line stroke="currentColor" x2="-6"></line><text fill="currentColor" x="-9" dy="0.32em">9</text></g>  
<g class="tick" opacity="1" transform="translate(0,40.499999999999999)"><line stroke="currentColor" x2="-6"></line><text fill="currentColor" x="-9" dy="0.32em">10</text></g>  
<g class="tick" opacity="1" transform="translate(0,20.499999999999996)"><line stroke="currentColor" x2="-6"></line><text fill="currentColor" x="-9" dy="0.32em">11</text></g>  
<g class="tick" opacity="1" transform="translate(0,0.5)"><line stroke="currentColor" x2="-6"></line><text fill="currentColor" x="-9" dy="0.32em">12</text></g></g>  
<circle cx="0" cy="80" r="5" stroke="#2222ff" fill="blue"></circle>  
<circle cx="100" cy="200" r="5" stroke="#2222ff" fill="blue"></circle>  
<circle cx="300" cy="0" r="5" stroke="#2222ff" fill="blue"></circle>  
</g>  
</svg>
```

```
a[0].x);  
a[0].x);
```

```
a[0].y);  
a[0].y);
```

```
'red'); })
```

```
'blue'); })
```

D3JS : C'est cool SVG mais...

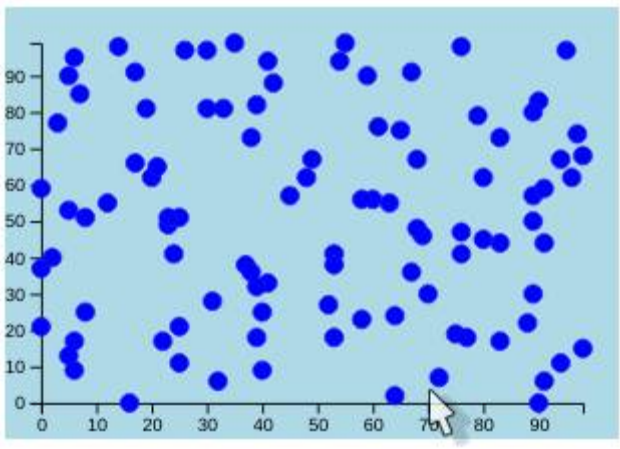
JAVASCRIPT CODE 1/2

```
// Recuperation du champ Body du DOM
var myBody = d3.select("body")

var graphWidth = 300;    // Largeur
var graphHeight = 200 ; // Hauteur

// definition de l'espace de dessin
// Ajout d'un champ svg au body
var mySVG = myBody.append("svg")
  .attr('width',    graphWidth + 40) // Largeur du SVG
  .attr('height',  graphHeight + 40) // Hauteur du SVG
  .style("background-color", '#ADD8E6') // Couleur
  .append("g") // Ajout d'un groupe
  .attr("transform", "translate(20,20)"); // On translate ce g
  // pour laisser une marge
```

OUTPUT



08/12/20 Franck Samsen LaMME

JAVASCRIPT CODE 2/2

```
var myData = [ ];
for (var i=0 ; i < 100 ; i++ ) {
  myData.push({ x : Math.floor(Math.random() * 100) ,
                y : Math.floor(Math.random() * 100)} ) ;
}

var minX = myData.reduce((min, b) => Math.min(min, b.x), myData[0].x);
var maxX = myData.reduce((max, b) => Math.max(max, b.x), myData[0].x);

var x = d3.scaleLinear() // creation de l'axe des X
  .domain([minX, maxX]) // Valeurs affichees
  .range([ 0, graphWidth ]); // Espace de travail

mySVG.append("g") // Ajout de l'axe au svg
  .attr("transform", "translate(0," + graphHeight + ")")
  .call(d3.axisBottom(x));

var minY = myData.reduce((min, b) => Math.min(min, b.y), myData[0].y);
var maxY = myData.reduce((max, b) => Math.max(max, b.y), myData[0].y);

var y = d3.scaleLinear() // creation de l'axe des Y
  .domain([minY, maxY]) // Valeurs affichees
  .range([ graphHeight, 0]); // Espace de travail

mySVG.append("g") // Ajout de l'axe au svg
  .call(d3.axisLeft(y));

