



POLYTECHNIQUE  
MONTRÉAL

LE GÉNIE  
EN PREMIÈRE CLASSE

# Guide TP4

INF8808

Version JavaScript

Cahier *Observable*

# Cahier Observable

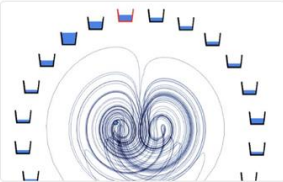
- *Observable* est un site web permettant de créer des cahiers contenant du code JavaScript
- Il est souvent utilisé en visualisation de données avec D3
  - *Observable* est créé par Mike Bostock, créateur de D3

**Observable** Explore Learn Community Search New


Trending Recent Most liked

### Trending


Showing 1-30 of 120 notebooks



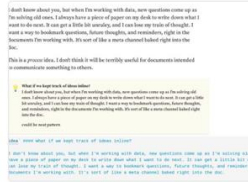
**Malkus Waterwheel**  
Ricky Reusser  
Feb 13 · ♥ 26



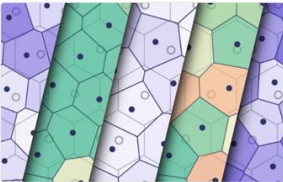
**Pretty Printing**  
Dylan Freedman  
Feb 15 · ♥ 6



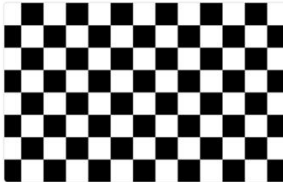
**Heart** ♥  
Mike Bostock  
Feb 14 · ♥ 13



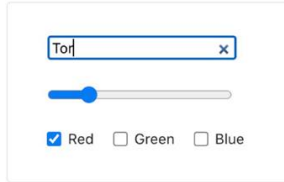
**Thought Process**  
Zeke Nierenberg  
Feb 13 · ♥ 17 15 13




**Lloyd's Algorithm Regular Tilings**  
Matt Dzuqan



**Shader**  
Mike Bostock



**Observable Inputs**  
Mike Bostock in Observable



**Static Site Generator (Netlify)**  
Tom Larkworthy in Endoint Serv

# Cahier Observable

## Étapes pour remplir le cahier

1. Créez un compte sur <https://observablehq.com/>
2. Aller sur le lien du cahier pour le TP :
  - <https://observablehq.com/d/836639e52760f996>
3. Utilisez le bouton « fork » pour créer votre propre copie du cahier
4. Remplissez les sections demandées directement

**Les données sont disponibles comme ceci dans le cahier :**

```
countries = ▶ Object {2000: Array(174), 2015: Array(174)}
```

```
countries = FileAttachment("countriesData.json").json()
```

# Cahier Observable

## « Fork » du cahier



Olivia Gelinas

Link shared Sep 4 Fork of TP4 | INF8808 (Solution) · 1 fork · 1 file

### TP4 | INF8808

In this notebook, we will start by creating a simple app to help explore our data. This step is important in conceiving data visualizations. It helps to determine which type of data visualization is appropriate for our data set, as well as which features we'd like to include.

In the rest of this notebook, you will have to complete some cells to prepare for the implementation of an animated bubble chart in the next steps.

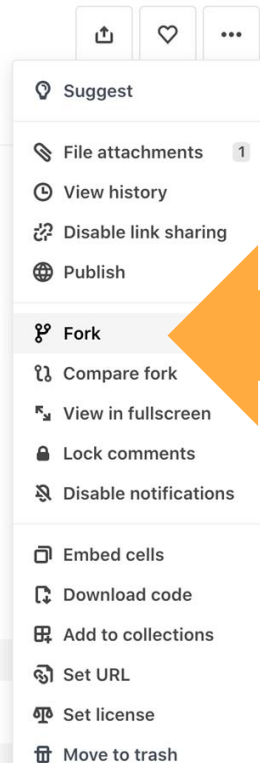
To begin, take a look at the following cells, where the final result will be displayed. Note that typically code is written from bottom to top in Observable notebooks. Thus, it may be helpful to answer the questions in this notebook in reverse order.

