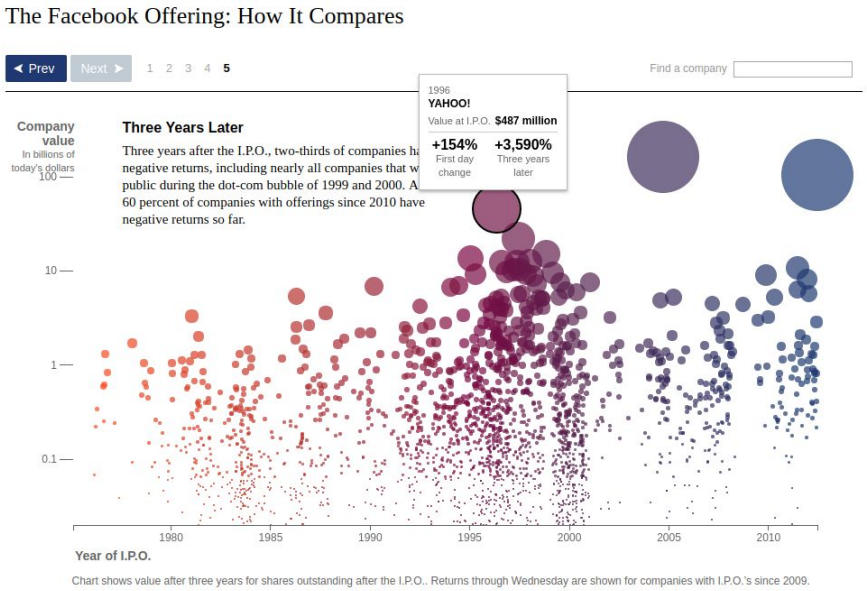
Visualização de Informação com D3.js

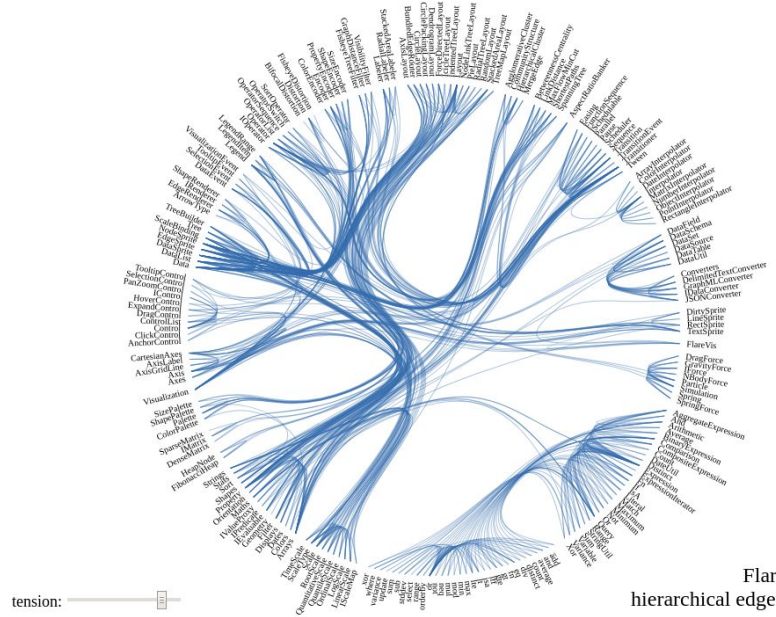
Evandro Ortigossa

25 de setembro de 2017

Introdução

Visualização de dados que não tem forma ou cor inerente.





Flare imports
hierarchical edge bundling

Introdução

Porque visualização de informação?

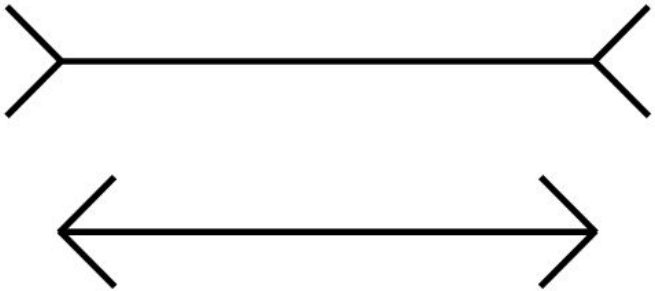
- 40% das fibras nervosas conectadas ao cérebro estão ligadas à retina
- 50% dos tecidos neurais estão direta ou indiretamente ligados à visão

