3. Layouts en D3.js

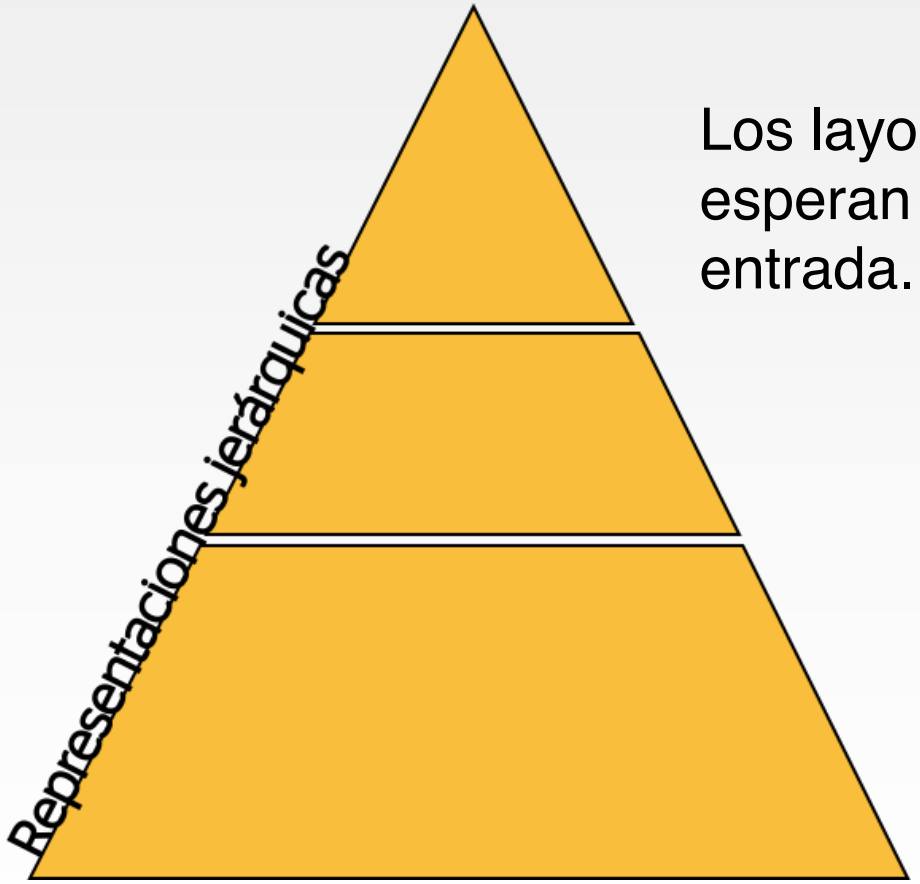
A partir de una estructura de datos JavaScript, calculan posiciones y tamaños

Cada layout espera una entrada diferente y produce una salida diferente.

https://github.com/mbostock/d3/wiki/Layouts

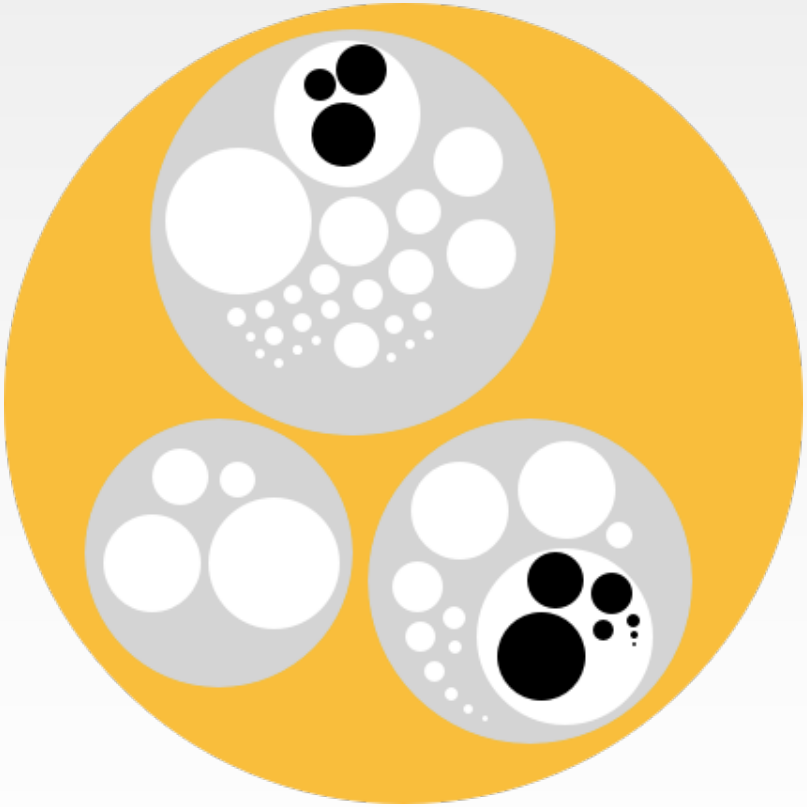
https://github.com/mbostock/d3/wiki/Hierarchy-Layout

3. Layouts en D3.js



Los layouts jerárquicos esperan todos la misma entrada.

3. Layouts en D3.js



Representa mediante círculos anidados la jerarquía

Los círculos padre reflejan aproximadamente el acumulado del tamaño de sus hijos

https://github.com/mbostock/d3/wiki/Pack-Layout

3. Layouts en D3.js

Inicialización del layout:

```
var cp = d3.layout.pack()
.size([<ancho>, <alto>])
.value(function(d) {
    return <atributo que define el tamaño del nodo>;
});
```

Aplicación del layout a los datos:

```
var leaves = cp(<variable con los datos>);
```