undefined

```
draw(2015, svg2015, xScale(), yScale())
```

undefined

```
draw(2000, svg2000, xScale(), yScale())
```

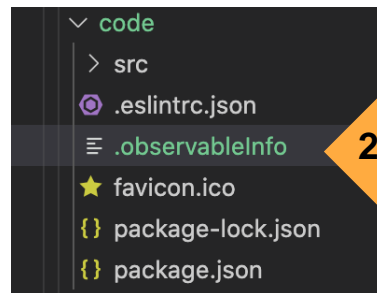
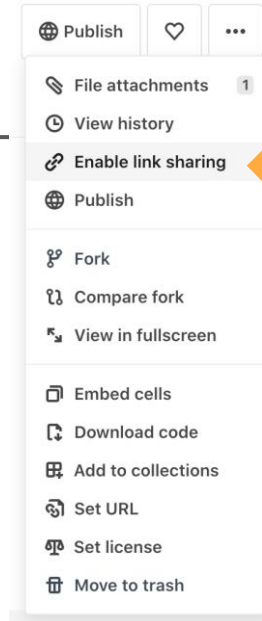


# Cahier Observable

## Soumission du cahier

1. Donner les permissions de «link sharing» au cahier, pour que les chargés puisse y avoir accès
1. Copier/coller l'URL du cahier dans le fichier « .observableInfo » dans le dossier du TP

**Important : NE PAS PUBLIER LE CAHIER PUBLIQUEMENT!!!**



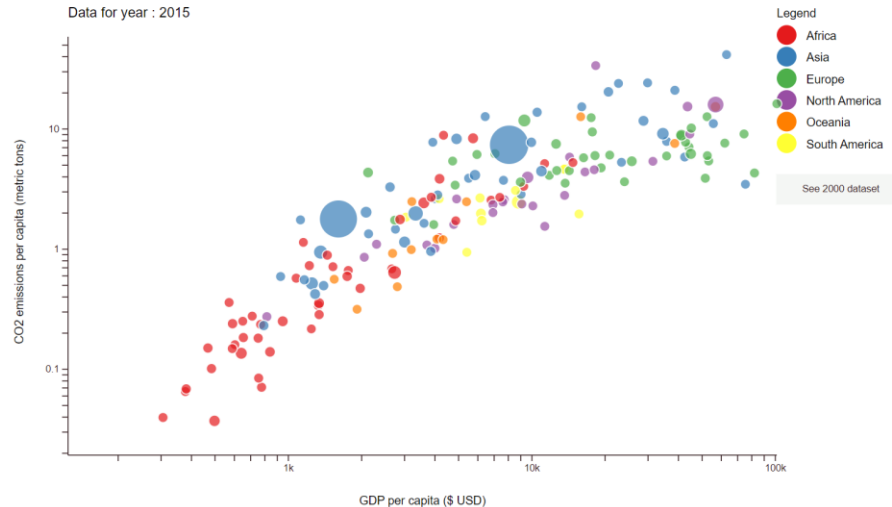
TP4

# Objectifs

- L'objectif de ce travail pratique est de créer un graphique à bulles (*bubble chart*) animé à l'aide de données ouvertes en format JSON.

