INTRODUCTION À D3.JS

ATELIER THÉMATIQUE EN VISUALISATION DE DONNÉES

Antoine Béland

9 octobre 2018

QUI SUIS-JE?

- Baccalauréat en génie logiciel
- Maîtrise recherche en génie informatique (en cours)
- Spécialisation en visualisation de données
- Collaboration avec « Le Devoir »

QUI ÊTES-VOUS?

- Expérience en programmation?
- Expérience en JavaScript?
- Expérience avec D3.js?

OBJECTIFS DES ATELIERS PRATIQUES

1. Se familiariser avec les fonctionnalités de D3.js
2. Connaître les possibilités qu'offre D3.js
3. Être en mesure de réaliser une visualisation de base avec D3.js

PLAN DE L'APRÈS-MIDI

1. Qu'est-ce que D3.js?
2. Exemples d'application
3. Introduction aux langages web
4. Introduction aux fonctionnalités de D3.js
5. Mise en pratique

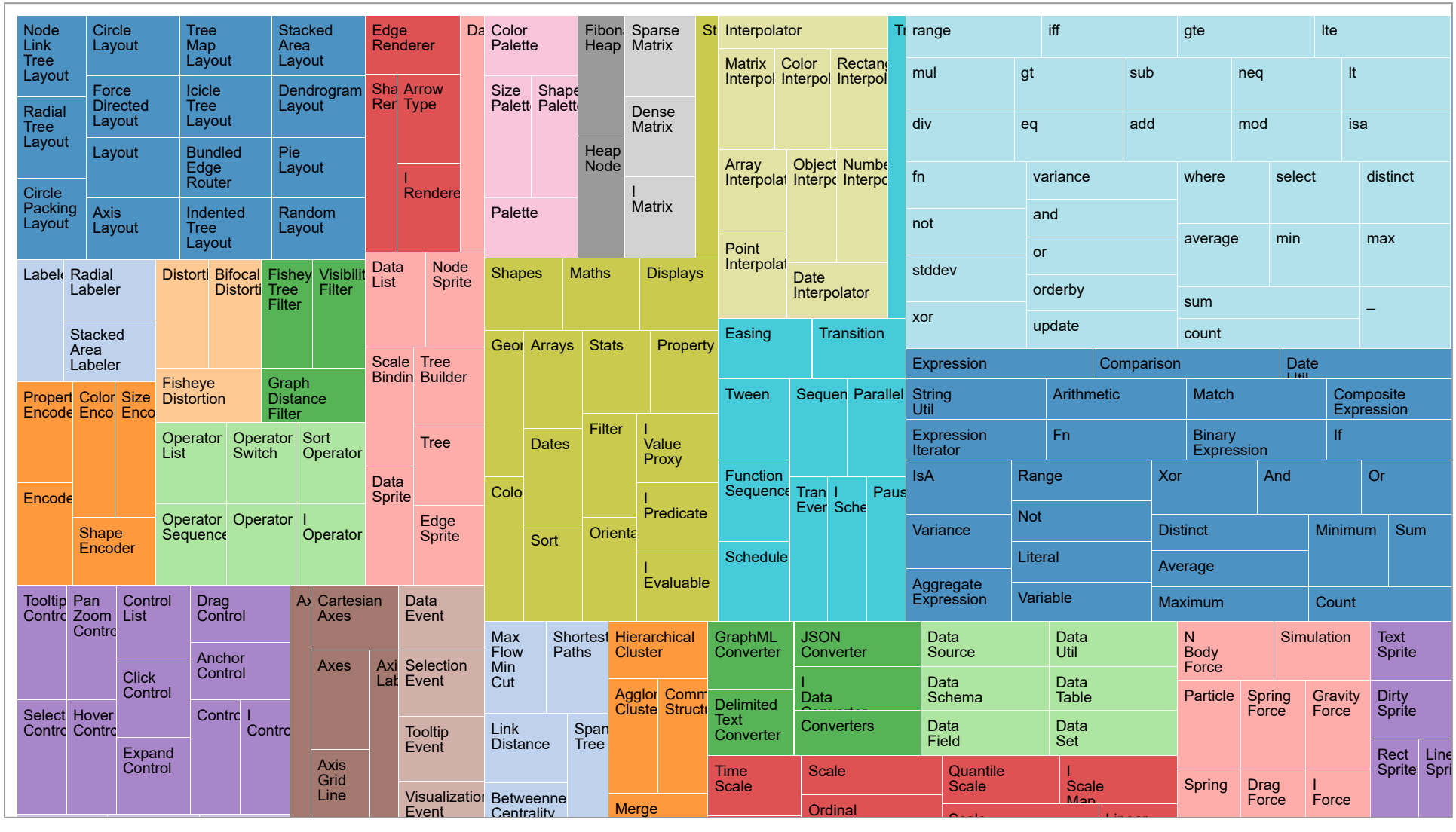
QU'EST-CE QUE D3.JS?