Para cada dato se habrá calculado:

x, y, r

3. Layouts en D3.js

Abrir circlePackingTheDoors.html

3. Layouts en D3.js

Ejercicio

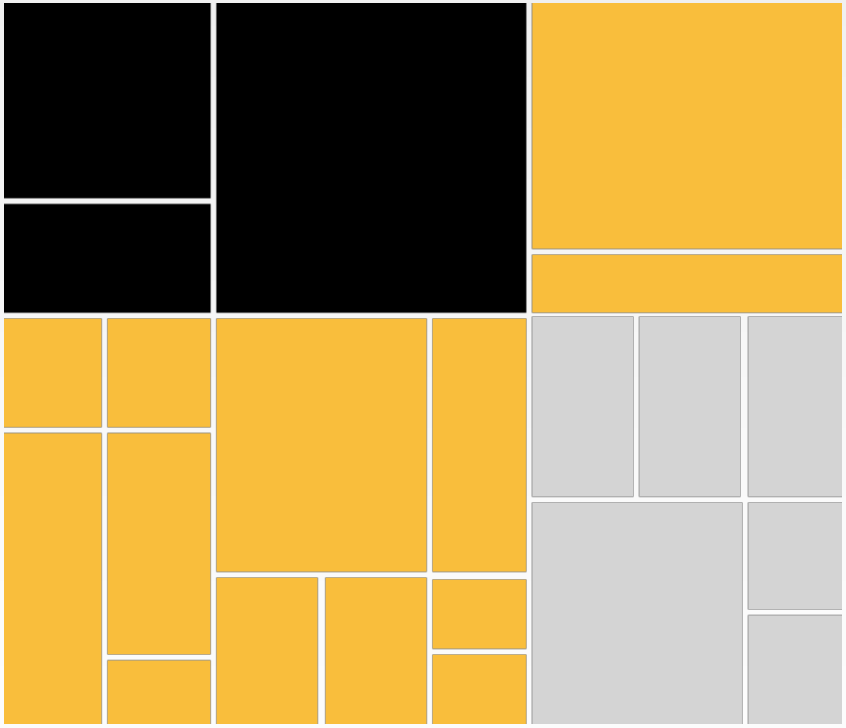
Cambiar a negativo las canciones en un mouseover

Quitar el tooltip en el nodo padre

El tooltip debe mostrar la longitud de la canción en minutos y el álbum asociado en negrita

En el caso de un álbum mostrar su duración total en minutos

3. Layouts en D3.js



Subdivide recursivamente un área en rectángulos

El tamaño de un nodo en el árbol es rápidamente revelado

https://github.com/mbostock/d3/wiki/Treemap-Layout

Inicialización del layout:

3. Layouts en D3.js

```
var tm = d3.layout.treemap()
.size([<ancho>, <alto>])
.value(function(d) {
    return <atributo que define el tamaño del nodo>;
});
```

Aplicación del layout a los datos:

```
var leaves = tm(<variable con los datos>);
```

Para cada dato se habrá calculado:

x, y, dx, dy

3. Layouts en D3.js

Ejercicio

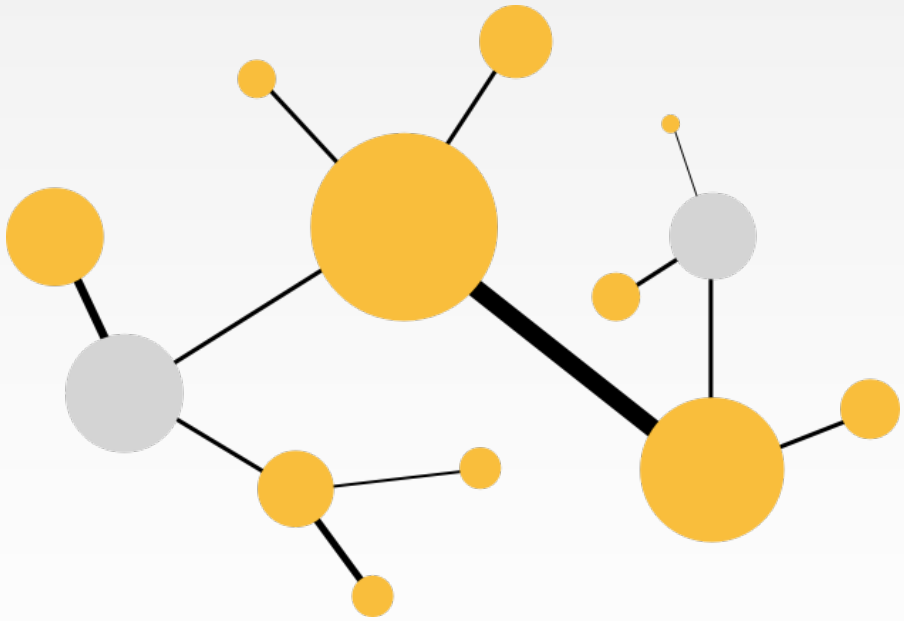
Dibujar un treemap en lugar de un circle packing

Asignar una escala de color ordinal para cada canción, pero con el color de su álbum.

Solucionar las colisiones de nombres con la función anónima de data (Unicidad)

3. Layouts en D3.js

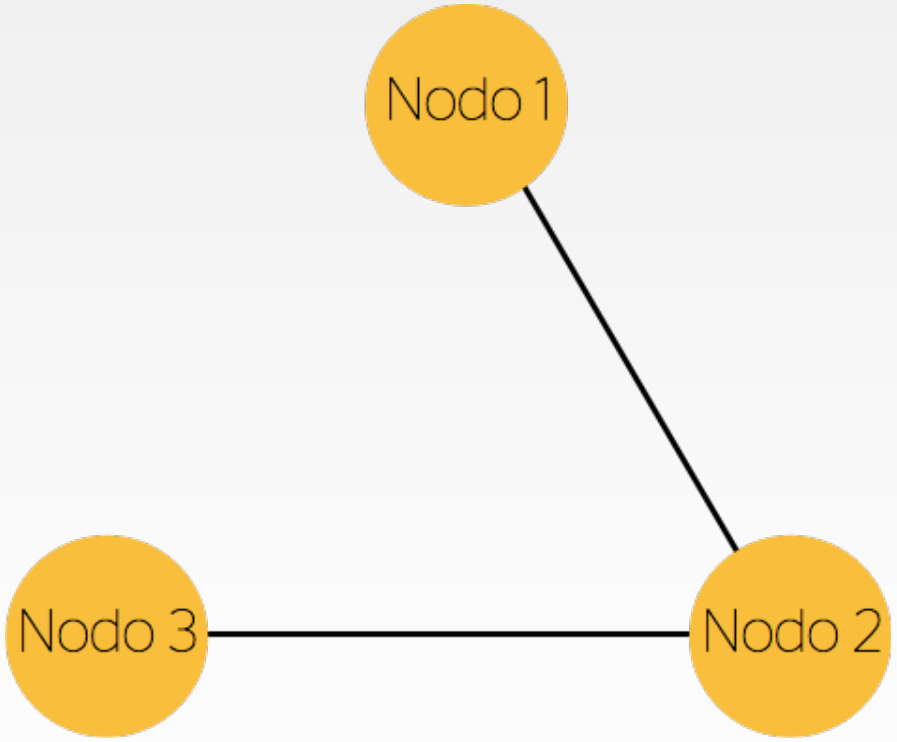
La entrada es una lista de nodos y una lista de aristas



