Visualisation de données

05 - Échelles et axes

1 mars 2024

Noemi Romano
noemi.romano@heig-vd.ch

Cours précédent

Cours précédent

Charger les données `json()`, `csv()`

Cours précédent

Charger les données `json()`, `csv()`

Statistiques `mean()`, `max()`, `extent()` etc.

Cours précédent

Charger les données `json()`, `csv()`

Statistiques `mean()`, `max()`, `extent()` etc.

Manipuler les données `map()`, `filter()`,`reduce()` etc.

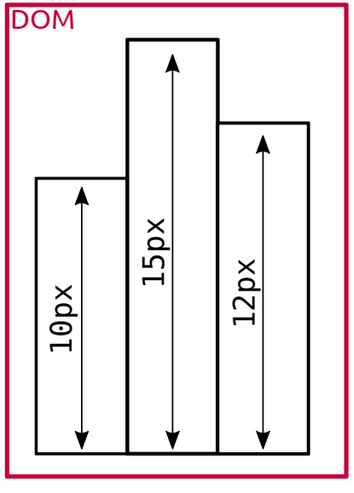
Jusqu'à maintenant

`Données = Pixels`

Jusqu'à maintenant

Données = Pixels

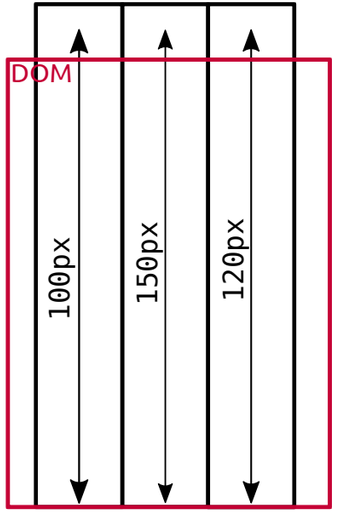
```
const data = [10,15,12]
```



Jusqu'à maintenant

Données = Pixels

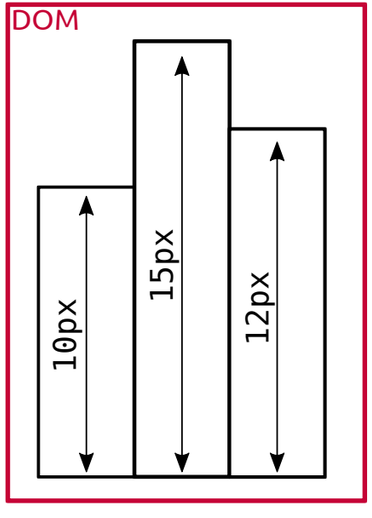
```
const data = [100,150,120]
```



Dès aujourd'hui

Données = Echelle * Pixels

```
const data = [100,150,120]
const scale = 0.1
```



d3-scale

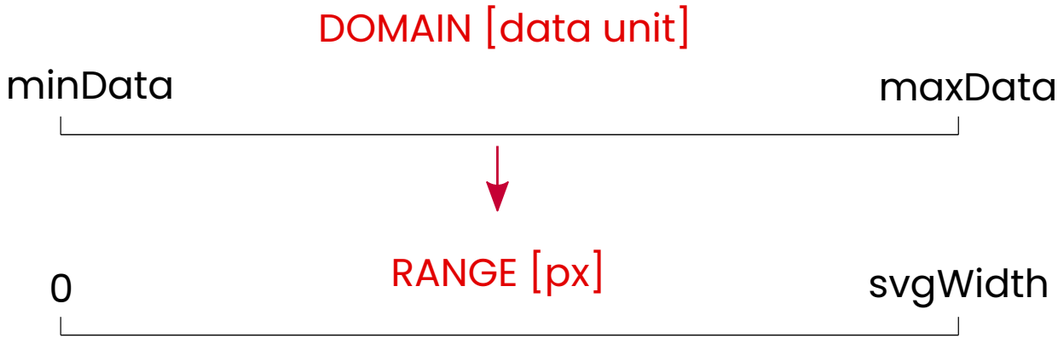
Installation

```
npm install d3-scale
```