- Bibliothèque JavaScript permettant de réaliser des visualisations sur le web
- Version initiale en 2011, développée par Mike Bostock
- Fournie une panoplie de fonctions utiles pour faciliter la création d'une visualisation
- Permet de **personnaliser** grandement le rendu

Pour l'atelier, nous utiliserons la **version 5** de D3.js.

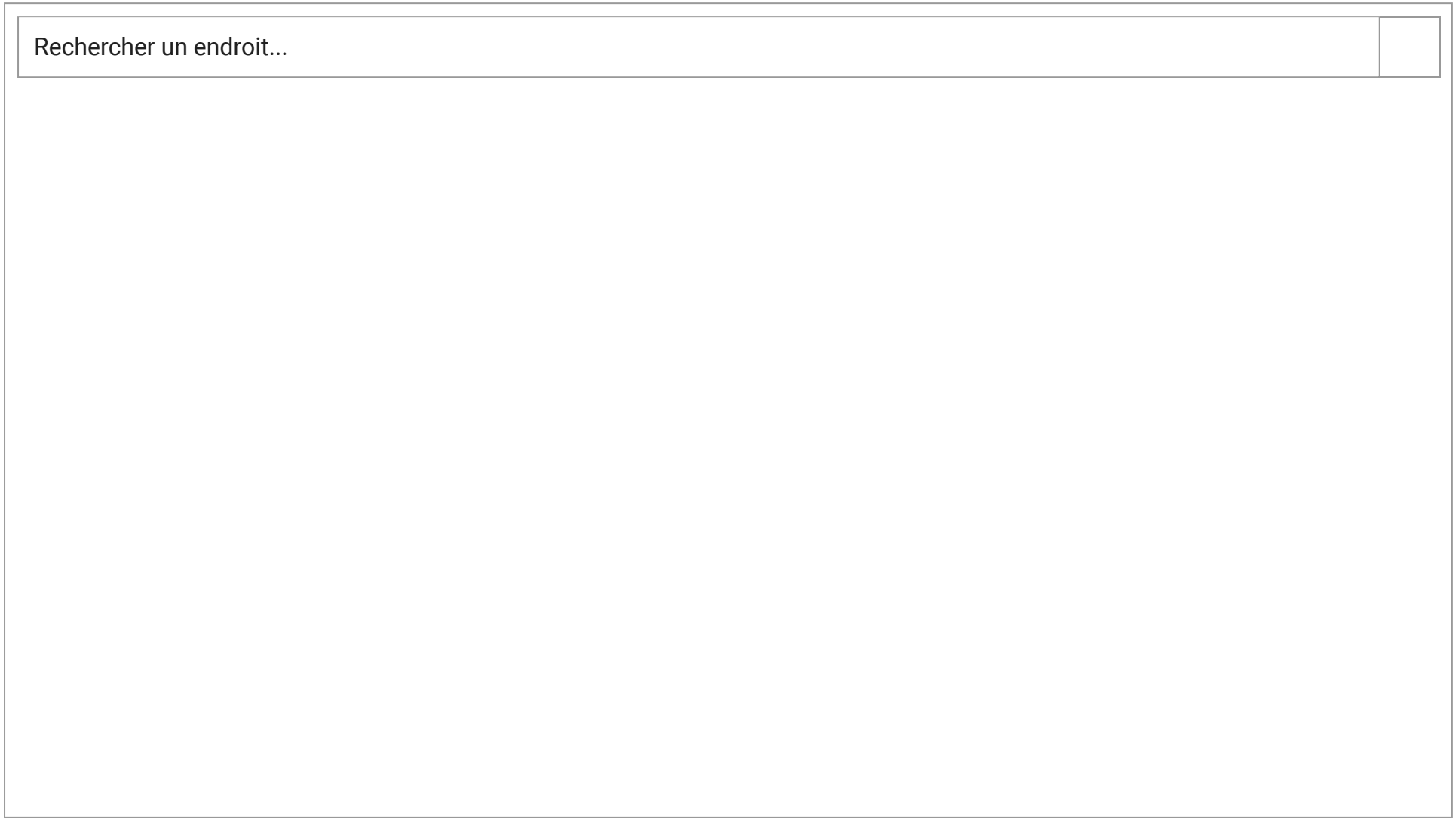
EXEMPLES D'APPLICATION