https://github.com/mbostock/d3/wiki/Force-Layout

3. Layouts en D3.js

```
{
    “nodes”: [
        { “name”: “Nodo 1” },
        { “name”: “Nodo 2” },
        { “name”: “Nodo 3” }
    ],
    “edges”: [
        { “source”: 0, “target”: 1 },
        { “source”: 1, “target”: 2 },
    ]
}
```



3. Layouts en D3.js

Inicialización del *layout*:

```
var force = d3.layout.force()
.size([<ancho>, <alto>])
.charge(<fuerza a aplicar entre los nodos>)
.linkDistance(<longitud deseada de las aristas>)
.gravity(<fuerza de gravedad a utilizar>);
```

Aplicación del *layout* a los datos:

```
force.nodes(<lista de nodos>)
    .links(<lista de aristas>)
    .start();
```

3. Layouts en D3.js

Cada vez que se recalculan posiciones se produce un evento tick:

```
force.on("tick", function() {
    …
})
```

Para cada nodo se calcula:

index, x, y, px, py, weight

3. Layouts en D3.js

Abrir network.html

3. Layouts en D3.js

Ejercicio

- Dibujar el radio del círculo en función del número de conexiones
- Dibujar el stroke-opacity de cada vértice en función del número de conexiones del nodo

3. Layouts en D3.js

Ejercicio

Dibujar en lugar de un círculo una etiqueta text con el nombre del nodo

El tamaño de la fuente tiene que ser dependiente del número de conexiones del nodo

3. Layouts en D3.js

Ejercicio

Descargar de https://apps.facebook.com/netvizz/ la red de contactos de Facebook de cada uno

Visualizarla con un *Force layout*

En el caso de no tener cuenta en Facebook, utilizar el fichero flocker_network.json

3. Layouts en D3.js

US Homicides 2010 - Jérôme Cukier

Population Choropleth - Mike Bostock

Situación de los supermercados Walmart - Zachari Forest (indiemaps)

Plant Hardiness Zones - Mike Bostock

Map Projection Transitions - Jason Davies

Un Mundo de Migrantes - Outliers Collective

3. Layouts en D3.js

Cartográficos

polígonos que forman las CCAA

tiles

Cuantitativos asociados a Geo

población

incendios

tweets geolocalizados

3. Layouts en D3.js

¿Dónde conseguir datos cartográficos?

Natural Earth

http://www.naturalearthdata.com/

Spatial Data Repository

http://spatialdata.dhsprogram.com

UCLA Spatial Data Repository

http://gis.ats.ucla.edu

3. Layouts en D3.js

Formato de archivo informático propietario de **datos espaciales** desarrollado por la compañía ESRI

Actualmente se ha convertido en formato estándar de facto

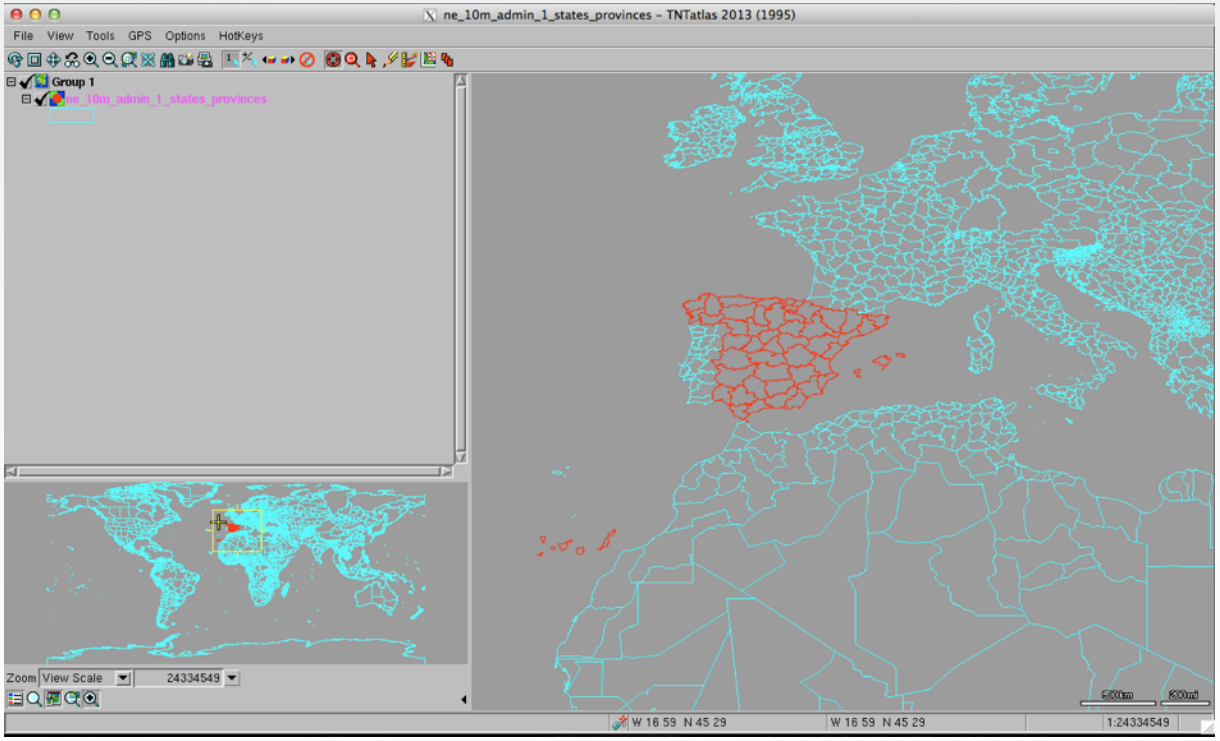
Permite almacenar atributos asociados a las diferentes geometrías

http://es.wikipedia.org/wiki/Shapefile

3. Layouts en D3.js

Visor SHP

TNTAtlas



http://www.microimages.com/products/tntatlas/

3. Layouts en D3.js

GeoJSON

http://en.wikipedia.org/wiki/GeoJSON

TopoJSON

https://github.com/mbostock/topojson

3. Layouts en D3.js

De SHP a TopoJSON

Descargar los **SHP**:

Natural Earth, Admin 1 – States & Provinces
Natural Earth, Populated Places

Utilizando **ogr2ogr** y **topojson** generar TopoJSON:

```
ogr2ogr -f GeoJSON -where "iso_a2 IN ('ES', 'PT', 'FR', 'MA')" subunits.json
ne_10m_admin_1_states_provinces.shp

ogr2ogr -f GeoJSON -where “ISO_A2 = ‘ES' AND SCALERANK < 8" places.json
ne_10m_admin_1_states_provinces.shp

topojson --id-property adm1_code -p name=name -p name -p population=POP_MAX -p
population -o es.json subunits.json places.json
```

3. Layouts en D3.js

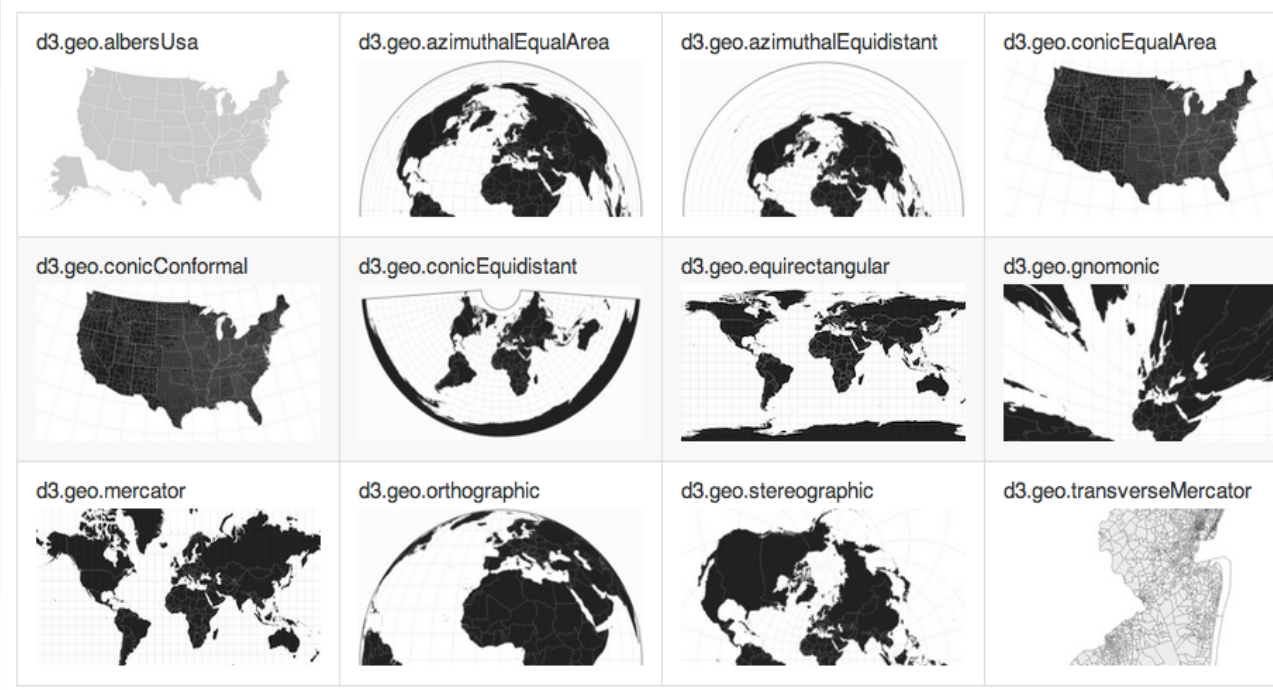
Para renderizar cartografías necesitaremos una **proyección** y un **generador de paths**

Una **proyección** proyecta coordenadas esféricas al plano cartesiano

Un **generador de paths** toma la geometría 2D proyectada y la formatea adecuadamente a SVG

3. Layouts en D3.js

```
var mercatorProjection = d3.geo.mercator();
/* Albers centrada sobre el Reino Unido */
var albersProjections = d3.geo.albers()
                        .center([0, 55.4])
                        .rotate([4.4, 0])
                        .parallels([50, 60])
                        .scale(6000);
```



https://github.com/mbostock/d3/wiki/Geo-Projections

3. Layouts en D3.js

Obtener las coordenadas de un punto en una proyección

```
var projection = d3.geo.mercator();
var punto = {
    “latitude”: 40.4165000,
    “longitude”: -3.7025600
};
var x = projection([punto.longitude, punto.latitude])[0];
var y = projection([punto.longitude, punto.latitude])[1];
```

