Data-Driven Documents

Lucas Pagliosa

Aula de Visualização - Parte Teórica

9 de setembro de 2014

Sumário

Motivação

Mercado de trabalho:

- **Na empresa:** programação Web e mobile
- **Na academia:** papers com serviço cliente-servidor, disponibilização do cógido
- Em comum: visualização de dados:
  - Comparação entre resultados/técnicas
  - Porcentagem de lucro
  - Análise de vendas
  - Previsão de mercado

Motivação

É comum a necessidade de visualizar dados tanto nas empresa quanto nas universidades

Motivação

É comum a necessidade de visualizar dados tanto nas empresa quanto nas universidades

Problemas

Grande quantidade de dados;
Muitos atributos a serem visualizados;
Espaço visual limitado;

Motivação

É comum a necessidade de visualizar dados tanto nas empresa quanto nas universidades

Problemas

Grande quantidade de dados;
Muitos atributos a serem visualizados;
Espaço visual limitado;

Lado positivo

Os navegadores Web estão mais rápidos e melhores;
Depuração de código na Web;
HTML5 com maior suporte gráfico

Motivação

http://bl.ocks.org/mbostock/1557377

http://bl.ocks.org/sxv/4491174

http://bl.ocks.org/mbostock/7881887

http://bl.ocks.org/mbostock/4063663

http://bl.ocks.org/mbostock/929623

http://www.zcliu.org/archives/immens

http://www.nanocubes.net/

Sumário

Web browser

O Web browser é a interface criada para entrada e saída de dados na World Wide Web. Estes dados podem ser um vídeo, imagem, documento ou Web page

Web browser

O Web browser é a interface criada para entrada e saída de dados na World Wide Web. Estes dados podem ser um vídeo, imagem, documento ou Web page

Uma página Web é um documento de marcação formado por elementos, estes constituidos por tags e atributos

Web browser

O Web browser é a interface criada para entrada e saída de dados na World Wide Web. Estes dados podem ser um vídeo, imagem, documento ou Web page
Uma página Web é um documento de marcação formado por elementos, estes constituidos por tags e atributos

```
<tag> attribute </tag>
```

Elementos são bem definidos e sua sintaxe faz parte da linguagem HTML

Web browser

As informações sobre layout ou estilo da página é comumente atribuido a arquivos CSS (Cascade Style Sheet):

```
.axis text
{
  font-family: sans-serif;
  font-size: 15px;
}

body
{
  font-family: Palatino Linotype;
  font-size: 18px;
  background-color: #FFFFFF
}
```

Web browser

Exemplo simples de uma página Web:

```
<!DOCTYPE html>
<html>
  <head>
    <title>This is a title</title>
  </head>
<body>
  <p>Hello world!</p>
 </body>
</html>
```

Web browser

Exemplo simples de uma página Web:

```
<!DOCTYPE html>
<html>
  <head>
    <title>This is a title</title>
  </head>
<body>
  <p>Hello world!</p>
 </body>
</html>
```

Os documentos de marcação formam naturalmente uma hierárquia de elementos

Web browser

Exemplo simples de uma página Web:

```
<!DOCTYPE html>
<html>
  <head>
    <title>This is a title</title>
  </head>
<body>
  <p>Hello world!</p>
 </body>
</html>
```

Os documentos de marcação formam naturalmente uma hierárquia de elementos

Uma página Web é formada estaticamente por elementos HTML

DOM - Document Object Model

- Antigamente, cada browser implementava seu próprio modo de manipulação dos elementos de uma página Web

DOM - Document Object Model

- Antigamente, cada browser implementava seu próprio modo de manipulação dos elementos de uma página Web
- Inconsistência de códigos

DOM - Document Object Model

- Antigamente, cada browser implementava seu próprio modo de manipulação dos elementos de uma página Web
- Inconsistência de códigos
- W3C (World Wide Web Consortium) utilizou a especificação DOM para padronizar a programação Web

DOM - Document Object Model

- Antigamente, cada browser implementava seu próprio modo de manipulação dos elementos de uma página Web
- Inconsistência de códigos
- W3C (World Wide Web Consortium) utilizou a especificação DOM para padronizar a programação Web
- DOM é uma API que fornece uma maneira padrão de se acessar, criar e modificar elementos em um documento XML