EXEMPLE 1: CARTE PROPORTIONNELLE



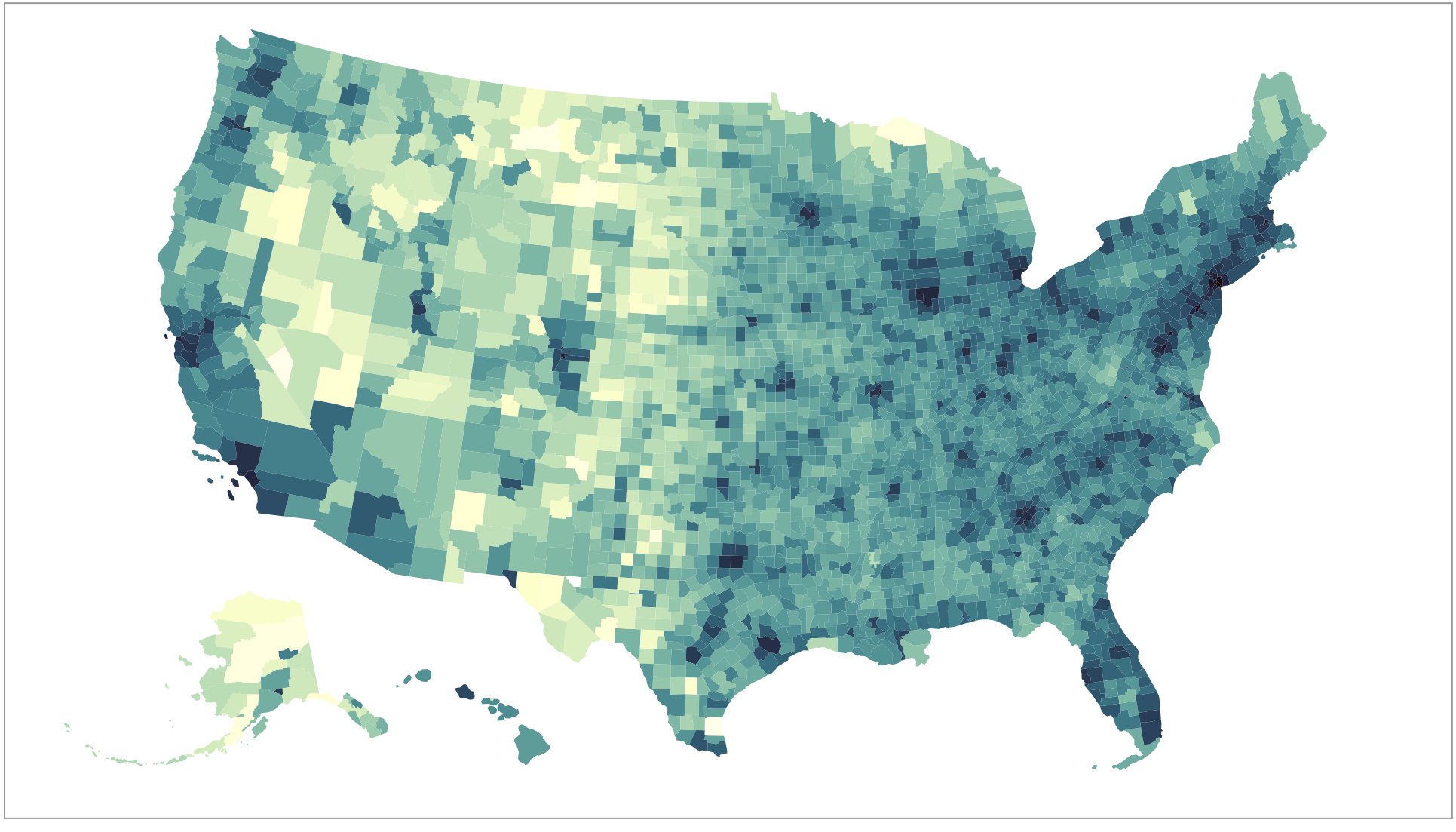
Treemap · Mike Bostock, 2018

EXEMPLE 2: DIAGRAMME DE FLUX



Navettage à partir du lieu de résidence au lieu de travail · Antoine Béland, 2017

EXEMPLE 3: CHOROPLÈTHE



Population Choropleth · Mike Bostock, 2016

INTRODUCTION AUX LANGAGES WEB

HTML

- Langage de balisage permettant de définir la structure d'une page web