// Creation des points du graph
mySVG
  .selectAll('circle') // Selection des cercles du SVG
  .data(myData) // en selectionnant mes donnees
  .enter() // Enter cree les rect qui n'ont
  // pas encore ete cree a partir des data
  .append("circle") // Ajouter des cercles SVG
  .attr("cx", function (d) { return x(d.x); } )
  .attr("cy", function (d) { return y(d.y); } )
  .attr("r", 5)
  .attr("stroke", "#2222ff")
  .attr("fill", "blue")
  .on("mouseover",
    function() { d3.select(this).attr('fill', 'red'); })
  .on("mouseout",
    function() { d3.select(this).attr('fill', 'blue');})
```

D3JS : C'est cool SVG mais...

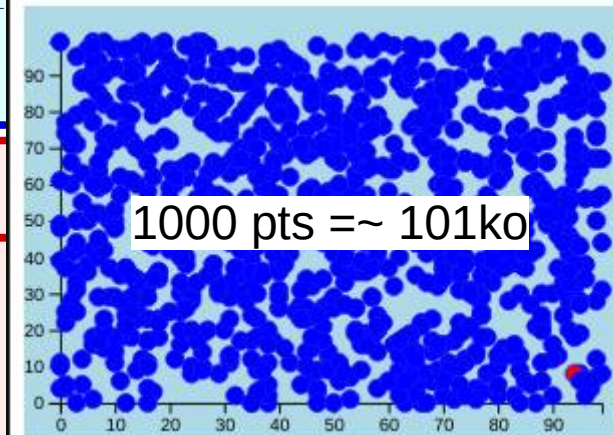
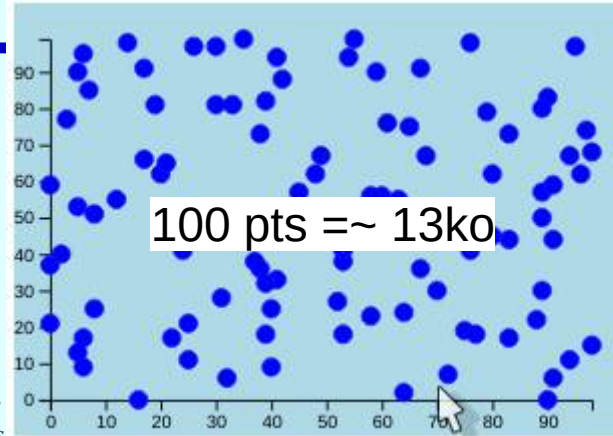
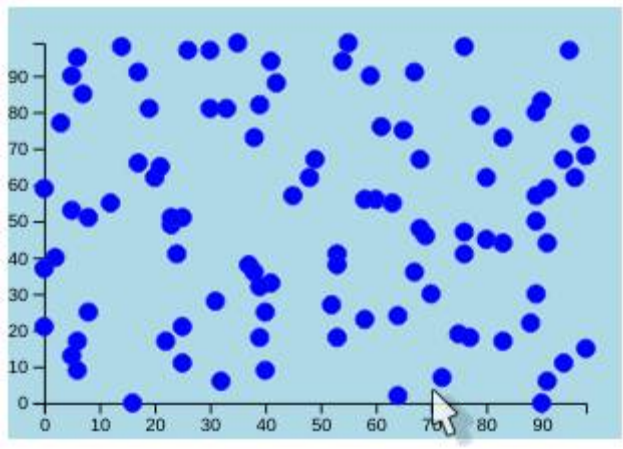
JAVASCRIPT CODE 1/2

```
// Recuperation du champ Body du DOM
var myBody = d3.select("body")

var graphWidth = 300; // Largeur
var graphHeight = 200 ; // Hauteur

// definition de l'espace de dessin
// Ajout d'un champ svg au body
var mySVG = myBody.append("svg")
  .attr('width', graphWidth + 40) // Largeur
  .attr('height', graphHeight + 40) // Hauteur
  .style("background-color", '#ADD8E6') // Couleur
  .append("g") // Ajout d'
  .attr("transform", "translate(20,20)"); // On trans
  // pour lai
```

OUTPUT



JAVASCRIPT CODE 2/2

```
); i-
Math
Math

n, b) =
x, b) =

// c
// Va
// Es
; // E

// i
ate(0, " + graphHeight + ")")

, b) => Math.min(min, b.y), myData[0].y);
, b) => Math.max(max, b.y), myData[0].y);

// creation de l'axe des Y
/ Valeurs affichees
/ Espace de travail

// Ajout de l'axe au svg

ph

ction des cercles du SVG
electionnant mes donnees

.enter() // Enter cree les rect qui n'ont
// pas encore ete cree a partir des data