## GDP vs. CO2 emissions

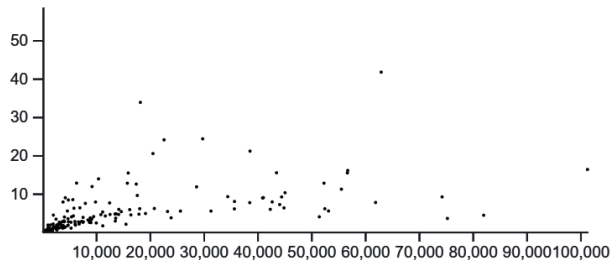
In countries around the world



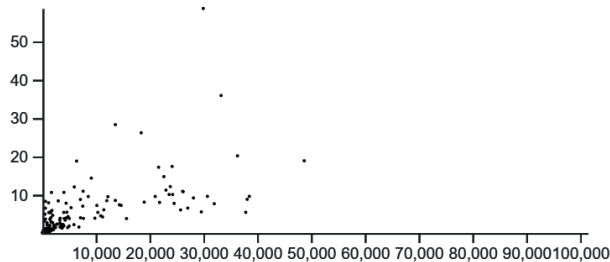


# Objectifs

- Avant de compléter le code du graphique à bulles, vous en implémenterez une version simplifiée à l'aide d'un cahier Observable.
- Lien vers le cahier : <https://observablehq.com/d/836639e52760f996>



```
draw(2015, svg2015, xScale(), yScale())
```



# Données

---

## *countriesData.json*

- Les données représentent des informations sur plusieurs pays du monde sur deux ans
- Elles se situent dans le fichier `src/assets/data/countriesData.json`
- Colonnes :
  - **Country Name:** Le nom du pays.
  - **PIB:** PIB par habitant en dollars américains courants.
  - **CO2:** Les émissions de CO2 par habitant en tonnes métriques.
  - **Population:** La population du pays.
  - **Continent:** Le continent du pays.

# Données

## *Extrait*

```
{
  "2015": [
    {
      "Country Name": "Albania",
      "GDP": 3952.8012152447,
      "CO2": 1.6026480342,
      "Population": 2880703,
      "Continent": "Europe"
    },
    { ... },
    ...
  ],
  "2000": [ ... ]
}
```

# Exploration des données

***But : Explorer les données en remplissant les sections demandées du cahier Observable***

Dans le cahier *Observable* :

1. Remplir les questions de bas en haut pour définir les échelles et tracer le nuage de points
  - La convention est de présenter le code de bas en haut en *Observable*

**Question 3 :**

```
xScale = f()  
  
xScale = function setXScale () {  
  // TODO : Define the linear scale in x for the scatter plot  
}
```

**Question 2 :**

```
yScale = f()  
  
yScale = function setYScale () {  
  // TODO : Define the linear scale in y for the scatter plot  
}
```

**Question 1 :**

```
draw = f(year, g, xScale, yScale)  
  
function draw (year, g, xScale, yScale) {  
  // TODO : Draw scatter plot, including axes and circular markers  
}
```

# Création des échelles

---

***But :** Générer les échelles utilisées pour afficher le graphique à bulles*

- Dans le fichier `scales.js` :
  1. Définissez l'échelle linéaire déterminant le rayon des cercles (fonction **setRadiusScale**)
  2. Définissez l'échelle de couleurs déterminant la couleur des cercles (fonction **setColorScale**)
  3. Définissez l'échelle logarithmique déterminant la position x des centres des cercles (fonction **setXScale**).
  4. Définissez l'échelle logarithmique déterminant la position y des centres des cercles (fonction **setYScale**).

