Visualisation d'informations

D3 (5)

Dans ce TP, nous allons afficher un arbre avec D3 en utilisant un *sunburst.* Les exercices sont librement adaptés d'un code proposé par David Richards[1,2].

Exercice 1 : Créer de la page

Commencez par créer une page HTML. Importez la librairie D3.js.

Dans le body, insérez un script JavaScript (balise `script`) et définissez quatre variables :

- `var w = 500;` - largeur de votre visualisation,
- `var h = 500;` - hauteur de votre visualisation,
- `var r = Math.min(w,h)/2;` - rayon de votre visualisation (moitié du minimum entre la largeur et la hauteur de votre visualisation)

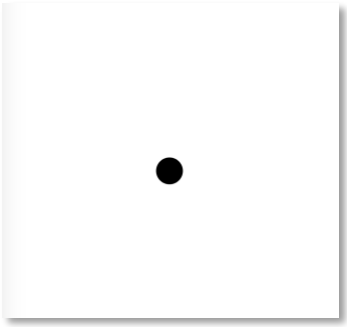
Ajoutez maintenant un objet SVG à votre body en utilisant les fonctions D3 appropriées. Ajoutez à cet objet un objet `g` et positionnez-le au centre de votre visualisation :

```
var g = svg.append('g')
    .attr('transform', 'translate(' + w/2 + ',' + h/2 + ')');
```

C'est dans cet objet que vous allez créer le *sunburst*. Comme il est placé au centre de la visualisation, les arcs pourront être placés de façon circulaire autour du point (0,0).

Ajoutez un cercle au centre de votre visualisation pour vérifier que votre code fonctionne correctement :

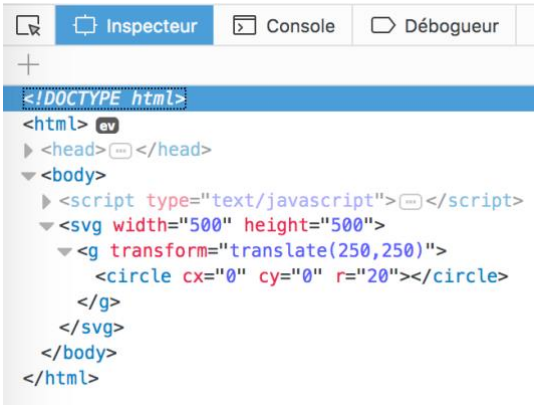
```
var circle = g.append("circle")
       .attr("cx", 0)
       .attr("cy", 0)
       .attr("r", 20);
```



[1] https://bl.ocks.org/denjn5/e1cdbbe586ac31747b4a304f8f86efa5

[2] https://bl.ocks.org/denjn5/f059c1f78f9c39d922b1c208815d18af

Avant d'aller plus loin, observez l'imbrication des balises générées grâce à l'inspecteur de votre navigateur :



Vous pouvez maintenant supprimer le cercle.

Exercice 2 : Afficher les arcs

Ouvrez le fichier flare.json et étudiez sa structure. Assurez-vous que vous comprenez comment l'arbre est encodé. Chargez ce fichier dans votre page HTML.

Ajoutez un objet partition :

```
var partition = d3.partition()
        .size([2 * Math.PI, r]);
```

Cet objet permet de d'organiser les données de façon à partitionner l'espace défini dans son attribut `size`. Il vous permettra de calculer automatiquement les tailles des objets graphiques de façon à ce qu'elles s'adaptent au volume des données. On peut passer en paramètre de l'attribut `size` une largeur et une hauteur. Dans notre cas, puisque l'on veut représenter nos données de façon circulaire, nous allons lui donner une longueur en radians (2*pi) et le rayon `r` calculé plus haut. Si vous voulez que votre *sunburst* soit placé sur un demi-cercle, il suffit d'enlever 2* devant le pi.

La commande `d3.hierarchy(data)` permet de spécifier à D3 que les données chargées sont organisées hiérarchiquement. Cette commande retourne la racine de l'arbre. Elle possède aussi une méthode `sum` permettant de calculer la taille de chaque sommet de l'arbre comme étant l'addition des tailles de ses enfants. Par exemple, un sommet ayant 2 enfants de taille 3 et 4 aura comme taille 7. Ajoutez cette commande à votre visualisation :

```
var root = d3.hierarchy(nomDeVosDonnees)
    .sum(function (d) { return d.size});
```

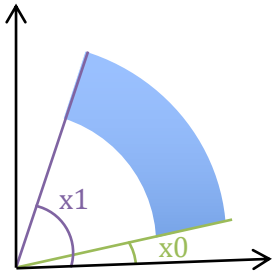
Ajoutez maintenant la fonction `partition(root);`. Avant cette ligne, chaque élément d des données est sous la forme `{name: "SonNom", size: 49}`. Ensuite, il est transformé en objet D3 de la forme `{data: Object, height: 0, depth: 2, parent: qo, value: 49…}`.

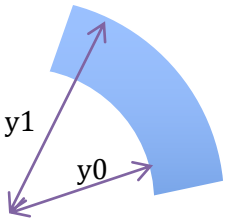
Vous pouvez maintenant créer les arcs correspondants à vos données :

```
var arc = d3.arc()
      .startAngle(function (d) { return d.x0 })
      .endAngle(function (d) { return d.x1 })
      .innerRadius(function (d) { return d.y0 })
      .outerRadius(function (d) { return d.y1 });
```

`d3.arc` permet de calculer la taille (graphique) de chaque arc :

- d.x0 correspond à la position en radian du début de l'arc,
- d.x1 correspond à la position en radian de la fin de l'arc,
- d.y0 correspond à la longueur entre le centre de la visualisation et l'intérieur de l'arc,
- d.y1 correspond à la longueur entre le centre de la visualisation et l'extérieur de l'arc.