ejemplo1.html

3. Layouts en D3.js

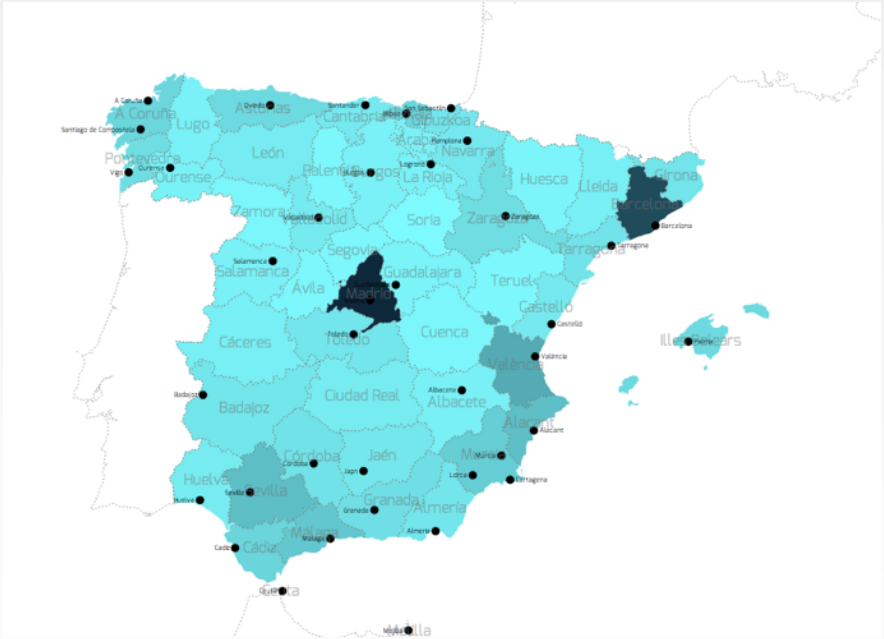
Generadores de paths en d3.geo

https://github.com/mbostock/d3/wiki/Geo-Paths

```
var svg = d3.select("body")
          .append("svg")
          .attr("width", 1200)
          .attr("height", 700);
d3.json("es.json", function (error, es) {
   var projection = d3.geo.mercator();
   var generador = d3.geo.path()
                .projection(projection);
   svg.selectAll("path")
      .data(topojson.feature(es, es.objects.subunits).features)
      .enter()
      .append("path")
      .attr("d", function (d) { return generador(d); });
});
```

ejemplo2.html
ejemplo3.html

3. Layouts en D3.js



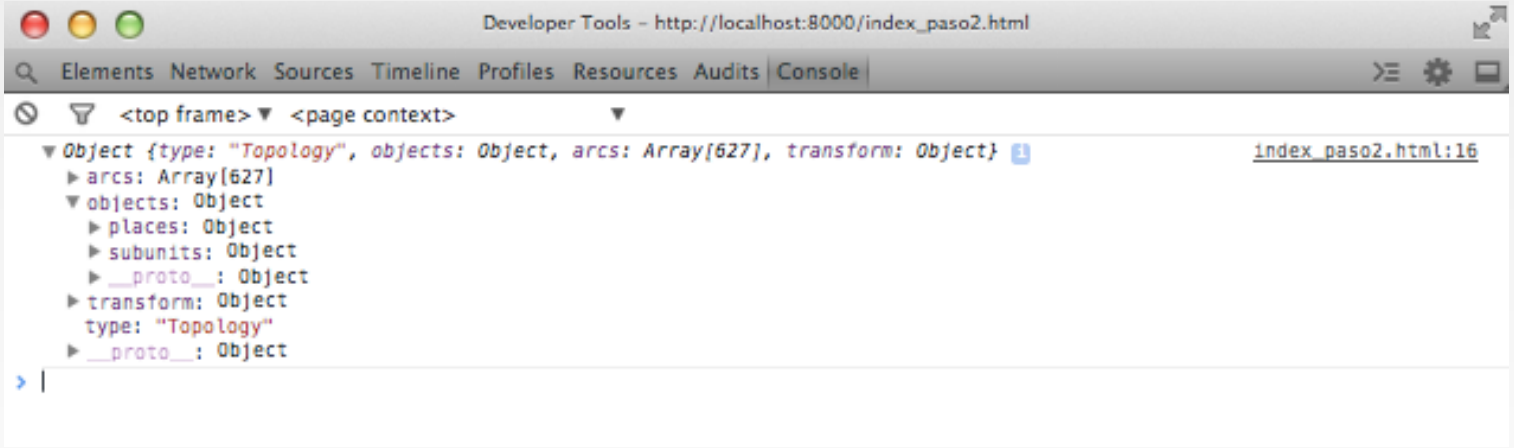
Vamos a dibujar el mapa de España con sus provincias y a colorearlo según diferentes estadísticas del INE

3. Layouts en D3.js

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <style>
         /* Código CSS */
      </style>
   </head>
   <body>
      <script src="http://d3js.org/d3.v3.min.js"></script>
      <script src="http://d3js.org/queue.v1.min.js"></script>
      <script src="http://d3js.org/topojson.v1.min.js"></script>
      <script>
         /* Código JavaScript */
      </script>
   </body>
</html>
```

PASO 1: Fichero HTML de partida

3. Layouts en D3.js




```
<script>
        /* Código JavaScript */
        d3.json("es.json", function(error, es) {
                console.log(es);
        });
</script>
```

PASO 2: Cargar los datos

3. Layouts en D3.js

```
<script>
        /* Código JavaScript */
        var width = 1200,
    height = 700;
    var svg = d3.select("body")
          .append("svg")
          .attr("width", width)
          .attr("height", height);
    d3.json("es.json", function (error, es) {
        svg.append("path")
      .datum(topojson.feature(es, es.objects.subunits))
      .attr("d", d3.geo.path().projection(d3.geo.mercator()));
    });
</script>
```

PASO 3: Dibujar los polígonos

3. Layouts en D3.js

```
d3.json("es.json", function (error, es) {
        var subunits = topojson.feature(es, es.objects.subunits),
            projection = d3.geo.albers()
                                                    .center([0, 40.23])
                                                    .rotate([3.43, 0])
                                                    .parallels([40, 60])
                                                    .scale(4000)
                                                    .translate([width / 2, height / 2]),
                path = d3.geo.path()
                                            .projection(projection);
        svg.append("path")
           .datum(subunits)
           .attr("d", path);
});
```

PASO 4: Ajustar la proyección

3. Layouts en D3.js

```
<style>
        /* Código CSS */
        .subunit.ESP { fill: #ddc; }
        .subunit.FRA, .subunit.MAR, .subunit.PRT { display: none; }
</style>
…
svg.append(“path")
  .datum(subunits)
  .attr("d", path);
svg.selectAll(".subunit")
  .data(subunits.features)
  .enter()
  .append("path")
  .attr("class", function (d) { return "subunit " + d.id.slice(0, 3); })
  .attr("d", path);
```

PASO 5: Aplicando estilos a los polígonos