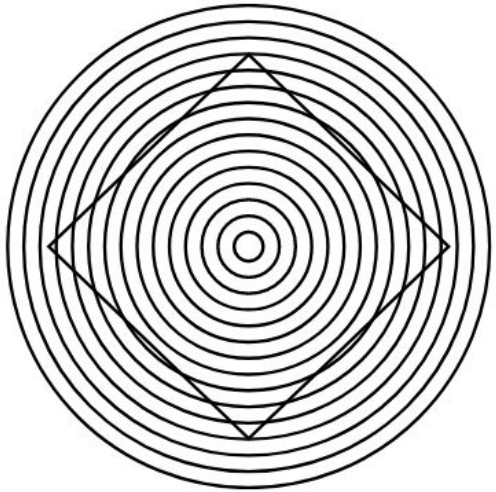
Representar dados visualmente, comunicando

- Padrões
- Tendências
- Anomalias

Introdução

No entanto, a visão não é perfeita...





Introdução

Guidelines

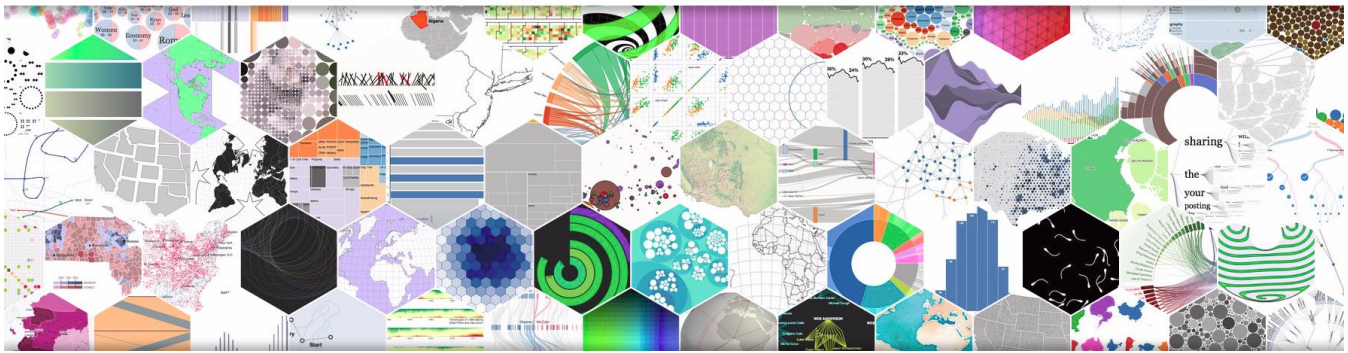
- Casamento entre *natureza* dos dados e *representação* visual
- Quando duas ou mais ferramentas podem desempenhar a mesma tarefa, escolha a que permite o trabalho mais valioso por unidade de tempo
- Adote novas soluções apenas quando o retorno for maior do que o custo de se aprender
- Representações devem levar em conta as capacidades sensoriais humanas, destacando elementos importantes e acelerando a percepção de padrões
- Simbologia gráfica deve ser padronizada dentro e entre aplicações

"Information Visualization: Perception for Design", Collin Ware

D3.js

- Biblioteca javascript para a visualização de informação
- Criada por Mike Bostock
- Publicada no IEEE Vis 2011
- https://d3js.org/

Data-Driven Documents



O que é D3 ?

- Não é uma biblioteca para gráficos
- Não é uma biblioteca para mapas
- Não é uma biblioteca para desenhar em SVG

D3 é...

Uma biblioteca para que você construa aplicações em que os dados entram puros e são dinamicamente associados em representações gráficas

D3 é web :)

Primeiros passos... JavaScript

- Funções são como outros tipos básicos (e.g. int, char)
- Definindo foo de maneiras equivalentes:

```
function foo(i) {
   return i + 1;
}
```

```
// Uma função anônima que agora tem nome 'foo'
var foo= function(i) { return i + 1; };
```

- A chamada é a mesma para as duas declarações:

```
console.log(foo(1)); // 2
```