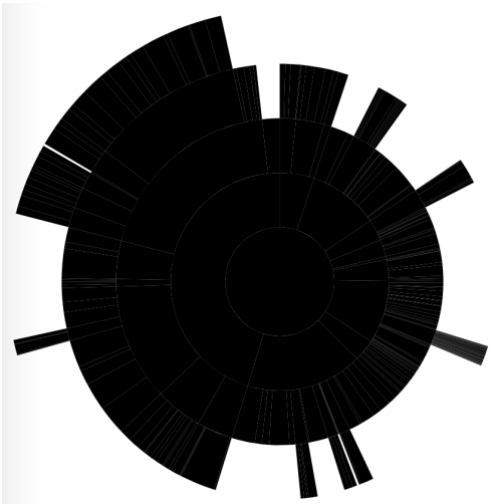


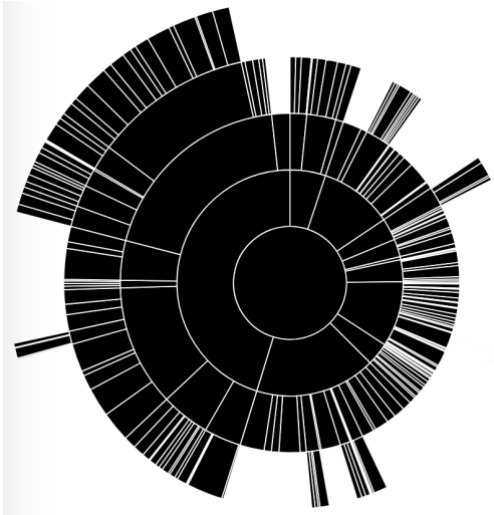


Vous pouvez maintenant créer un chemin (path) pour chaque arc avec la méthode `selectAll` :

```
var arcs = g.selectAll('.arcs')
        .data(root.descendants())
        .enter().append('path');
arcs.attr("d", arc);
```

Voici ce que vous devriez obtenir l'image suivante (à gauche) :

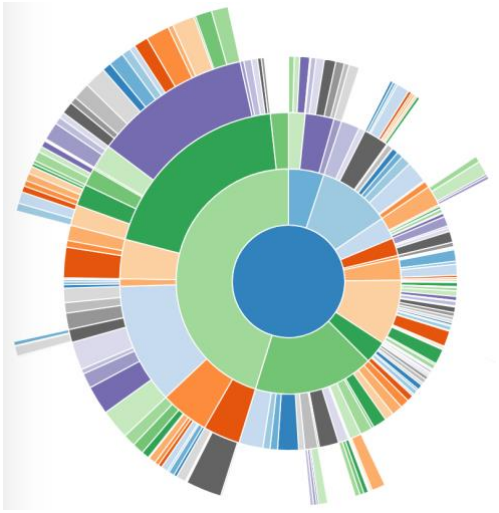




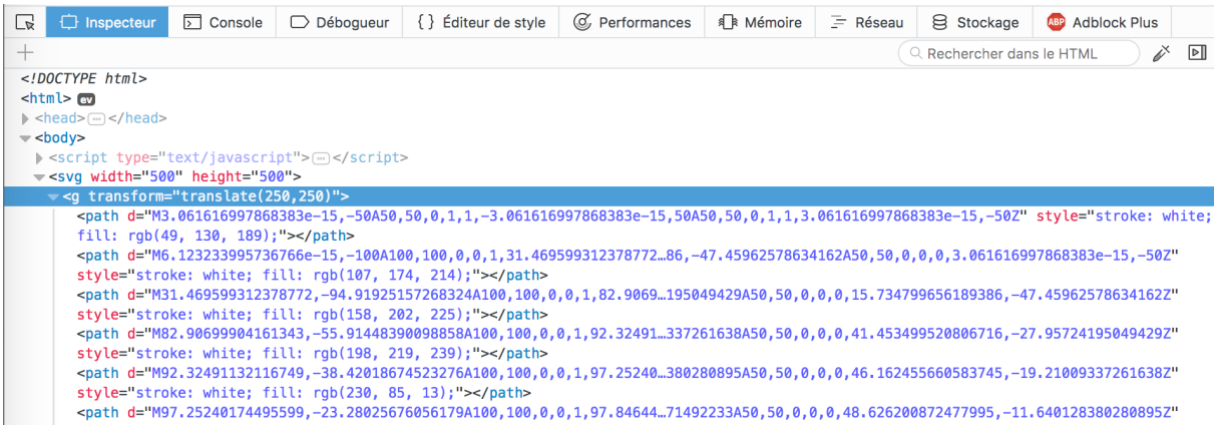
Exercice 3 : Ajouter des couleurs

Ajoutez une bordure blanche à vos arcs (attribut `stroke`) pour obtenir l'image de droite ci-dessus.

Ajoutez des couleurs à vos arcs.



Observez la liste des arcs (chemins) créés dans la balise SVG :



Exercice 4 : Interaction

Ajoutez sur les sommets une info-bulle permettant de visualiser les noms des personnages lorsque le curseur de la souris reste une seconde au-dessus.

Ajoutez un zoom.