```
<style>
        …
        .subunit-boundary {
                fill: none;
                stroke: #777;
                stroke-dasharray: 2,2;
                stroke-linejoin: round;
        }
        .subunit-boundary.OTHER { stroke: #aaa; }
</style>
…
svg.append("path")
  .datum(topojson.mesh(es, es.objects.subunits, function(a, b) { return a !== b && a.id === "ESP"; }))
  .attr("d", path)
  .attr("class", "subunit-boundary");
svg.append("path")
  .datum(topojson.mesh(es, es.objects.subunits, function(a, b) { return a === b && a.id !== "ESP"; }))
  .attr("d", path)
  .attr("class", "subunit-boundary OTHER");
```

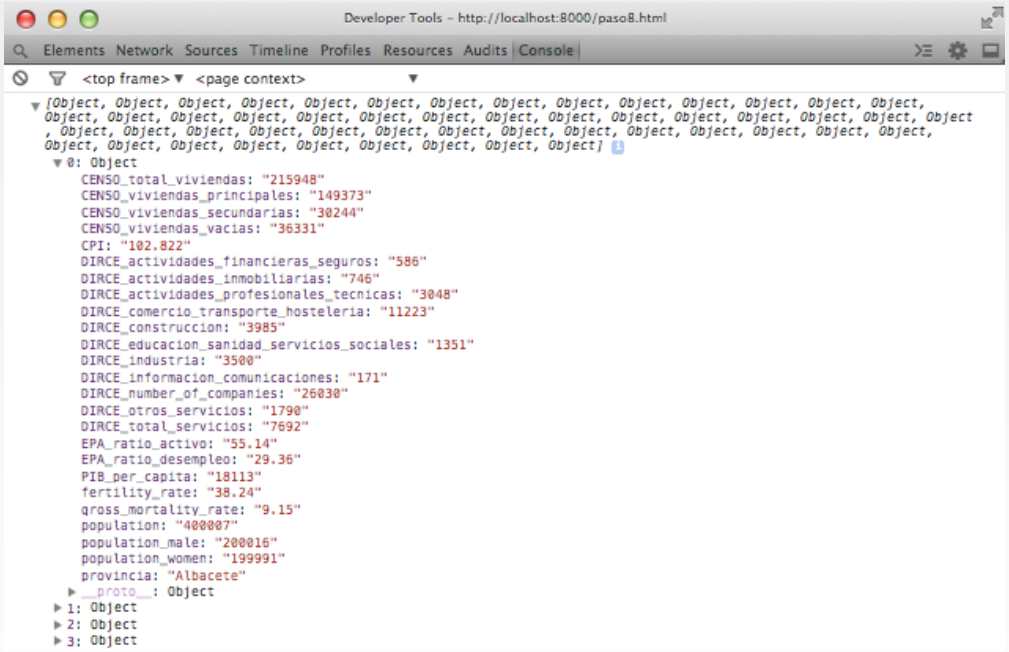
PASO 6: Dibujar los bordes

3. Layouts en D3.js

```
vg.append("path")
 .datum(topojson.feature(es, es.objects.places))
 .attr("d", path)
 .attr("class", "place");
vg.selectAll(".place-label")
 .data(topojson.feature(es, es.objects.places).features)
 .enter()
 .append("text")
 .attr("class", "place-label")
 .attr("transform", function(d) { return "translate(" + projection(d.geometry.coordinates) + ")"; })
 .attr("dy", ".35em")
 .text(function(d) { return d.properties.name; });
vg.selectAll(".place-label")
 .attr("x", function(d) { return d.geometry.coordinates[0] > -1 ? 6 : -6; })
 .style("text-anchor", function(d) { return d.geometry.coordinates[0] > -1 ? "start" : "end"; });
```

PASO 7: Mostrar las ciudades más pobladas

3. Layouts en D3.js



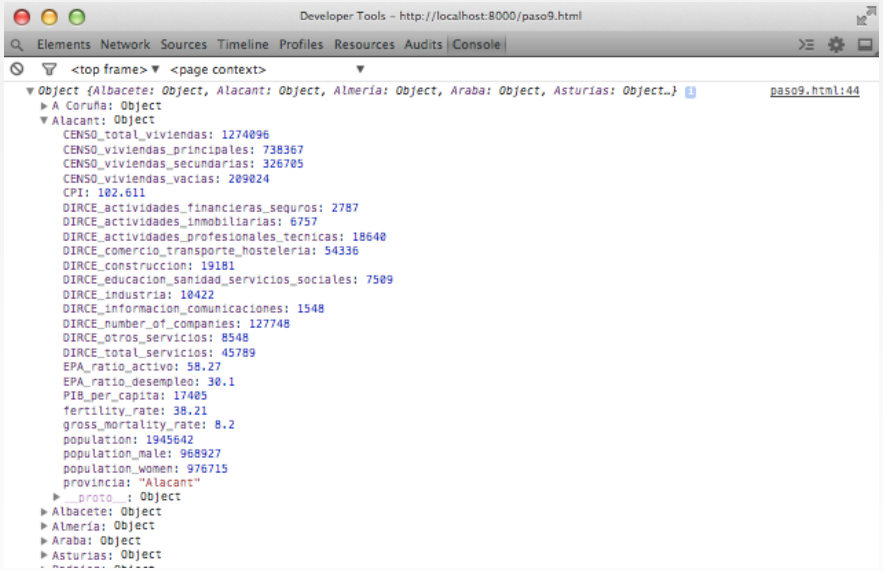
```
▼ [Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object,
  Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object
  , Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object,
  Object, Object, Object, Object, Object, Object, Object, Object, Object]
  ▼ 0: Object
      CENSO_total_viviendas: "215948"
      CENSO_viviendas_principales: "149373"
      CENSO_viviendas_secundarias: "30244"
      CENSO_viviendas_vacias: "36331"
      CPI: "102.822"
      DIRCE_actividades_financieras_seguros: "586"
      DIRCE_actividades_inmobiliarias: "746"
      DIRCE_actividades_profesionales_tecnicas: "3048"
      DIRCE_comercio_transporte_hosteleria: "11223"
      DIRCE_construccion: "3985"
      DIRCE_educacion_sanidad_servicios_sociales: "1351"
      DIRCE_industria: "3500"
      DIRCE_informacion_comunicaciones: "171"
      DIRCE_number_of_companies: "26030"
      DIRCE_otros_servicios: "1790"
      DIRCE_total_servicios: "7692"
      EPA_ratio_activo: "55.14"
      EPA_ratio_desempleo: "29.36"
      PIB_per_capita: "18113"
      fertility_rate: "38.24"
      gross_mortality_rate: "9.15"
      population: "400007"
      population_male: "200016"
      population_women: "199991"
      provincia: "Albacete"
    ▶ __proto__: Object
  ▶ 1: Object
  ▶ 2: Object
  ▶ 3: Object
```