# Graphique à bulles animé

---

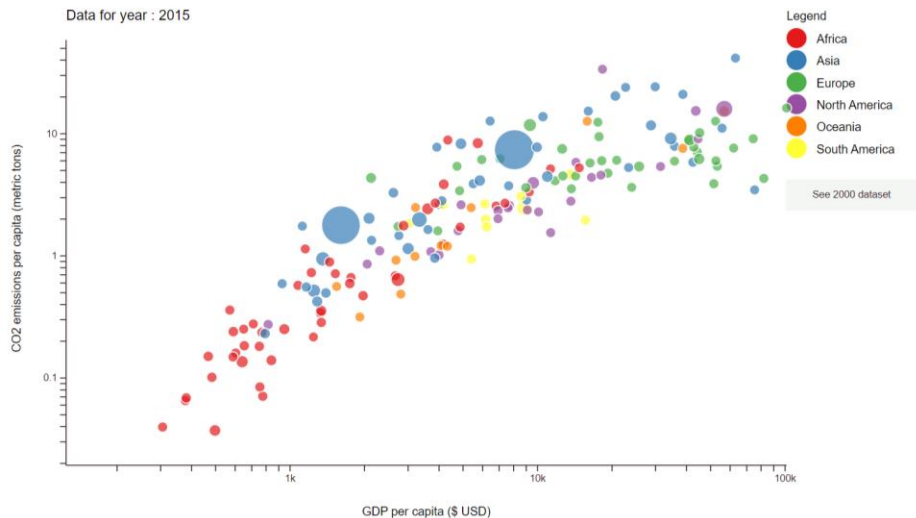
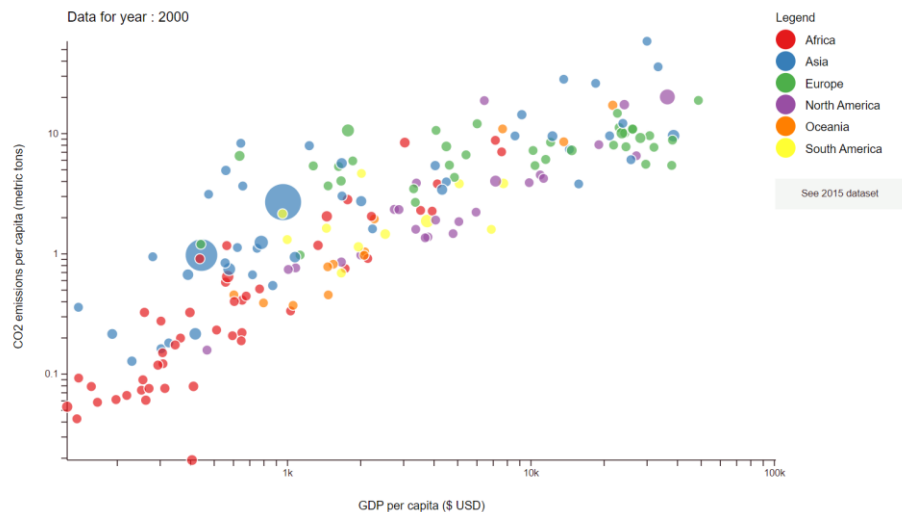
## *But*

Tracez le graphique à bulles où :

- La position du centre des cercles en **x** est correspondante au **PIB** du pays représenté par le cercle
- La position du centre des cercles en **y** est correspondante aux émissions en **CO2** du pays représenté par le cercle
- La **couleur** est correspondante au **continent** du pays représenté par le cercle
- Le **rayon** est correspondant à la **population** du pays représenté par le cercle
- Les axes sont étiquetées tel que dans l'énoncé
- L'opacité des cercles est **70%** lorsque le cercle **n'est pas survolé** par le curseur
- L'opacité des cercles est **100%** lorsque le cercle **est survolé** par le curseur
- Lorsque les données sont mises à jour, les cercles se déplacent vers leur nouvelle position en utilisant une **transition D3**