- Chacun des éléments est représenté par une balise d'un certain type (`h1`, `p`, etc.)
- Il est possible d'ajouter des attributs sur les balises (`id`, `class`, etc.)

HTML — EXEMPLE D'UNE PAGE

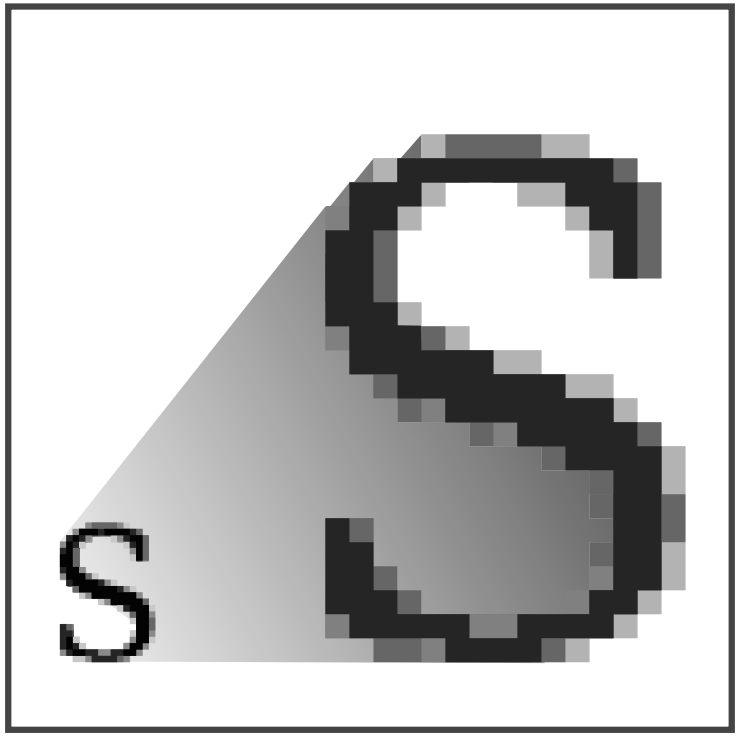
```
<!DOCTYPE html>
<html>
<body>
  <h1>Le titre de ma page</h1>
  <p>Un <strong>paragraphe</strong> associé à ma page.</p>
</body>
</html>
```