```
queue().defer(d3.json, "json/es.json")
      .defer(d3.tsv, "tsv/ine.tsv")
      .await(ready);
function ready (error, es, ine) {
    console.log(ine);
            /* Todo el código que teníamos dentro de la llamada a d3.json */
};
```

PASO 8: Cargar el TSV con los datos del INE

3. Layouts en D3.js

```
/* Al principio de la función ready */
var provincesData = {};
ine.forEach(function (d, i) {
        Object.keys(d).forEach(function (k) {
                var v = parseFloat(d[k]);
                if (!isNaN(v)) { d[k] = v; }
        });
        provincesData[d.provincia] = d;
});
console.log(provincesData);
```



PASO 9: Estructura auxiliar con los datos del INE

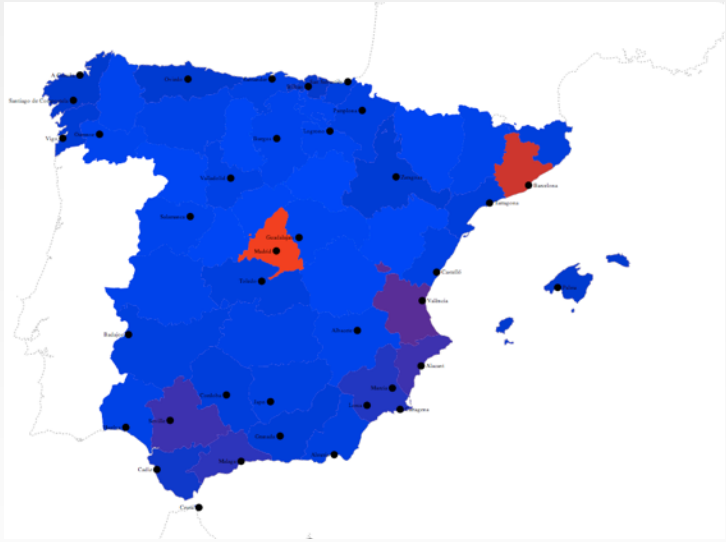
3. Layouts en D3.js