Syntaxe

```
typeEchelle
.domain([minData, maxData])
.range([0, svgWidth])
```

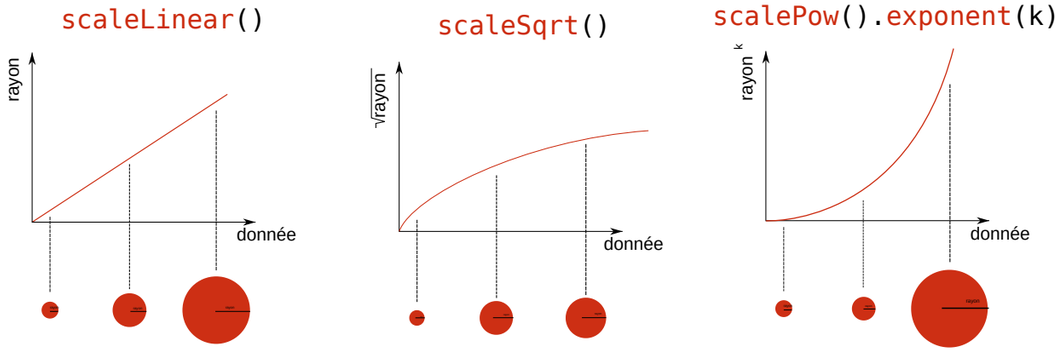
- **`typeEchelle`**: La fonction qui détermine quel type d'échelle sera utilisé, tels que linéaire, quadratique, exponontielle etc.
- **`domain([minData, maxData])`**: Définit l'intervalle des données en entrée, avec `minData` et `maxData`.
- **`range([0, svgWidth])`**: Définit l'intervalle de sortie, la plage de valeurs souhaitée, avec `0` comme début et `svgWidth` comme fin.



↳ D3 - d3-scale

Échelles continues

Données continues



JS index.js

```
const linearScale = d3.scaleLinear()
  .domain([0, 100])  // Input data range
  .range([0, 500]);   // Output range

console.log(linearScale(50));  // Output: 250

const sqrtScale = d3.scaleSqrt()
  .domain([0, 100])  // Input data range
  .range([0, 500]);   // Output range

console.log(sqrtScale(25));  // Output: ~111.8

const powScale = d3.scalePow()
  .exponent(2)        // Exponent value (power)
  .domain([0, 100])    // Input data range
  .range([0, 500]);    // Output range

console.log(powScale(5));  // Output: 25
```

↳ D3 - d3-scale

Échelles continues

Données temporelles

`scaleTime()`

`[dateDebut, dateFin]` ⟶ `[0 , svgWidth]`

domain — range

JS index.js

```js
const firstDay = new Date(new Date().getFullYear(), 0, 1);
const lastDay = new Date(new Date().getFullYear(), 11, 31);
const christmasDay = new Data(new Date().getFullYear(), 11, 25);

const timeScale = scaleTime
                .domain([firstDay, lastDay])
                .range([0, 100])


timeScale(firstDay) // Expected output 0
timeScale(lastDay)  // Expected output 100
timeScale(christmasDay) // Expected output 98.35164835164835
```

↳ D3 - d3-scale

Échelles discrètes

Données continues

`scaleQuantize()`

JS index.js

```js
const quantizeScale = scaleQuantize
                              .domain([0, 100])
                              .range(["white", "red", "blue"])


quantizeScale(0) // Expected output white
quantizeScale(50)  // Expected output red
quantizeScale(-10) // Expected output white
```

