matfyz

Computer Graphics Charles University

Data Driven Documents – základní principy

© 2015 Josef Pelikán, CGG MFF UK Praha

http://cgg.mff.cuni.cz/~pepca/

pepca@cgg.mff.cuni.cz

DDD, D3

- **Data-driven documents**
  - moderní přístup k zobrazování dat na WWW
  - data nejsou jen zobrazována (renderována) WWW stránkou, ale jsou přímo integrována do HTML objektů
- **W3C DOM**
  - Document Object Model (w3c.org)
  - HTML, XHTML, XML … jsou hierarchické formáty
  - HTML5 prohlížeče umožňují elegantně k jednotlivým uzlům přistupovat (např. přes JavaScript)

Zobrazování grafiky v HTML5

- **SVG**
  - Scalable Vector Graphics
  - také pod patronátem W3C
- **grafické objekty (primitiva)**
  - rect, circle, line, ..
  - Snadno přístupné parametry formou XML atributů

```
<circle cx="250" cy="25" r="25"/>
```

- možnost použití **CSS** pro definici vzhledu (stylu)

```
<circle cx="25" cy="25" r="22" class="pumpkin"/>
```

Knihovna D3 (JavaScript)

- velmi populární JS knihovna
  - **transformace**, ne nové reprezentace!
  - dobře dokumentovaná a udržovaná (d3js.org)
- přístupy (standardy!)
  - **W3C DOM** (Document Object Model)
  - **SVG grafika**, CSS
  - **Data binding**, **Data joins**
  - **W3C Selectors API**
- **estetika, interaktivita**
  - animace („transitions“)

Principy

- **mapování** dat na DOM elementy
  - každý prvek visualizace odpovídá nějakému DOM elementu
- **dynamické úpravy** WWW stránky
  - dynamické visualizace (simulace, realtime) nebo interakce uživatele
  - je třeba upravovat DOM
  - též přidávat a ubírat elementy!
- **… Data-Driven Documents**
  - proto jméno D3

Standardy – HTML5

- HTML5
  - mnoho zdrojů pro studium
- minimální platný HTML5 dokument

```
<!DOCTYPE html><title/>x
```

- stručný HTML dokument

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title>Simple valid HTML5 page</title>
 </head>
 <body>
  <p>Paragraph..</p>
 </body>
</html>
```

Standardy – SVG

- Scalable Vector Graphics
  - W3C standard
  - http://www.w3.org/Graphics/SVG/

- stručná HTML5 stránka se SVG grafikou

```
<!DOCTYPE html>
<meta charset="utf-8">
<title>SVG hello</title>
<svg width="800" height="400">
 <text y="12">
  Hello, world!
 </text>
</svg>
```

Standardy – CSS

- Cascading Style Sheets
  - W3C standard (CSS 2.2)
  - http://dev.w3.org/csswg/css2/
  - nepřidává obsah, pouze definuje styly zobrazení
- stručná HTML5 stránka s CSS stylem

```
<!DOCTYPE html>
<meta charset="utf-8">
<title>CSS hello</title>
<style>
body { background: steelblue; }
</style>
<body>
Hello, world!
</body>
```

Standardy – CSS selectors

- koncept hojně používaný v CSS
  - W3C doporučení
  - http://www.w3.org/TR/selectors/
  - zobrazení [výraz, element] → boolean
- příklady

```
.shrek { color: green; }
div#menubar { font-weight: bold; }
a:visited { color: #F28000; }
html:lang(de) { .. }
tr:nth-child(odd) { /* every odd row of a table */ }
body > p { /* p is immediate child of body */ }
h1.opener + h2 { /* adjacent siblings */ }
h1 ~ pre { /* general siblings */ }
```

JavaScript

- skriptování na straně WWW prohlížeče
  - ECMAScript je standardem
  - http://www.ecmascript.org/
- příklad

```
<!DOCTYPE html>
<meta charset="utf-8">
<title>JavaScript hello</title>
<body>
<script>
console.log("Hello, world!");
</script>
</body>
```

D3.js knihovna

- jádro knihovny v jediném .js souboru
  - je potřeba stránku opatřit vlastním scriptem
- šablona stránky používající d3.js