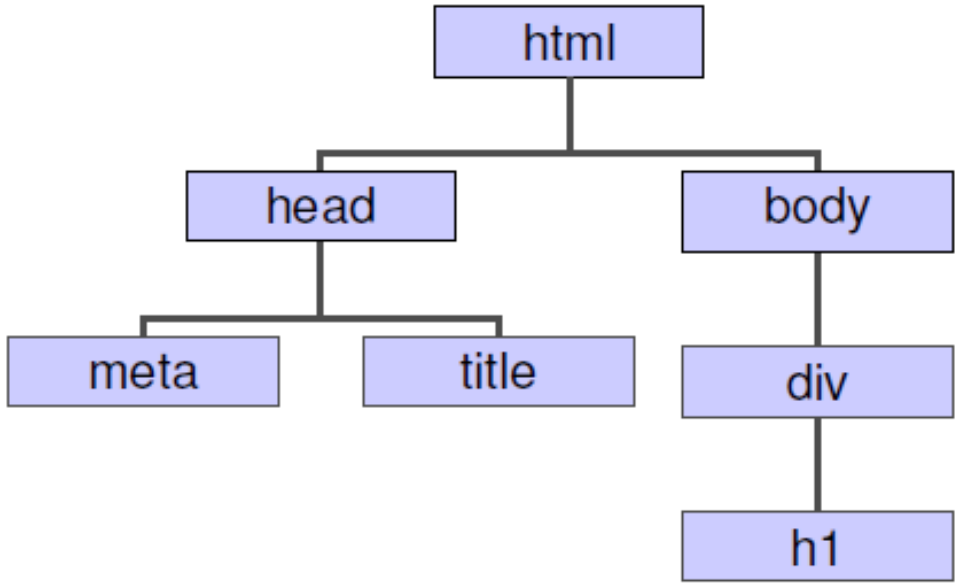
DOM - Document Object Model

- Antigamente, cada browser implementava seu próprio modo de manipulação dos elementos de uma página Web
- Inconsistência de códigos
- W3C (World Wide Web Consortium) utilizou a especificação DOM para padronizar a programação Web
- DOM é uma API que fornece uma maneira padrão de se acessar, criar e modificar elementos em um documento XML
- DOM organiza a estrutura de uma página na forma de uma árvore hierárquica

DOM - Document Object Model

```
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <title>Example</title>
  </head>
  <body>
  <div class="container">
    <h1>Some header</h1>
  </div>
```

(a)



(b)

Figura: Código em HTML (a) e sua respectiva árvore gerada pelo DOM (b)

JavaScript

JavaScript é a linguagem responsável pela dinamização das páginas Web

- . Criação de funções e objetos
- Parte lógica da página
- Controle de eventos e ações de iteração com o usuário

```
<script type="text/javascript">
  // Press G
  $(document).bind('keydown', function(e) {
    if (e.keyCode === 71 && canCreateGrid)
      createGrid("area");
  });
</script>
```

Sumário

JavaScript

- Apesar do nome, nenhuma relação com Java

JavaScript

- Apesar do nome, nenhuma relação com Java
- Permite manipulação dinâmica do DOM, portanto, de uma página Web

JavaScript

- Apesar do nome, nenhuma relação com Java
- Permite manipulação dinâmica do DOM, portanto, de uma página Web

Exemplo:

```
<script type="text/javascript">
  var w = 700;
  var h = 400;
  var padding = 35;
  var dataset;
  var svg = d3.select("body").append("svg")
      .attr("width", w)
      .attr("height", h)
      .append("g");
</script>
```

Algumas observações:

"Orientada a Objetos"

Algumas observações:

"Orientada a Objetos"

Principais componentes lógicos: vetores e objetos

JavaScript

Algumas observações:

"Orientada a Objetos"

Principais componentes lógicos: vetores e objetos

Não cria threads, apenas processamento assíncrono

Frameworks

Assim como qualquer linguagem, existem frameworks com métodos prontos que ajudam, na maioria das vezes, na otimização e implementação do seu programa

Frameworks

Assim como qualquer linguagem, existem frameworks com métodos prontos que ajudam, na maioria das vezes, na otimização e implementação do seu programa

Exemplos:

- jQuery.js - Muitas funcionalidades de interface, várias outros plugins usam jQuery
- numeric.js - Matemático
- d3.js - Visualização de dados
- croosfilter.js - Agragação de dados
- leaflet.js - Mapas
- etc

D3

D3 é uma ferramenta que permite a criação de gráficos (dos mais variados) e a sua associação dinâmica com dados. O elemento HTML mais usado internamente no D3 se chama *Scalable Vector Graphics* (SVG)

SVG permite o desenho vetorial de qualquer objeto possível

D3

D3 é uma ferramenta que permite a criação de gráficos (dos mais variados) e a sua associação dinâmica com dados. O elemento HTML mais usado internamente no D3 se chama *Scalable Vector Graphics* (SVG)

SVG permite o desenho vetorial de qualquer objeto possível

Elementos:

- path
- rect
- circle
- line
- marker
- etc

D3

D3 é uma ferramenta que permite a criação de gráficos (dos mais variados) e a sua associação dinâmica com dados. O elemento HTML mais usado internamente no D3 se chama *Scalable Vector Graphics* (SVG)