▶ Démo

HTML — BALISE SVG

- Permet de dessiner des éléments vectoriels sur une page web
- Balise **très importante** pour D3.js
- Non destructif aux agrandissements

HTML — MATRICIEL VS SVG



Matriciel

.jpeg .gif .png

Vectoriel

.svg

Différence conceptuelle entre les images matricielles et vectorielles · Yug, 2011 | CC BY-SA 2.5

HTML — ÉLÉMENTS VECTORIELLES

- **`circle`** (attributs: `cx`, `cy`, `r`)
- **`ellipse`** (attributs: `cx`, `cy`, `rx`, `ry`)
- **`polygon`** (attribut: `points`)
- **`rect`** (attributs: `x`, `y`, `width`, `height`)
- **`text`** (attributs: `x`, `y`)
- etc.

HTML — EXEMPLE D'UNE BALISE SVG

```
<svg width="300" height="200">
  <!-- Un rectangle -->
  <rect width="100" height="80" x="0" y="70" fill="green"/>

  <!-- Une ligne -->
  <line x1="5" y1="5" x2="250" y2="95" stroke="red"/>

  <!-- Un cercle -->
  <circle cx="90" cy="80" r="50" fill="blue"/>

  <!-- Un texte -->
  <text x="180" y="60">Un texte</text>
</svg>
```

▶ Démo

HTML — POUR EN SAVOIR EN PLUS

- Tutoriel sur le langage HTML
- Tutoriel sur les images SVG
- Référence sur les balises HTML
- Référence sur les éléments SVG

CSS

- Langage permettant de décrire la présentation d'un document HTML
- Applique un style particulier en utilisant des **règles** et des **sélecteurs**

CSS — RÈGLES

Durant l'atelier, nous utiliserons principalement:

- `fill`: couleur de remplissage
- `height`: hauteur d'un élément
- `stroke`: couleur de la bordure
- `stroke-width`: taille de la bordure
- `width`: largeur d'un élément

CSS — SÉLECTEURS

Pour appliquer un style, il faut utiliser un sélecteur:

- `element`: applique un style à tous les éléments sélectionnés
- `.classe`: applique un style à tous les éléments ayant la classe sélectionnée
- `#identifiant`: applique un style à l'élément ayant l'identifiant sélectionné

CSS — EXEMPLE

```
line {
  stroke: green;
  stroke-width: 4px;
}

.rouge {
  fill: red;
}

#rectangle {
  fill: yellow;
  stroke: blue;
  stroke-width: 5px;
}
```

▶ Démo

CSS — POUR EN SAVOIR EN PLUS

- Tutoriel sur le langage CSS

JAVASCRIPT

- Langage de programmation de script faiblement typé
- Permet de manipuler une page web, soit le DOM (*Document Object Model*)
- Permet l'interaction avec l'utilisateur via des évènements

JAVASCRIPT — SYNTAXE

```
var variable = 1;                    // Déclaration d'une variable
var liste = [ 1, 2, 3, 4, 5 ];       // Déclaration d'une liste

var obj = {                          // Déclaration d'un objet
  prenom: 'Antoine',
  nom: 'Béland'
};

function fct1() { /* ... */ }        // Déclaration d'une fonction
const fct2 = () => {}                // Déclaration d'une fonction

if (variable === 1) {                // Condition
  console.log('OUI');
}
```