↳ D3 - d3-scale

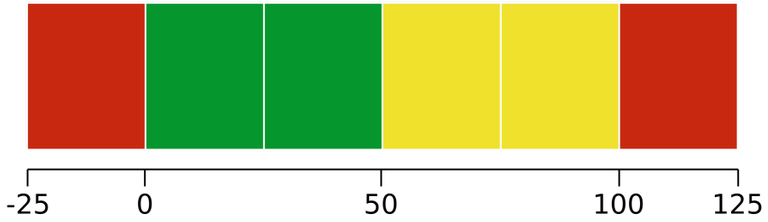
Échelles discrètes

Données continues

`scaleThreshold()`

`[0 , 50 , 100]` → `['red' , 'green' , 'yellow', 'red']`

domain *N* — range *N+1*



index.js

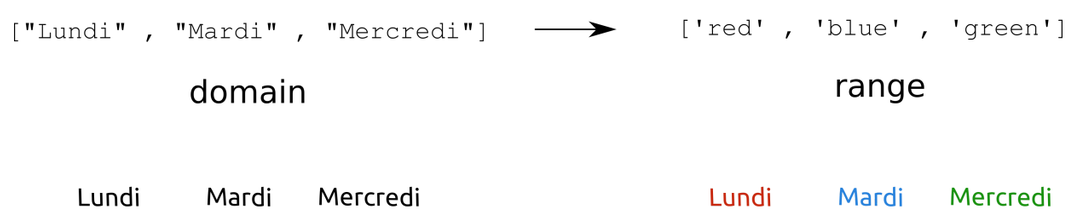
```javascript
const thresholdScale = d3
  .scaleThreshold()
  .domain([30, 60, 90]) // Threshold values
  .range(['red', 'yellow', 'green', 'blue']); // Corresponding color range

console.log(thresholdScale(20)); // Output: red
console.log(thresholdScale(45)); // Output: yellow
console.log(thresholdScale(75)); // Output: green
console.log(thresholdScale(100)); // Output: blue
```

↳ D3 - d3-scale

Échelles discrètes

Données discrètes

`scaleOrdinal()`



JS index.js

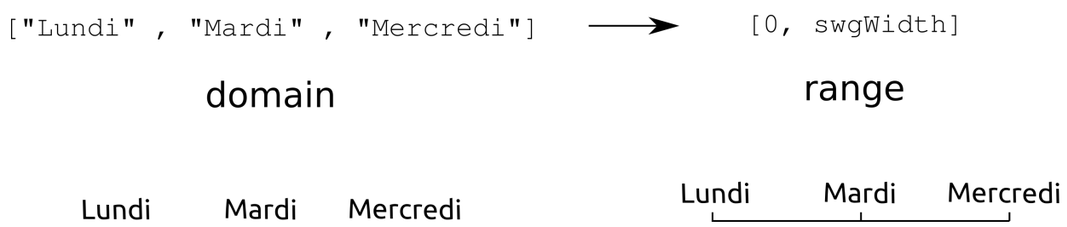
```js
const ordinalScale = d3.scaleOrdinal()
  .domain(['apple', 'banana', 'cherry'])
  .range(['red', 'yellow', 'pink']);

console.log(ordinalScale('banana'));  // Output: yellow
console.log(ordinalScale('cherry'));  // Output: pink
console.log(ordinalScale('apple'));   // Output: red
```

↳ D3 - d3-scale

Échelles discrètes

Données discrètes

`scaleBand()`



JS index.js

```javascript
// Sample data representing categories
const categories = ['Category A', 'Category B', 'Category C', 'Category D'];

// Setting up the scale
const bandScale = d3.scaleBand()
  .domain(categories)
  .range([0, 300])  // Width of the band scale
  .padding(0.1);    // Padding between bands (0.1 means 10% of the band width)

// Using the scale
console.log(bandScale('Category A'));  // Output: 0
console.log(bandScale('Category B'));  // Output: 75
console.log(bandScale('Category C'));  // Output: 150
console.log(bandScale('Category D'));  // Output: 225
```