.append("circle") // Ajouter des cercles SVG
  .attr("cx", function (d) { return x(d.x); } )
  .attr("cy", function (d) { return y(d.y); } )
  .attr("r", 5)
  .attr("stroke", "#2222ff")
  .attr("fill", "blue")
  .on("mouseover",
    function() { d3.select(this).attr('fill', 'red'); })
  .on("mouseout",
    function() { d3.select(this).attr('fill', 'blue');})
```

D3JS : SVG Limites

- D3js permet de faire des affichages de graph/courbes de manière très simple.
- Touche directement le DOM chaque graphe peut être très verbeux.
 - Il ne faut pas oublier que D3js génère le SVG mais le navigateur doit ensuite interpreter ce SVG pour le visualiser.
 - Plus il y aura d'objets plus l'affichage sera lent, pouvant même devenir rapidement bloquant pour votre application.
 - Possibilité d'ajouter des animations sur les graphes en utilisant des forces simulées comme la gravité, mais c'est tres rapidement limité en terme de performance.



D3JS : Passer au Canvas

- Le Canvas est une image dans laquelle on dessine directement.



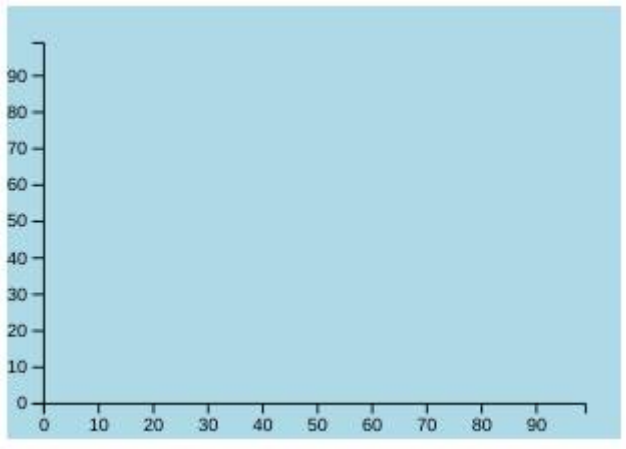
D3JS : SVG & Canvas

CSS CODE



```
.svg-plot, .canvas-plot {  
  position: absolute;  
}
```

OUTPUT



travailler sur SVG ou Canvas en D3.js

08/12/20 Franck Samsen - LaMME

JAVASCRIPT CODE

```
// Recuperation du champ Body du DOM  
var myBody = d3.select("body")  
  
var graphWidth = 300; // Largeur  
var graphHeight = 200; // Hauteur  
  
var myData = [ ]; // Generation de X pts  
for (var i=0 ; i < 100 ; i++ ) {  
  myData.push({ x : Math.floor(Math.random() * 100) ,  
                y : Math.floor(Math.random() * 100)});  
}  
  
// Recherche des min/max  
var minX = myData.reduce((min, b) => Math.min(min, b.x), myData[0].x);  
var maxX = myData.reduce((max, b) => Math.max(max, b.x), myData[0].x);  
var minY = myData.reduce((min, b) => Math.min(min, b.y), myData[0].y);  
var maxY = myData.reduce((max, b) => Math.max(max, b.y), myData[0].y);  
  
// definition de l'espace de dessin  
var mySVG = myBody.append("svg")  
  .attr('width', graphWidth + 40) // Largeur du SVG  
  .attr('height', graphHeight + 40) // Hauteur du SVG  
  .attr('class', 'svg-plot')  
  .style("background-color", '#ADD8E6') // Couleur  
  .append("g") // Ajout d'un groupe d'element SVG  
  .attr("transform", "translate(20,20)"); // On translate ce groupe, marge  
  
var x = d3.scaleLinear() // creation de l'axe des X  
  .domain([minX, maxX]) // Valeurs affichees  
  .range([ 0, graphWidth ]); // Espace de travail  
  
mySVG.append("g") // Ajout de l'axe au svg  
  .attr("transform", "translate(0," + graphHeight + ")")  
  .call(d3.axisBottom(x));  
  
var y = d3.scaleLinear() // creation de l'axe des Y  
  .domain([minY, maxY]) // Valeurs affichees  
  .range([ graphHeight, 0]); // Espace de travail  
  
mySVG.append("g") // Ajout de l'axe au svg  
  .call(d3.axisLeft(y));
```

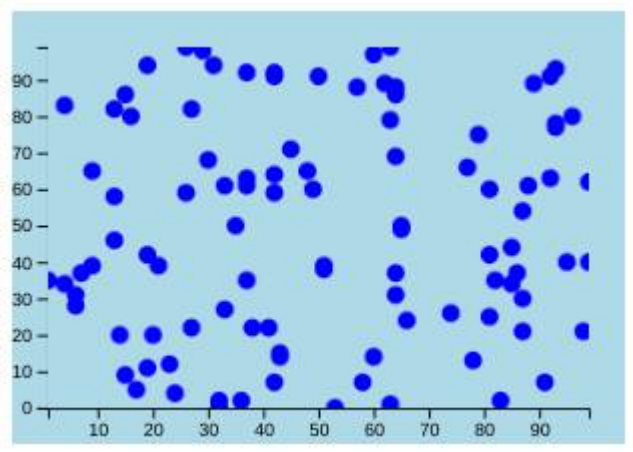
D3JS : SVG & Canvas

CSS CODE



```
.svg-plot, .canvas-plot {  
  position: absolute;  
}
```

OUTPUT



JAVASCRIPT CODE