Primeiros passos... JavaScript

- Objetos

```
var obj = {
   // Atributos
   "val": 0,
   "name": "foo",
   // Métodos
   "show": function() { console.log(this.name + ": " + this.val); },
   "setValue": function(val) { this.val= val; },
   "setName": function(name) { this.name= name; }
};
obj.show(); // "foo: 0"
obj.setValue(1);
obj.setName("bar");
obj.show(); // "bar: 1"
```

Primeiros passos... JavaScript

- Objetos, como deixar mais elegante?

```
obj.setValue(1)
  .setName("bar")
  .show(); // "bar: 1"
```

- Convenção: métodos retornam o próprio objeto quando alteram estado

```
  "setValue": function(val) {
       this.val= val;
       return this;
    },
    "setName": function(name) {
       this.name= name;
       return this;
    },
```

Mapeando dados

- Suponha o seguinte conjunto de dados

```
var dataset= [ 50, 380, 470, 110, 390, 40, 80, 190 ];
```

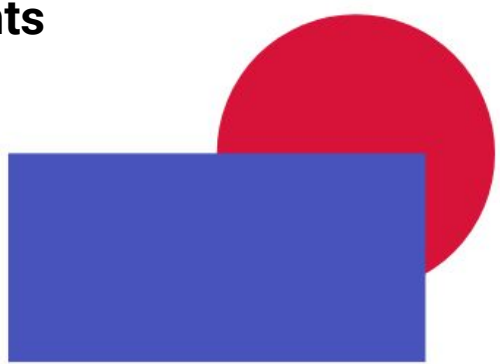
- Como mostrar esses dados em um gráfico de barras?

Como o D3.js "desenha" na tela?

- Ele não desenha!
- A essência do D3.js: gerar documentos a partir de dados

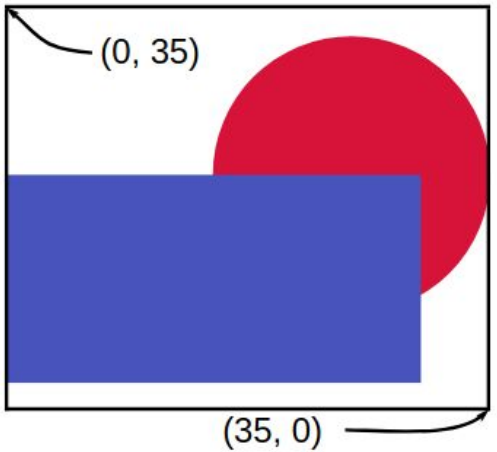
D3: Data-Driven Documents

- SVG (Scalable Vector Graphics)



Como o D3.js "desenha" na tela?

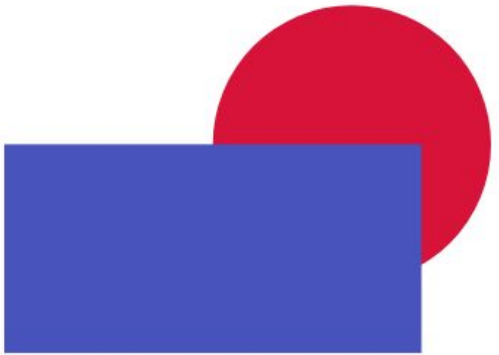
```
<?xml version="1.0"
     encoding="UTF-8"
     standalone="no"?>
<svg viewBox="0 0 35 35"
     xmlns="http://www.w3.org/2000/svg">
   <circle cx="25"
           cy="15"
           r="10"
           style="fill: #d13e2e;"/>
   <rect x="0"
         y="15"
         width="30"
         height="15"
         style="fill: #2d6ac2;"/>
</svg>
```



Como o D3.js "desenha" na tela?

- A API do D3 permite manipular elementos para gerar um documento

```
svg.append("circle")
   .attr("cx", 25)
   .attr("cy", 15)
   .attr("r", 10)
   .style("fill", "#d13e2e");
svg.append("rect")
   .attr("x", 0)
   .attr("y", 15)
   .attr("width", 30)
   .attr("height", 15)
   .style("fill", "#2d6ac2");
```

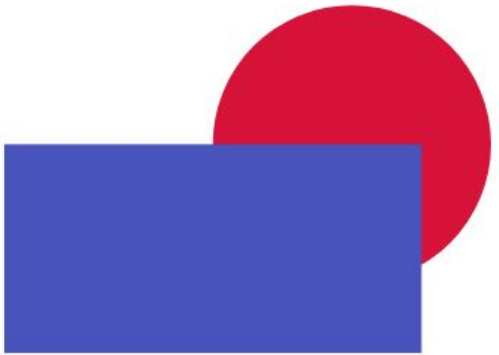


Como o D3.js "desenha" na tela?