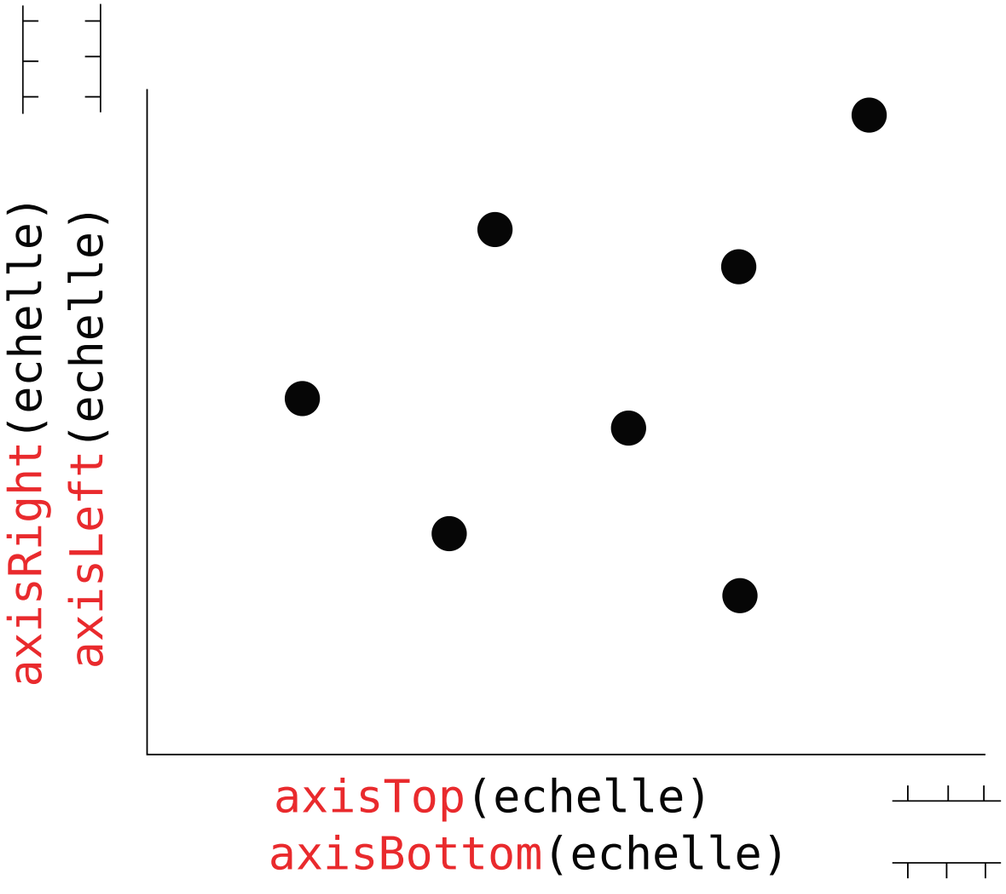
↳ D3 - d3-scale

d3-axis

Installation

```
npm install d3-axis
```

Axes


axisRight(echelle)
axisLeft(echelle)
axisTop(echelle)
axisBottom(echelle)

Axes

Créer

```
const axe = axisBottom(echelle)
```

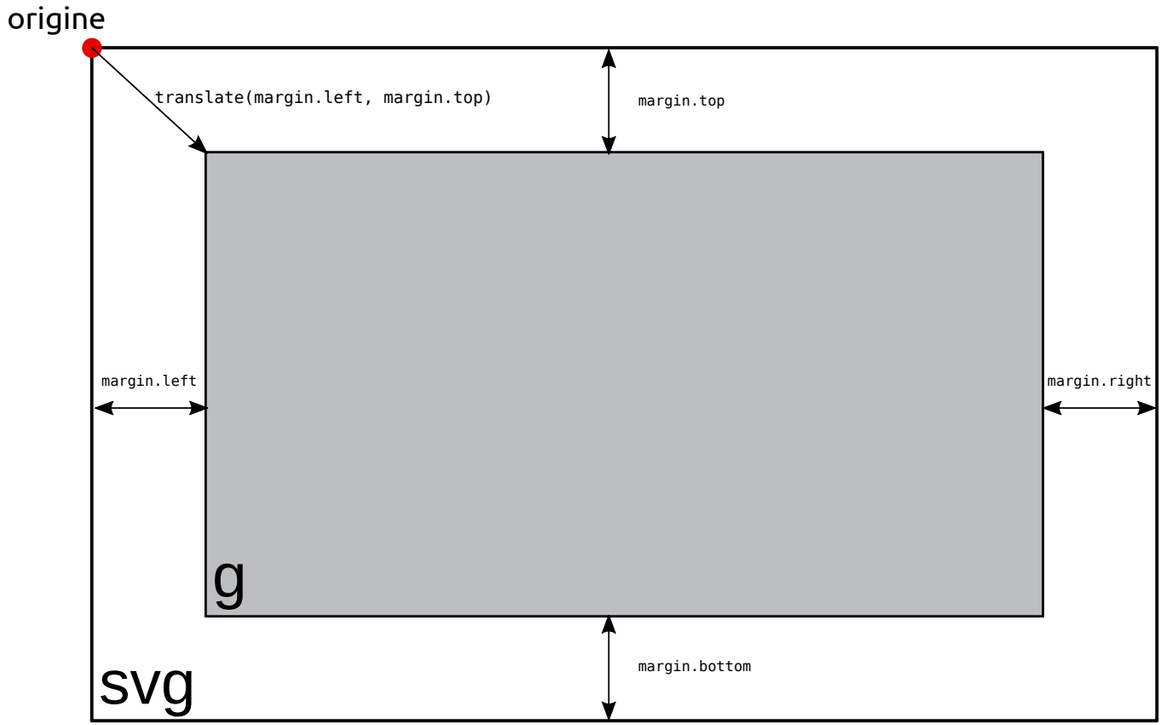
- axisBottom: Fonction de D3 qui crée un axe horizontal en bas.
- echelle: L'échelle à associer à l'axe.