# Graphique à bulles animé

*But : Générer les échelles utilisées pour afficher le graphique à bulles*



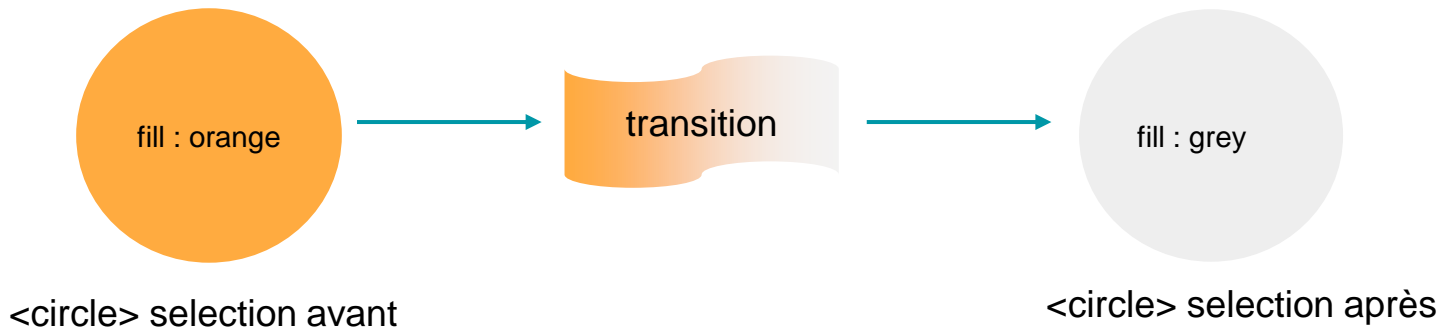
# Graphique à bulles animé

## Transition D3

- Pour faire en sorte que la position des cercles se met à jour de façon animée, on utilisera **d3.transition()**

```
d3.selectAll(...)  
.attr(...) //attribut avant la transition (optionnel, au besoin)  
.transition()  
.delay(X).duration(Y).ease(Z) //config optionnelle de la transition  
.attr(...) //valeur d'attribut vers laquelle transitionner
```

Exemple :





# Légende

---

## Legend

-  Africa
-  Asia
-  Europe
-  North America
-  Oceania
-  South America

[See 2015 dataset](#)

# Légende

*La légende indiquera le continent auquel correspond chaque couleur dans l'échelle de couleurs.*

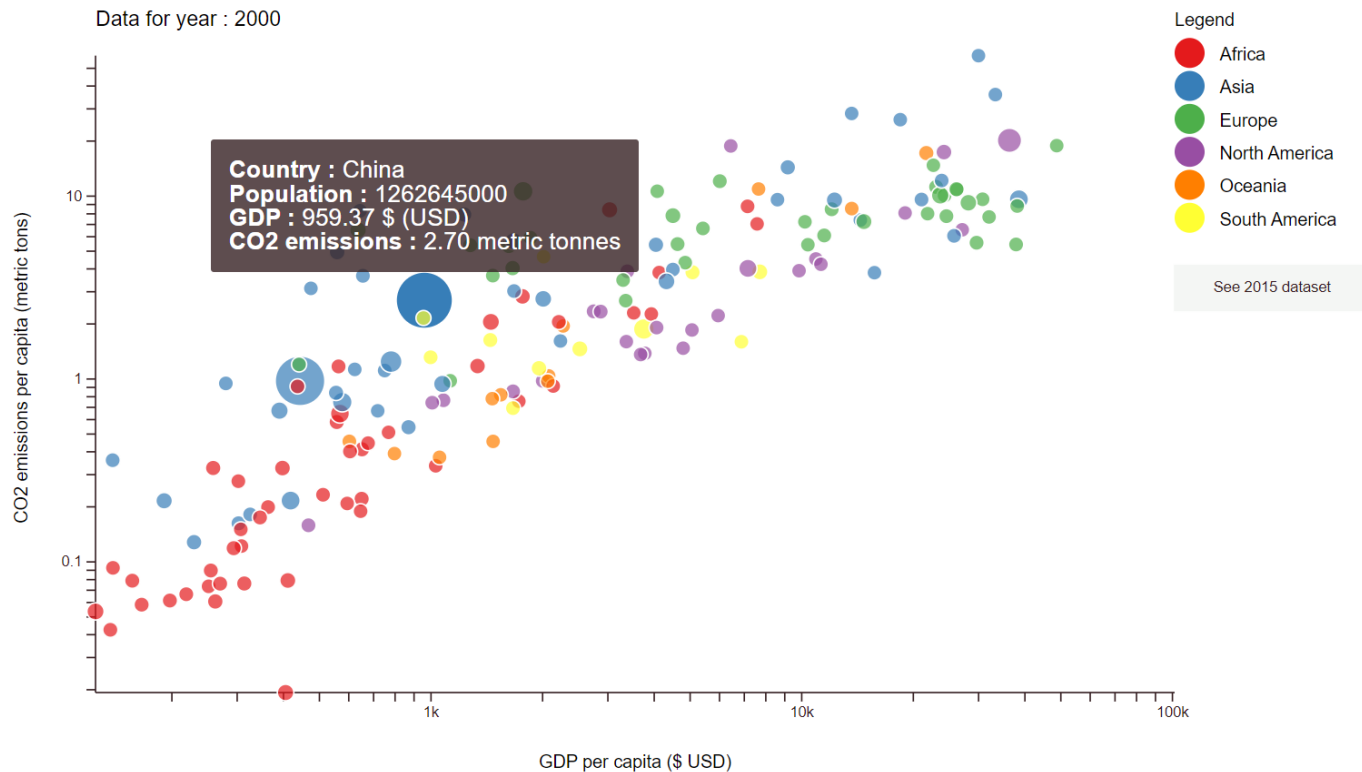
- Utilisez la bibliothèque importée en haut du fichier où on trace la légende, `legend.js` :

```
import d3Legend from 'd3-svg-legend'

/**
 * Draws the legend.
 *
 * @param {*} colorScale The color scale to use
 * @param {*} g The d3 Selection of the graph's g SVG element
 * @param {number} width The width of the graph, used to place the legend
 */
export function drawLegend (colorScale, g, width) {
  // TODO : Draw the legend using d3Legend
  // For help, see : https://d3-legend.susielu.com/
}
```