- Podemos também usar CSS para definir a aparência dos elementos

```
.blue { fill: #d13e2e; }
.red  { fill: #2d6ac2; }
```

```
svg.append("circle")
    .attr("cx", 25)
    .attr("cy", 15)
    .attr("r", 10)
    .attr("class", "blue");
svg.append("rect")
    .attr("x", 0)
    .attr("y", 15)
    .attr("width", 30)
    .attr("height", 15)
    .style("class", "red");
```



Mapeando dados

- Agora sabemos como desenhar um retângulo

```
svg.append("rect")
```

- Gráfico de barras:
  - Um retângulo para cada elemento
  - A altura do retângulo representa o "tamanho" do número
- O D3.js já possui mecanismos para facilitar tudo isso

Seleções

- Seletores como os de CSS
- **`d3.select()`**: o primeiro elemento de um seletor

```
var svg= d3.select("svg"); // Primeiro SVG que encontrar
```

- **`d3.selectAll()`**: todos os elementos

```
var allBars= d3.selectAll("svg rect"); //Todos os rects dentro de todos os SVGs
var bars= svg.selectAll("rect"); // Todos os rects dentro de uma seleção
```

Associação de dados

- Opera sobre seleções

```
// associa cada elemento de 'dataset' a um elemento da seleção
svg.selectAll("rect").data(dataset);
```

- Como criar/atualizar/remover elementos com base nos dados?
  - **`.enter()`**
  - **`.update()`**
  - **`.exit()`**

Associação de dados

- **`.enter()`**

```
// Seleção especial que contém elementos AINDA NÃO CRIADOS
var newBars= svg.selectAll("rect")
      .data(dataset)
      .enter();
```

- Lembre-se: para criar elementos usamos **`.append()`**

```
// Cria um rect para cada dado que ainda não possui um rect associado
var newBars= svg.selectAll("rect")
      .data(dataset)
      .enter()
      .append("rect");
```

- Como fazer os atributos serem baseados nos dados? (cadê o *data-driven*?)

Associação de dados

- **`.enter()`**
- Cadê o *data-driven*?
  - **`.attr(atributo, valor)`** define o valor de um atributo
  - Se valor é uma função, o D3 a chama passando o dado e o índice do dado

```
svg.selectAll("rect")
   .data(dataset)
   .enter()
   .append("rect")
       .attr("class", "bar")
       .attr("x", function(d, i) { return i * (width/ dataset.length); })
       .attr("y", function(d) { return height- d; })
       .attr("width", (width/ dataset.length))
       .attr("height", height);
```

Associação de dados

- **`.enter()`**
- Cadê o *data-driven*?
  - **`.attr(atributo, valor)`** define o valor de um atributo
  - Se valor é uma função, o D3 a chama passando o dado e o índice do dado

```
svg.selectAll("rect")
   .data(dataset)
   .enter()
   .append("rect")
       .attr("class", "bar")
       .attr("x", function(d, i) { return i * (width/ dataset.length); })
       .attr("y", function(d) { return height- d; })
       .attr("width", (width/ dataset.length))
       .attr("height", height);
```

Associação de dados

- **`.enter()`**
- Cadê o *data-driven*?
  - **`.attr(atributo, valor)`** define o valor de um atributo
  - Se valor é uma função, o D3 a chama passando o dado e o índice do dado

```
svg.selectAll("rect")
   .data(dataset)
   .enter()
   .append("rect")
       .attr("class", "bar")
       .attr("x", function(d, i) { return i * (width/ dataset.length); })
       .attr("y", function(d) { return height- d; })
       .attr("width", (width/ dataset.length))
       .attr("height", height);
```

Associação de dados

- **`.enter()`**
- Cadê o *data-driven*?
  - **`.attr(atributo, valor)`** define o valor de um atributo
  - Se valor é uma função, o D3 a chama passando o dado e o índice do dado

```
svg.selectAll("rect")
   .data(dataset)
   .enter()
   .append("rect")
       .attr("class", "bar")
       .attr("x", function(d, i) { return i * (width/ dataset.length); })
       .attr("y", function(d) { return height- d; })
       .attr("width", (width/ dataset.length))
       .attr("height", height);
```