Dessiner

```
selecteur.append("g").call(axe)
```

- selecteur: Le sélecteur représentant l'élément auquel l'axe sera ajouté.
- g: L'élément "g" (groupe) SVG qui servira de conteneur pour l'axe.
- call: Une méthode D3 qui invoque une fonction (dans ce cas, l'axe) sur l'élément sélectionné.
- axe: L'axe créé avec `const axe = axisBottom(echelle)`, prêt à être dessiné.

Marges & Translations



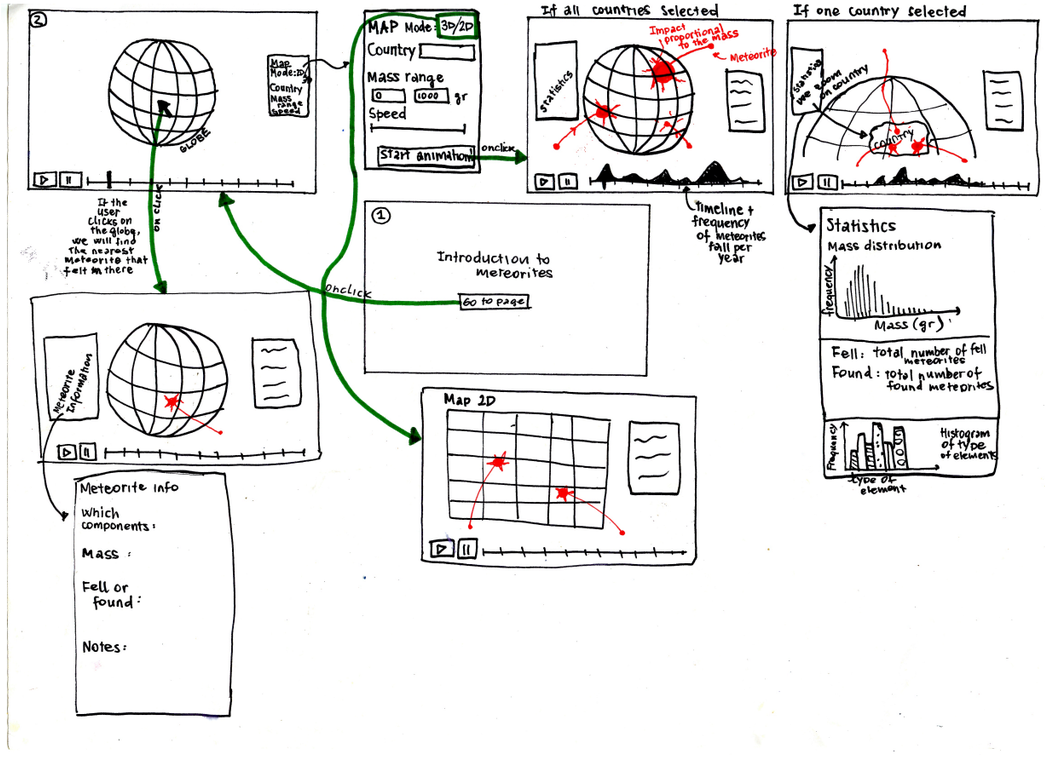
Exemples

↳ Observable: d3-scale, d3-axis

Projet

Thématique (8 mars)

Wireframe (15 mars)



Figma

invision