```
<!DOCTYPE html>
<meta charset="utf-8">
<title>d3.js template</title>
<style>
 /* my CSS */
</style>
<body>
<script src="http://d3js.org/d3.v2.js"></script>
<script>
 /* my JavaScript */
</script>
</body>
```

D3.js zdroje

http://d3js.org/

- **D3 API reference**
  - https://github.com/mbostock/d3/wiki/API-Reference
- **D3 WiKi**
  - https://github.com/mbostock/d3/wiki
- **D3 Google group**
  - https://groups.google.com/group/d3-js
- **D3 Stack Overflow**
  - http://stackoverflow.com/questions/tagged/d3.js

Selectors API

- **výběr („selection“)**
  - … viz jQuery, CSS
  - pole HTML uzlů
  - bohatě vybavené funkce vybírající podle typu uzlu, atributů, „id“, „name“, „class“
- příklady:

```
#foo          // <any id="foo">
foo           // <foo>
.foo          // <any class="foo">
[foo=bar]     // <any foo="bar">
foo bar       // <foo><bar></foo>
foo.bar       // <foo class="bar">
foo#bar       // <foo id="bar">
```

Selectors API

- selektor podle W3C standardu:
  - `document.querySelectorAll("#foo")`
  - nejsou tam selekce, musí se iterovat přes výsledek
- **`d3.select("#foo")`**
  - vybírá jeden uzel (první, který splňuje dané podmínky)
  - zachová globální selekci (selectAll())
- **`d3.selectAll("#foo")`**
  - vybere všechny uzly podle dané podmínky

Práce se selekcí

- **selekce je pole**
  - snadno se mění hromadně vlastnosti všech prvků selekce

```
// select all <circle> elements
var circle = d3.selectAll("circle");

// set some attributes and styles
circle.attr("cy", 12);
circle.attr("r", 24);
circle.style("fill", "red");
```

Řetězení metod

- … elegantnější a úspornější zápis

```
// select all <circle> elements
// and set some attributes and styles
d3.selectAll("circle")
  .attr("cy", 12);
  .attr("r", 24);
  .style("fill", "red");
```

selection.append

- přidávání obsahu za vybrané elementy
  - … opět možnost hromadné operace

```
// select the <body> element
var body = d3.select("body");

// add an <h1> element
var h1 = body.append("h1");
h1.text("Hello!");
```

Hromadné příklady

- stejně jednoduchý zápis:

```
// select all <section> elements
var section = d3.selectAll("section");

// add an <h1> element to each
var h1 = section.append("h1");
h1.text("Hello!");
```

- komplikovanější příklad:

```
var h1 = d3.selectAll("section")
    .style("background", "steelblue")
  .append("h1")
    .text("Hello!");
```

Data

- Co když potřebuji přidávat **mnoho elementů** ?
  - lepší je na to jít přes data..
- data jsou také **pole**
  - … stejně jako selekce …

```
// A bar chart, perhaps?
var data1 = [1, 1, 2, 3, 5, 8];

// A scatterplot, perhaps?
var data2 = [ {x: 10.0, y: 9.14},
              {x:  8.0, y: 8.14},
              {x: 13.0, y: 8.74},
              {x:  9.0, y: 8.77},
              {x: 11.0, y: 9.26} ];
```

Data binding

- **datové pole:**
  - libovolné pole hodnot (čísel, JS objektů, dalších polí, ...)
  - přímo se dá v kódu psát v JSON formátu
  - umí načítat velké množství formátů (JSON, CSV, ..)
- **plán:** jeden prvek datového pole se přiřadí jednomu uzlu selekce
  - elegantně ošetřen nedostatek nebo přebytek uzlů
  - integrace dat do HTML („__data__“)

Data joins

- **enter**
  - přidává nové elementy, když je jich málo (méně než prvků pole)
- **exit**
  - ošetření opačné situace
  - nejčastěji se přebytečné HTML elementy prostě odstraní..
- **update**
  - existující elementy se upraví podle nových přiřazených dat

Příklad – bodový graf

- datové položky obsahují souřadnice [x, y]:

```
Var circles = svg.selectAll("circle")
    .data(data2);

circles.enter()          // new data elements
  .append("circle")
    .attr("cx", x)
    .attr("cy", y)
    .attr("r", 2.5);

Circles                  // updated (existing) elements
    .attr("cx", x)
    .attr("cy", y)
    .attr("r", 5);
```

Zkratka: enter + update

- „append“ přidává existující položky k těm novým
  - pokud se mají všechny (staré i nové) nastavit stejně, kód se zjednoduší