# Info-bulle

Illustré



# Info-bulle

- Dans le fichier `tooltip.js` :
  - L'info-bulle doit contenir le nom du pays, la population, le PIB par habitant et les émissions de CO2 par habitant, dans cet ordre.
  - Chaque information doit être précédée d'une étiquette correspondante et suivie, le cas échéant, des unités de mesure.
  - Assurez-vous d'avoir également géré l'info-bulle lors de la création des cercles du graphique à bulles

```
/**
 * Defines the contents of the tooltip. See CSS for tooltip styling. The tooltip
 * features the country name, population, GDP, and CO2 emissions, preceded
 * by a label and followed by units where applicable.
 *
 * @param {object} d The data associated to the hovered element
 * @returns {string} The tooltip contents
 */
export function getContents (d) {
  // TODO : Generate tooltip contents
  return ''
}
```

# Conseils TP4

---

## *Observable*

- Dans le cahier *Observable*, pour que quelque chose s'affiche, la fonction *draw()* doit retourner un noeud HTML
  - Voir la fonction *.node()*
- Si les scripts dans *Observable* semblent ne rien faire, essayer de **désactiver les extensions** du fureteur pouvant empêcher l'exécution de code sur la page
  - Ex : *Ad Block*

# Conseils TP4

## Échelles

- Rappel : Une échelle d3 sert à convertir des valeurs du *domain* vers des valeurs du *range* selon une transformation donnée
- Pour TP4, voir : **scaleLinear**, **scaleOrdinal** et **scaleLog** de D3
- Il existe des *scheme* de couleur en D3 pouvant être utilisés pour configurer un échelle couleur, par exemple **d3.schemeSet1**
- Pour créer une échelle avec ce *scheme* comme *range* :
  - `color = d3.scaleOrdinal(d3.schemeSet1)`

An array of ten categorical colors represented as RGB hexadecimal strings.

D3.js schemeSet1 Method

|                  |
|------------------|
| D3.schemeSet1[0] |
| D3.schemeSet1[1] |
| D3.schemeSet1[2] |
| D3.schemeSet1[3] |
| D3.schemeSet1[4] |
| D3.schemeSet1[5] |
| D3.schemeSet1[6] |
| D3.schemeSet1[7] |
| D3.schemeSet1[8] |

# Conseils TP4

---

## *Légende*

- Pour la légende, regardez la documentation de la bibliothèque pour plus d'aide : <https://d3-legend.susielu.com/>
- En particulier, voir comment faire des légendes de couleur et avec formes (« *shapes* ») spécifiques