▶ Démo

JAVASCRIPT — TRUCS ET ASTUCES

- Utiliser la **console web** pour votre débogage (`console.log`)
- Utiliser l'**inspecteur du DOM** du navigateur

JAVASCRIPT — POUR EN SAVOIR PLUS

- Tutoriel sur le langage JavaScript
- Guide sur la manipulation de tableaux en JavaScript

INTRODUCTION AUX FONCTIONNALITÉS DE D3.JS

D3.JS — À SAVOIR

- L'appel aux fonctions de D3.js commence toujours par l'appel à l'objet **d3**
- Il est possible d'effectuer un chainage de fonctions:

```
d3.fonction1()
  .fonction2()
  .fonction3();
```

D3.JS — SÉLECTION D'ÉLÉMENTS

- Élément fondamental de la bibliothèque
- Permet de sélectionner un ou plusieurs éléments dans le document HTML
- On utilise la fonction **`d3.select`** ou la fonction **`d3.selectAll`**
- Les fonctions de sélection prennent en paramètre un **sélecteur CSS** (élément, classe ou identifiant)

D3.JS — MODIFICATION D'UNE PROPRIÉTÉ

- Une fois une sélection effectuée, il est possible de modifier des propriétés sur les éléments
- On utilise la fonction **`attr`** pour modifier la valeur d'un attribut
- On utilise la fonction **`style`** pour modifier un élément de style

D3.JS — EXEMPLE (SÉLECTION SIMPLE)

```
d3.select('line')
  .attr('x1', 50)                 // Modification de la position X
  .attr('y1', 150)                // Modification de la position Y
  .style('stroke', 'purple'); // Modification de la couleur
```

▶ Démo

D3.JS — EXEMPLE (SÉLECTION MULTIPLE)

```
d3.selectAll('.rouge')
  .style('fill', 'gray'); // Modification de la couleur
```

▶ Démo

D3.JS — CRÉATION D'UN ÉLÉMENT

- Une fois une sélection effectuée, il est possible d'ajouter de nouveaux éléments
- On utilise la fonction **append** pour créer un nouvel élément dans l'élément sélectionné

D3.JS — EXEMPLE (CRÉATION)