```
svg.selectAll("circle")
    .data(data2)
.enter()                  // new data elements
.append("circle")         // existing+new elements
    .attr("cx", x)
    .attr("cy", y)
    .attr("r", 2.5);
```

Data join pomocí klíče

- implicitně se operace join dělá přes index
  - ve zvláštních případech (dynamická data) potřebujeme jiný klíč

```
var data3 = [
  { name: "Alice", x: 10.0, y: 9.14 },
  { name:   "Bob", x:  8.0, y: 8.14 },
  { name: "Carol", x: 13.0, y: 8.74 },
  { name:  "Dave", x:  9.0, y: 8.77 },
  { name: "Edith", x: 11.0, y: 9.26 }
];
```

Klíčová funkce

- rozšířená vatianta operace „data(d,k)“

```
function key(d) { return d.name; }

var circle = svg.selectAll("circle")
    .data(data, key)
    .attr("cx", x)
    .attr("cy", y)
    .attr("r", 2.5);
```

Datové soubory

- externí data: CSV, TSV, JSON, ..
  - pohodlné načítání mnoha formátů
  - často se musí konvertovat datum, někdy i čísla..
- příklad s CSV formátem (stocks.csv):

```
symbol,date,price
S&P 500,Jan 2000,1394.46
S&P 500,Feb 2000,1366.42
S&P 500,Mar 2000,1498.58
S&P 500,Apr 2000,1452.43
S&P 500,May 2000,1420.6
S&P 500,Jun 2000,1454.6
S&P 500,Jul 2000,1430.83
```

Připojování CSV dat

- je třeba převést řetězce na čísla a převést datumy

```
var format = d3.time.format("%b %Y");

d3.csv("stocks.csv", function(stocks) {
  stocks.forEach(function(d) {
    d.price = +d.price;    // coercion from string
    d.date  = format.parse(d.date);
                           // we need JS date here
  });
});
```

Připojování JSON dat

- pořád je potřeba převádět datumy

```
var format = d3.time.format("%b %Y");

d3.json("stocks.json", function(stocks) {
  stocks.forEach(function(d) {
    d.date  = format.parse(d.date);
                              // we need JS date here
  });
});
```

Další operace s daty

- samotný JavaScript obsahuje mnoho užitečných metod pro práci s poli
  - array.filter(), array.map(), array.sort(), ..
- další transformace dat v knihovně d3
  - d3.nest(), d3.keys(), d3.values(), ..

Měřítka (Scales)

- převod souřadných systémů
  - číslo → číslo
  - lineární měřítko, logaritmické, apod.
- mapování dat na atributy visualizace
  - barva, tloušťka čáry, ..
- **.. Scales**

Domain → Range

- zobrazení z prostoru vstupních dat do prostoru visualizace
  - lze napsat vlastní funkci:

```
function x(d) {
  return d *42 + “px“;
}
```

- kvantitativní měřítka, např. lineární:

```
var x = d3.scale.linear()
  .domain([12, 24])
  .range([0, 720]);

// x(16) == 240
```

Další kvantitativní měřítka

- d3.scale.log()
- d3.scale.sqrt()
- **interpolátory**
  - např. interpolace barvy

```
var x = d3.scale.linear()
    .domain([12, 24])
    .range(["steelblue", "brown"]);
```

- alternativní interpolátory (HSL barva..)

```
    .interpolate(d3.interpolateHsl);
```

Po částech lineární převod

- definiční obor a obor hodnot mohou mít více uzlových bodů:

```
var x = d3.scale.linear()
    .domain([-10, 0, 100])
    .range(["red", "white", "green"]);