```
....  
  
mySVG.append("g") // Ajout de l'axe au svg  
  .attr("transform", "translate(0," + graphHeight + ")")  
  .call(d3.axisBottom(x));  
  
var y = d3.scaleLinear() // creation de l'axe des Y  
  .domain([minY, maxY]) // Valeurs affichees  
  .range([ graphHeight, 0]); // Espace de travail  
  
mySVG.append("g") // Ajout de l'axe au svg  
  .call(d3.axisLeft(y));  
  
var myCanvas = myBody.append('canvas')  
  .attr('width', graphWidth ) // Largeur du Canvas  
  .attr('height', graphHeight ) // Hauteur du Canvas  
  .style("background-color", '#ADD8E6') // Couleur  
  .style('margin-left', 20 + 'px')  
  .style('margin-top', 20 + 'px')  
  .attr('class', 'canvas-plot');  
  
var context = myCanvas.node().getContext("2d");  
  
// Creation des points du graph  
var px, py;  
var angle = 2 * Math.PI;  
context.fillStyle = "blue";  
  
myData.forEach( point => {  
  context.beginPath();  
  px = x(point.x);  
  py = y(point.y);  
  context.arc(px, py, 5, 0, angle,true);  
  context.fill();  
});
```


D3JS : SVG vs Canvas

- Dessiner dans le canvas est extrêmement rapide
- Le DOM n'est pas touché, la taille ne change pas
- Utiliser le SVG pour les parties "fixes" du graphe, les axes par exemple
- Inconvénient :
 - Download de l'image plus complexe
 - Gestion des évènements cauchemardesque!!
 - En cas de zoom tout doit être redessiné

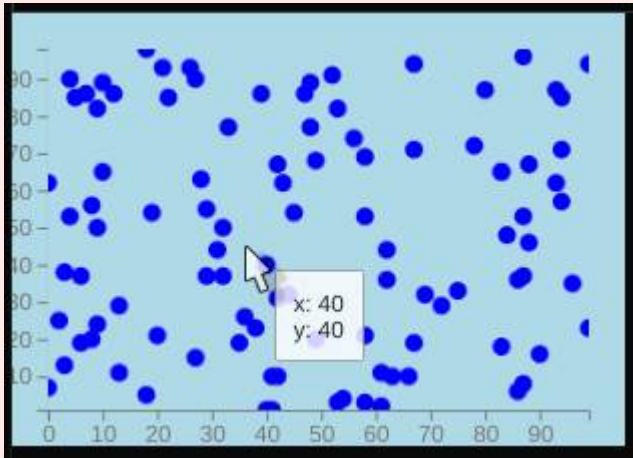


D3JS : Canvas & Events

CSS CODE

```
.svg-plot, .canvas-plot {
  position: absolute;
}
/* Ajouter : <div id="tooltip"></div>
Dans le HTML ! */
div#tooltip {
  position: absolute;
  display: inline-block;
  padding: 10px;
  font-family: 'Open Sans' sans-serif;
  color: #000;
  background-color: #fff;
  border: 1px solid #999;
  border-radius: 2px;
  pointer-events: none;
  opacity: 0;
  z-index: 1;
}
```

OUTPUT



tutoriel sur SVG ou Canvas en D3.js

08/12/20 Franck Samsen - LaMME

JAVASCRIPT CODE

```
myCanvas.on('mousemove', checkPoint);

function checkPoint(e) {
  var node = -1;
  var posX = e.clientX - 20;
  var posY = e.clientY - 20;
  var realPosX = x.invert(posX);
  var realPosY = y.invert(posY);

  var minDistance = Infinity;

  myData.forEach( point => {
    var dx = point.x - realPosX;
    var dy = point.y - realPosY;
    var distance = Math.sqrt((dx * dx) + (dy * dy));
    if (distance < m.inDistance && distance < 5) {
      minDistance = distance;
      node = point;
    }
  });
  if (node !== -1) {
    d3.select('#tooltip')
      .style('opacity', 0.8)
      .style('top', posY + 25 + 'px')
      .style('left', posX + 25 + 'px')
      .html('x: ' + node.x + '<br>' + 'y: ' + node.y + '<br>');
  }
  else {
    d3.select('#tooltip')
      .style('opacity', 0);
  }
}
```

D3JS : Alors SVG ou Canvas ?

- SVG : simple rapide à mettre en place gestion des évènements ultra simple
- Canvas : légèrement plus complexe, moins dans l'esprit jquery, nettement plus rapide, mais très complexe pour les évènements sur les objets graphiques
- Il n'y a pas vraiment de solution idéale il faut trouver un bon compromis entre rapidité d'exécution et facilité d'utilisation...
- Exemple d'utilisation de SVG et Canvas :
 - <https://www.vendeeglobe.org/fr/cartographie>