SVG permite o desenho vetorial de qualquer objeto possível

Elementos:

- path
- rect
- circle
- line
- marker
- etc

Além disso, D3 possui outras funcionalidades interessantes como uma eficiente implementação de molas e colisão de partículas

Sumário

Introdução a D3

D3 possui muitas funções interessantes e otimizadas como:
quadritrees
construção e manipulação de grafos
colisão de partículas

Introdução a D3

D3 possui muitas funções interessantes e otimizadas como:
quadritrees
construção e manipulação de grafos
colisão de partículas

Porém, o uso mais comum de D3 envolve visualização de gráficos na Web

Introdução a D3

D3 possui muitas funções interessantes e otimizadas como:
quadritrees
construção e manipulação de grafos
colisão de partículas

Porém, o uso mais comum de D3 envolve visualização de gráficos na Web

Em geral, as funções relacionadas a construção e manipulação de gráficos implementadas em D3 nada mais são simplificações do uso "direto" em JavaScript

Selection: função responsável em identificar e obter elemento(s) no DOM

```
// JavaScript
var paragraphs = document.getElementsByTagName("p");
for (var i = 0; i < paragraphs.length; i++) {
  var paragraph = paragraphs.item(i);
  paragraph.style.setProperty("color", "white", null);
}

// D3
d3.selectAll("p").style("color", "white");
}
```

Introdução a D3

Selection: função responsável em identificar e obter elemento(s) no DOM

```
// JavaScript
var paragraphs = document.getElementsByTagName("p");
for (var i = 0; i < paragraphs.length; i++) {
  var paragraph = paragraphs.item(i);
  paragraph.style.setProperty("color", "white", null);
}

// D3
d3.selectAll("p").style("color", "white");
}
```

No entanto, é preferível, em alguns casos, usar o próprio JavaScript ao invés de D3

Introdução a D3

D3 vincula dados dinamicamente ao DOM, criando e modificando elementos HTML. As três operações bases do D3 são:

- **Enter:** adiciona elementos no DOM

```
d3.select("body").selectAll("p")
  .data([4, 8, 15, 16, 23, 42])
  .enter().append("p")
  .text(function(d) { return "I'm number " + d + "!"; });
```

- **Update:** modifica elementos no DOM

```
d3.select("body").selectAll("p")
  .data([-4, -8, -15, -16, -23, -42])
  .text(function(d) { return "I'm number " + d + "!"; });
```

- **Exit:** remove elementos no DOM

```
d3.select("body").selectAll("p")
  .data([4, 8, 15, 16, 23]).exit().remove();
```

Introdução a D3

Alterações no DOM é o reflexo de alterações dinâmicas na página Web

- **Enter:**

```
<body> // dataset = [1, 2];
  <p>1</p>
  <p>2</p>
</body>
```

- **Update:**

```
<body> // dataset = [1, -2];
  <p>1</p>
  <p>-2</p>
</body>
```

- **Exit:**

```
<body> // dataset = [1];
  <p>1</p>
</body>
```

Introdução a D3

Transiction: função responsável por modificar elemento(s) no DOM ao longo do tempo (animação)

Lembrando

Páginas Web dispõem de apenas uma única thread

Introdução a D3

Transiction: função responsável por modificar elemento(s) no DOM ao longo do tempo (animação)

Lembrando

Páginas Web dispõem de apenas uma única thread

JavaScript é baseado em eventos

Introdução a D3

Transiction: função responsável por modificar elemento(s) no DOM ao longo do tempo (animação)

Lembrando

Páginas Web dispõem de apenas uma única thread

JavaScript é baseado em eventos

Funções assíncronas

setTimeout(function, milliseconds) - Executa a função uma única vez
setInterval(function, milliseconds) - Executa a função repetidas vezes a cada intervalo

Introdução a D3

Transiction: função responsável por modificar elemento(s) no DOM ao longo do tempo (animação)

Lembrando

Páginas Web dispõem de apenas uma única thread

JavaScript é baseado em eventos

Funções assíncronas

setTimeout(function, milliseconds) - Executa a função uma única vez
setInterval(function, milliseconds) - Executa a função repetidas vezes a cada intervalo

!