x(-5); // #ff8080
x(50); // #80c080
```

Diskrétní měřítka

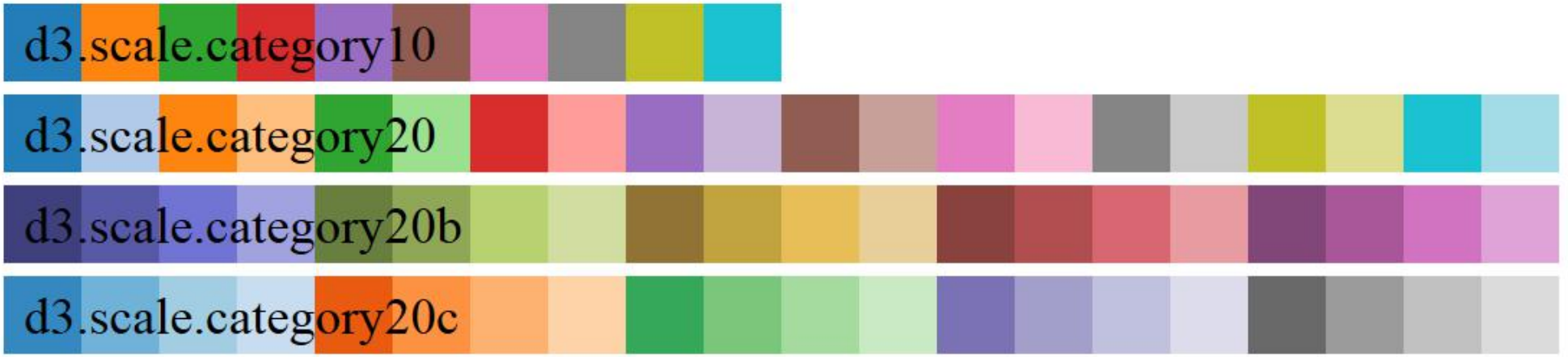
- je to vlastně pouhé diskrétní zobrazení:

```
var x = d3.scale.ordinal()
    .domain(["A", "B", "C", "D"])
    .range([0, 10, 20, 30]);

x("B"); // 10
```

- barevné palety:

```
var x = d3.scale.category20()
    .domain(["A", "B", "C", "D"]);

x("B"); // #aec7e8
```

Barevné škály



- další pěkné sady barev: ColorBrewer
  - http://colorbrewer2.org/

Osy grafů

- pohodlné vykreslování os
- označování: čárky, popisky

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

- vykreslení v rámci selekce <g>

```
svg.append("g")
    .attr("class","y axis")
    .call(yAxis);
```

Čárky na osách (Ticks)

- kvantitativní měřítka obsahují mechanismy pro umisťování čárek
- můžeme požádat o množství čárek v rámci celé škály
  - je to jen doporučení

```
var x = d3.scale.linear()
    .domain([12, 24])
    .range([0, 720]);

x.ticks(5);    // [12, 14, 16, 18, 20, 22, 24]
```

Příklad rozvržení prostoru grafu

- souřadný systém: vlevo nahoře je počátek
- okraje (margins) se obvykle použijí pro dekorace (osy)

```
var svg = d3.select("body").append("svg")
    .attr("width",  outerWidth)
    .attr("height", outerHeight);

var g = svg.append("g")
    .attr("transform", "translate("
      + marginLeft + ","
      + marginTop + ")");
```

SVG tvary

- rect
- circle
- line
- text
- path!
  - komplikovanější popis tvaru
  - vlastní jazyk
  - lomené čáry, splines, ..

Generátory cest (Paths)

- d3.svg.line (lomená čára)

```
var x = d3.scale.linear(),
    y = d3.scale.linear();

var line = d3.svg.line()
    .x(function(d) { return x(d.x); })
    .y(function(d) { return y(d.y); });
```

- připojení bez pomoci data-join:

```
svg.append("path")
    .datum(objects)
    .attr("class", "line")
    .attr("d", line);
```

d3.svg.area

- např. pro plošné grafy funkcí (x, y_0, y_1)

```
var x = d3.scale.linear(),
    y = d3.scale.linear();

var area = d3.svg.area()
    .x(function(d) { return x(d.x); })
    .y0(height)
    .y1(function(d) { return y(d.y); });
```

Interpolace čar

- lineární
  - oblasti s teplejší barvou (žlutá, červená)
- schody
  - před hodnotou
  - za hodnotou
- spline
  - basis
  - cardinal
  - monotone (nepřekmitává)

d3.svg.arc

- pro koláčové grafy

```
var sampleArc = {
  "innerRadius": 0,
  "outerRadius": 360,
  "startAngle": 0,     // 12 o'clock
  "endAngle": 1.2      // radians
};

var arc = d3.svg.arc()
    .innerRadius(0)
    .outerRadius(360);

var pie = d3.layout.pie(); // default pie layout
var myArcs = pie(numbers); // data for d3.svg.arc
```

Literatura

- http://d3js.org/ – d3.js homepage
- Scott Murray: ***Interactive Data Visualization for the Web***, O'Reilly, free online (2015)
- M. Bostock's blog: http://bl.ocks.org/mbostock/