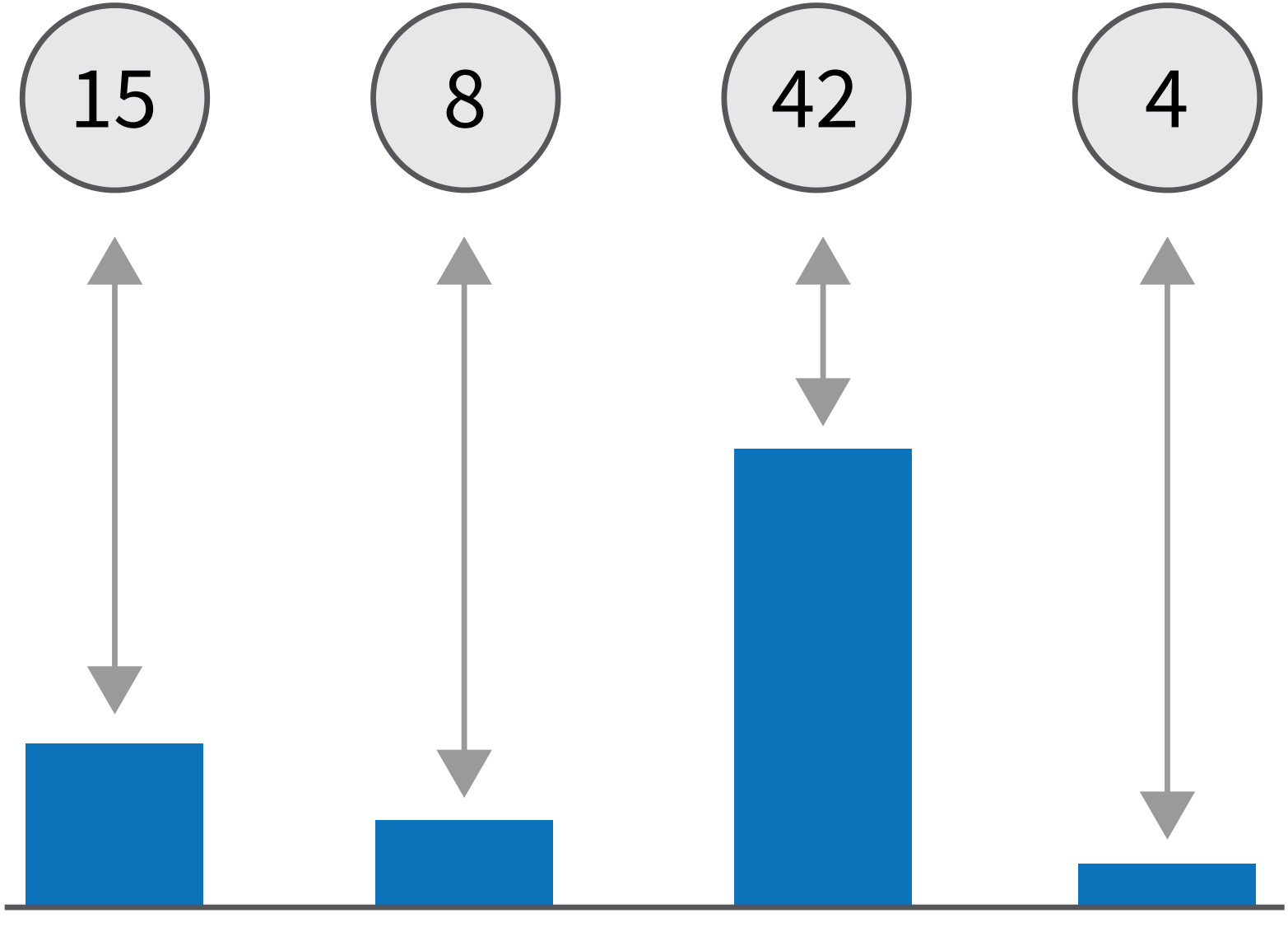
```
d3.select('svg')
  .append('rect') // Création d'un rectangle
  .attr('x', 10)
  .attr('y', 10)
  .attr('width', 50)
  .attr('height', 50)
  .style('fill', 'red');
```

▶ Démo

D3.JS — ASSOCIATION DE DONNÉES

- L'association de données (*data binding*) est un autre concept fondamental à comprendre
- Permet de lier des données aux éléments d'une sélection
- On utilise la fonction **`data`** sur une **sélection multiple** pour y associer un tableau de données

D3.JS — ASSOCIATION DE DONNÉES



D3.JS — ASSOCIATION DE DONNÉES

```
const svg = d3.select('svg');    // Élément de base

svg.selectAll('rect')            // Sélection multiple
  .data([ 15, 8, 42, 4 ])        // Association des données
  .enter()                       // Doit créer de nouveaux élém
  .append('rect');               // Création d'un rectangle
  .attr('width', d => d);        // Utilise les données courant
```

- La fonction **`selectAll`** peut être vue comme une boucle qui utilise les données de **`data`**
- Pour chacun des éléments à créer, les données courantes sont spécifiées (d)

D3.JS — EXEMPLE (ASSOCIATION)

```
const data = [ 100, 50, 75 ];

d3.select('svg')
  .selectAll('rect')              // Sélection multiple
  .data(data)                     // Association des données
  .enter()
  .append('rect')
  .attr('x', (d, i) => i * 50) // i représente l'index courant
  .attr('y', d => 100 - d)
  .attr('width', 50)
  .attr('height', d => d)
  .style('fill', 'red');
```

▶ Démo

D3.JS — ÉVÈNEMENTS

- Lorsqu'un élément est sélectionné, il est possible de lui associer des évènements.
- On utilise la fonction **on** pour y associer un évènement particulier

D3.JS — ÉVÈNEMENTS

- `click`: survient lorsque l'élément est cliqué
- `mouseenter`: survient lorsque la souris vient de commencer à survolé l'élément
- `mouseleave`: survient lorsque la souris vient de finir de survoler l'élément
- etc.