Associação de dados

- Outras seleções especiais
  - **.update()**

```
svg.selectAll("rect")
   .data(dataset)
   .update()
// atualizar usando .attr()
```

  - **.exit()**

```
// Remove do documento todos os elementos que perderam associação
svg.selectAll("rect")
   .data(dataset)
   .exit()
      .remove();
```

Atividade 1

- Mostrar o seguinte conjunto de dados em um gráfico de barras:

```
var dataset= [ 50, 380, 470, 110, 390, 40, 80, 190 ];
```

- Biblioteca D3 precisa ser declarada!

```
<script src="http://d3js.org/d3.v3.min.js"></script>
```

Atividade 2

- Adaptar o gráfico anterior para receber dados de um arquivo
  - **`d3.csv()`** - Comma Separated Values
  - **`d3.tsv()`** - Tab Separated Values
  - **`d3.json()`** - JavaScript Object Notation
  - **`d3.text()`** - Texto plano que pode ser codificado de algum modo particular
  - **`d3.xml()`** - Extensible Markup Language
  - **`d3.html()`** - HyperText Markup Language

https://bl.ocks.org/mbostock/3885304

Atividade 2

- *Callback* para requisição dos dados

```
function(error, dataset) {
    if (error) {
        console.log(error);
        return;
    }
    // ...
}
```

- Ao fim de operações assíncronas
- Operar dentro de outro contexto

Atividade 2

- Preparando a área de desenho

```
var margin= {top: 20, right: 30, bottom: 30, left: 40};

var width= 800- margin.left- margin.right;
var height= 600- margin.top- margin.bottom;
// Dimensoes internas do gafico
```

Atividade 2

- Preparando a área de desenho: o canvas SVG

```
var svg= d3.select("body")
       .append("svg")
       .attr("width", width+ margin.left+ margin.right)
       .attr("height", height+ margin.top+ margin.bottom)
       .append("g")
          .attr("transform",
                "translate("+ margin.left +", "+ margin.top +")");
```

Atividade 2

- Escalas
  - No nosso gráfico,

(letra, frequência) →  → **(x, y)**

- Escalas são funções que mapeiam um intervalo em outro
- O D3 oferece vários tipos de escalas (lineares, logarítmicas, etc.)

Atividade 2

- Escalas
- x

```
// Para mapear as letras de A a Z nos pixels 0 a width
var scaleX= d3.scale.ordinal()
      .rangeBands([0, width])
      .domain(dataset.map(function(d) { return d.letter; }));
// x(A) retorna 0 e x(Z) retorna 'width'
```

- y

```
var scaleY= d3.scale.linear()
      .range([height, 0])
      .domain([0, d3.max(dataset, function(d) { return d.frequency; })]);
```

Atividade 2

- Eixos
  - Não conseguimos comunicar o que os dados mapeados representam
  - Eixos servem para mostrar os valores que as escalas mapeiam

```
var xAxis= d3.svg.axis()
       .scale(scaleX)
       .orient("bottom");

var yAxis= d3.svg.axis()
       .scale(scaleY)
       .orient("left");
```

Atividade 2

- Eixos

```
svg.append("g")
   .attr("class", "x axis")
   .attr("transform", "translate(0, " + height + ")")
   .call(xAxis);

svg.append("g")
   .attr("class", "y axis")
   .call(yAxis)
   .append("text")
      .attr("transform", "rotate(-90)")     // Label do eixo
      .attr("y", 5)
      .attr("dy", ".75em")
      .style("text-anchor", "end")
      .text("Frequency");
```

Atividade 2

- Um pouco de CSS

```
.axis path,
.axis line {
   fill: none;
   stroke: #000;
   shape-rendering: crispEdges;
}

.bar:hover {
   fill: #000;
}

.bar {
   cursor: pointer;
}
```

Relembrando...

- No Terminal, acesse o diretório em que o *index.html* se encontra e utilize

```
python -m SimpleHTTPServer
```

- No navegador

```
localhost:8000
```

Obrigado pela atenção!

Visualização de Informação com D3.js

Evandro Ortigossa
evortigosa@gmail.com