Por possuir apenas umas thread, a página escalona as funções assíncronas, dando a impressão de paralelismo

Introdução a D3

Exemplo: criando circulos

```
var dataset = [{"id": 0, "x": 20, "y": 20, "label": "icmc"}];
var canvas = d3.select("body")
  .append("div")
  .append("svg")
  .attr("width", width + "px")
  .attr("height", height + "px");

canvas.selectAll("circle").data(dataset)
  .enter()
  .append("circle")
  .attr("id", function(d) {return d.id;})
  .attr("cx", function(d) {return d.x;})
  .attr("cy", function(d) {return d.y;})
  .attr("r", radius)
  .attr("fill", randColor());
```

Exemplo: criando circulos

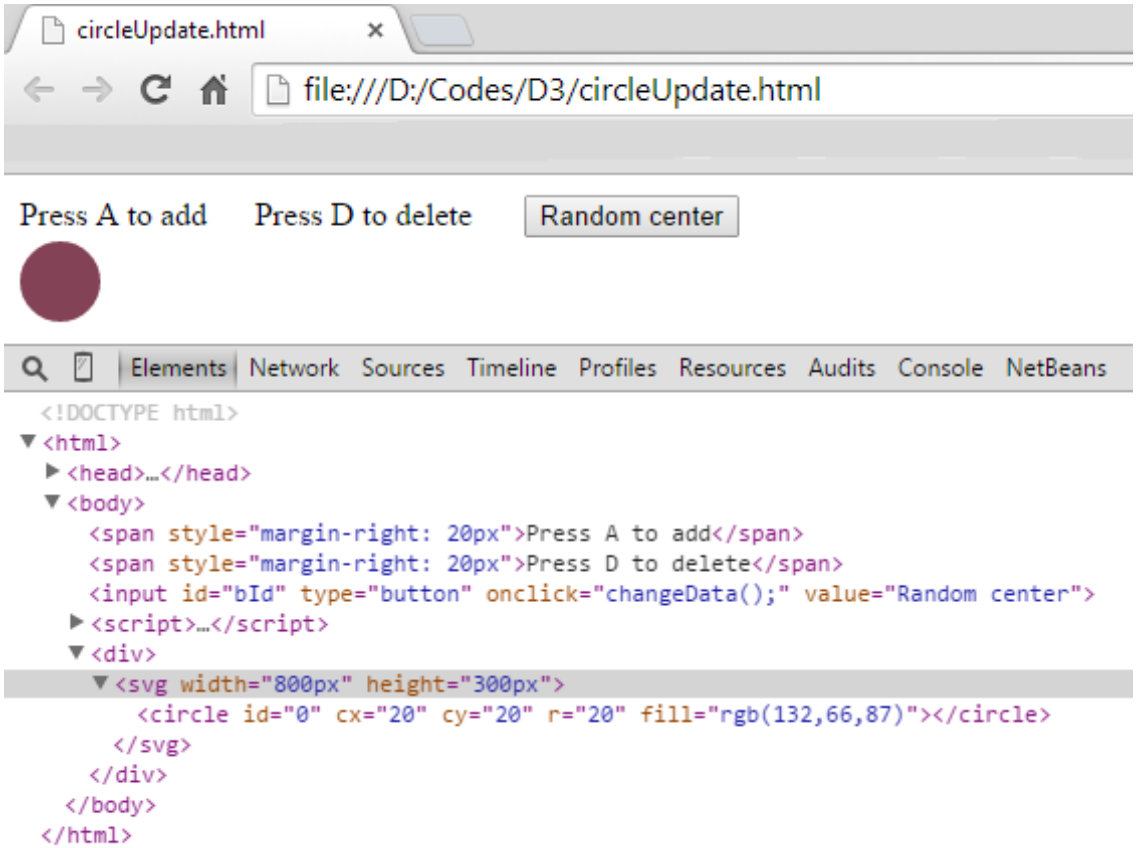


Figura: Resultado do código anterior

Exemplo: modificando atributos do elemento “suavemente”

Timers no D3

A função transition() implementa setTimeout()

Exemplo: modificando atributos do elemento “suavemente”

Timers no D3

A função transition() implementa setTimeout()

```
canvas.selectAll("circle").data(dataset)
  .transition()
  .delay(500)
  .duration(500)
  .attr("cx", function(d) {return d.x;})
  .attr("cy", function(d) {return d.y;});
```

Timers no D3

d3.ease(type[, arguments...]) informa qual o método de interpolação usado na transição:

linear - the identity function, t
poly(k) - raises t to the specified power k (e.g., 3)
quad - equivalent to poly(2)
cubic - equivalent to poly(3)
sin - applies the trigonometric function sin
exp - raises 2 to a power based on t
circle - the quarter circle
elastic(a, p) - simulates an elastic band; may extend slightly beyond 0 and 1
back(s) - simulates backing into a parking space
bounce - simulates a bouncy collision

`https://github.com/mbostock/d3/wiki/Transitions#d3_ease`

Sumário

```
file:///D:/Codes/D3/circleUpdate.html
```

```
http://blog.visual.ly/
creating-animations-and-transitions-with-d3-js/
```

The End

The End

Perguntas??