D3.JS — EXEMPLE (ÉVÈNEMENTS)

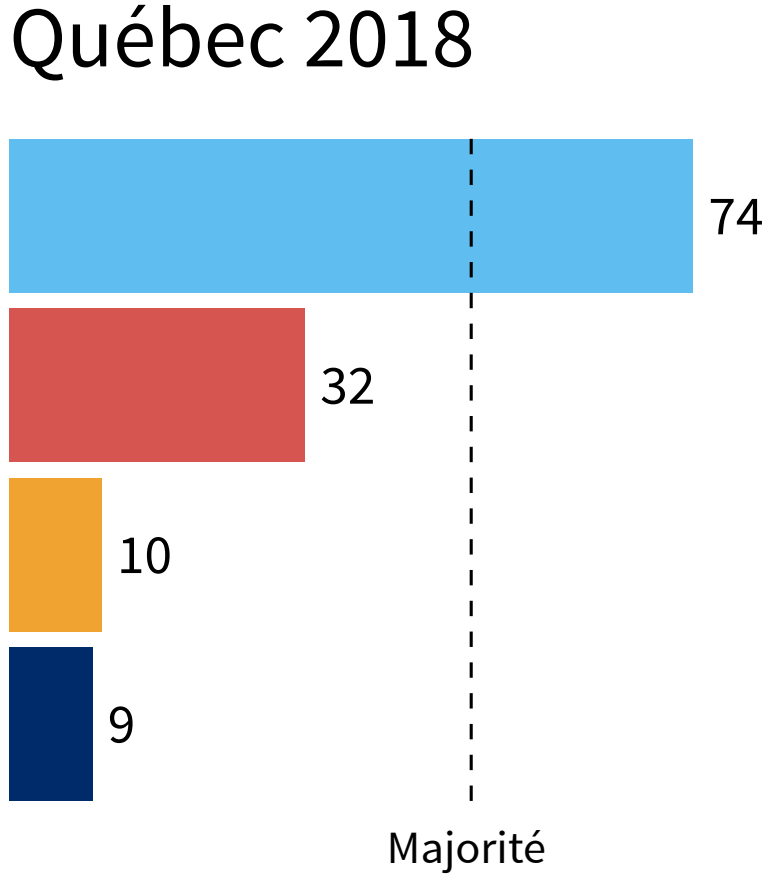
```
const colors = [ 'yellow', 'blue', 'green' ];
d3.select('svg')
  .selectAll('rect')
  .data([ 100, 50, 75 ])
  .enter()
  .append('rect')
  .attr('x', (d, i) => i * 50)
  .attr('y', d => 100 - d)
  .attr('width', 50)
  .attr('height', d => d)
  .style('fill', 'red')
  .on('click', (d, i, elements) => {
    d3.select(elements[i]).style('fill', colors[i]);
  });
```

▶ Démo

MISE EN PRATIQUE

MISE EN PRATIQUE

- Réaliser un ***bar chart* horizontal** à partir des données sur le nombre de députés élus par parti



MISE EN PRATIQUE — DIRECTIVES

- La longueur maximale d'une barre doit être de **300 px** (si 100% des députés élus)
- La hauteur de chacune des barres doit être de **50 px**
- Un espacement de **5 px** doit être présent entre les barres
- Une ligne pointillée doit indiquer une majorité

MISE EN PRATIQUE

- Pour débuter, téléchargez le dossier ZIP contenant le code de départ pour l'exercice
- Complétez, par la suite, le fichier **script.js**

Télécharger ▶ Correction