```
/* Primero definimos una escala de color */
var colorScale = d3.scale
                .linear()
                .domain([
                  d3.min(ine, function (d) { return d.population; }),
                  d3.max(ine, function (d) { return d.population; }),
                ])
                .range(["#0000ff", "#ff0000"]);
```

PASO 10: Colorear las provincias según su población

3. Layouts en D3.js

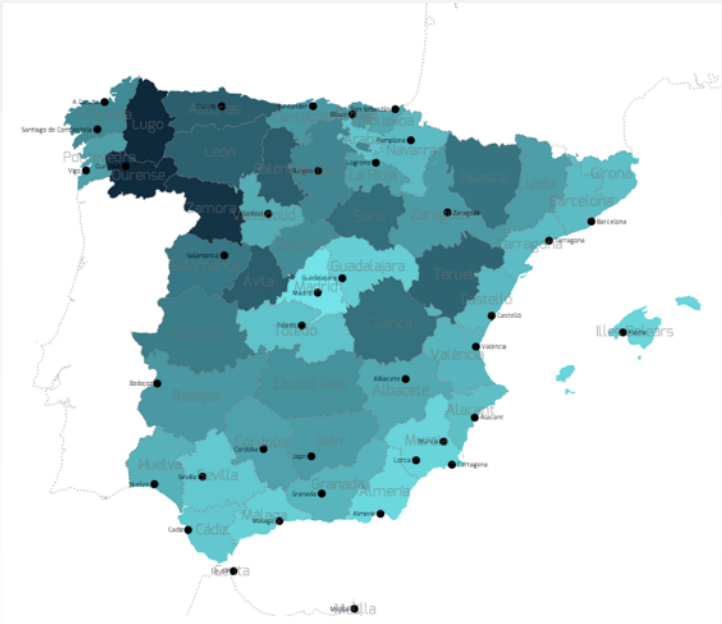
```
/*
 Debemos aplicar el color a los polígonos
 de las provincias
*/
svg.selectAll(".subunit")
   .data(subunits.features)
   .enter()
   .append("path")
   .attr("class", function (d) { return "subunit " + d.id.slice(0, 3); })
   .attr("d", path)
   .style("fill", function (d) {
     if ("properties" in d){
       if ("name" in d.properties) {
         if (d.properties.name in provincesData) {
           return colorScale(provincesData[d.properties.name].population);
         }
       }
     }
     return "none";
   });
```



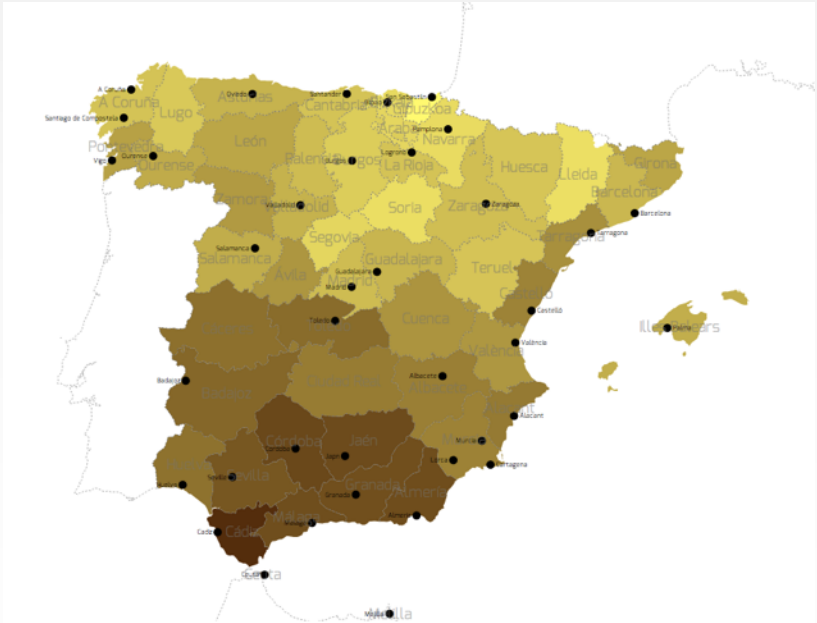
PASO 10: Colorear las provincias según su población

3. Layouts en D3.js

Índice de mortalidad

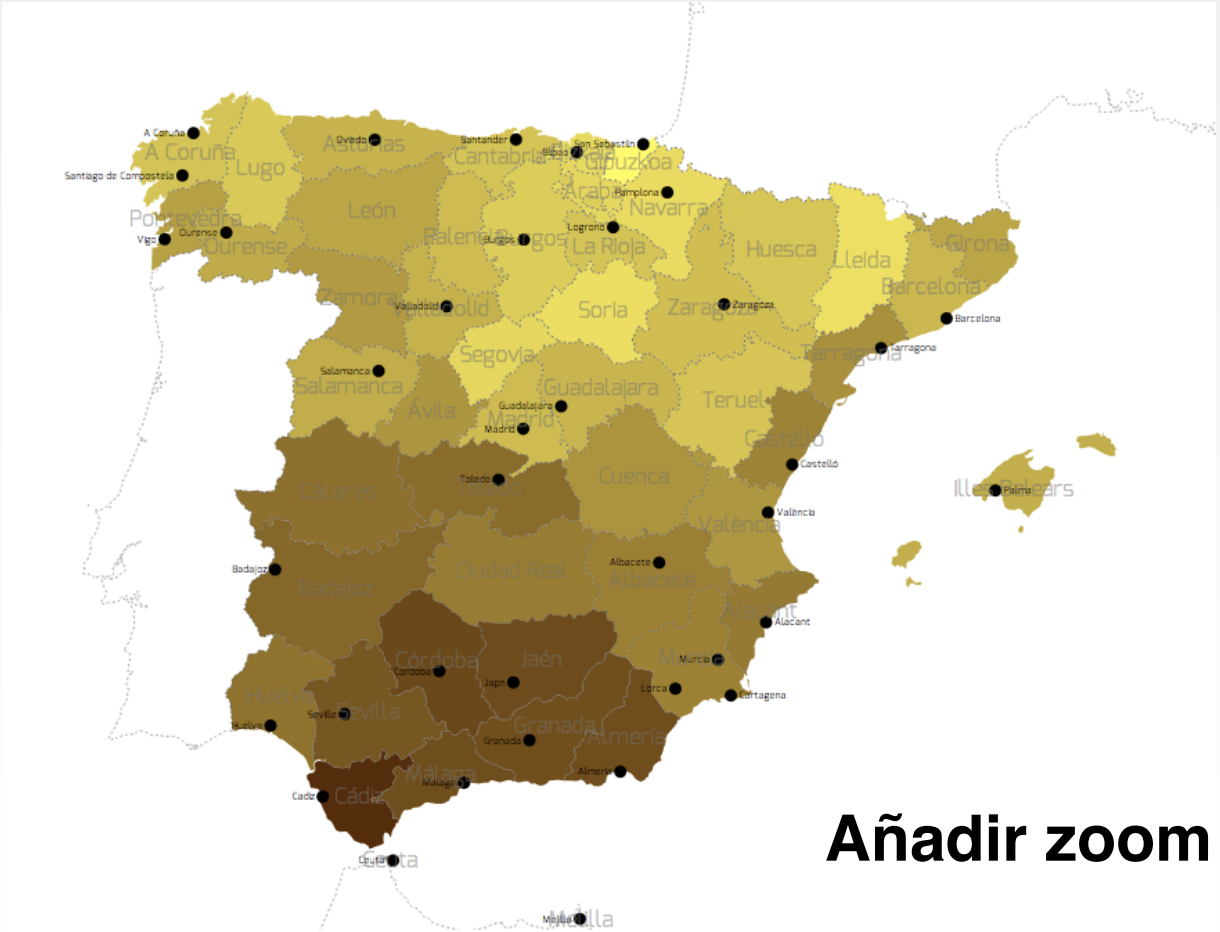


Ratio de desempleo



Colorear las provincias utilizando otras estadísticas del INE

3. Layouts en D3.js



Añadir zoom al mapa

3. Layouts en D3.js

Añadir tooltips al mapa que muestren el valor de la estadística actual para la provincia señalada

3. Layouts en D3.js

Las cuerdas son elementos complejos

Metáfora de visualización para ver relaciones entre N entidades

3. Layouts en D3.js

Inicialización del *layout*:

```
var chord = d3.layout.chord()
.padding(<d3.ascending o d3.descending>)
.sortSubgroups(<d3.ascending o d3.descending>)
.sortChords(<d3.ascending o d3.descending>);
```

Aplicación del *layout* a los datos:

```
chord.matrix(<matriz con los datos>)
```

Calcula para los datos:

groups, chords

3. Layouts en D3.js

Función para generar *ticks* dentro de un arco concreto

```
function groupTicks(d) {
    var k = (d.endAngle - d.startAngle) / d.value;
    return d3.range(0, d.value, 1000).map(function(v, i) {
        return {
            angle: v * k + d.startAngle,
            label: i % 5 ? null : v / 1000 + "k"
        };
    });
}
```

3. Layouts en D3.js

Ejercicio

Añadir zoom y panning

```
var matrix = [
        [11975,  5871, 8916, 2868],
        [ 1951, 10048, 2060, 6171],
        [ 8010, 16145, 8090, 8045],
        [ 1013,   990,  940, 6907]
    ];
```

Resaltar la cuerda sobre la que te pongas y los arcos que une

Cambiar la opacidad de las cuerdas según el arco donde tengamos el ratón