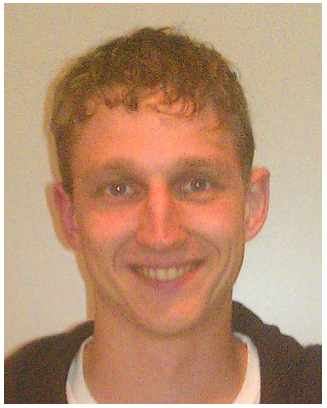
Data Visualisation Practical
Intro to D3

Thomas Höllt

Practical Outline I



Niels de Hoon

Nicola Pezotti

Thomas Höllt

datavis@graphics.tudelft.nl

Practical Outline II

- Session 1 (today)
  - D3 Crash Course, several small exercises on the way
  - Build a scatterplot
- Session 2
  - Build a geospatial plot
  - Build an angular histogram
- Session 3
  - Putting it all together
    link your visualisations to a small analysis framework

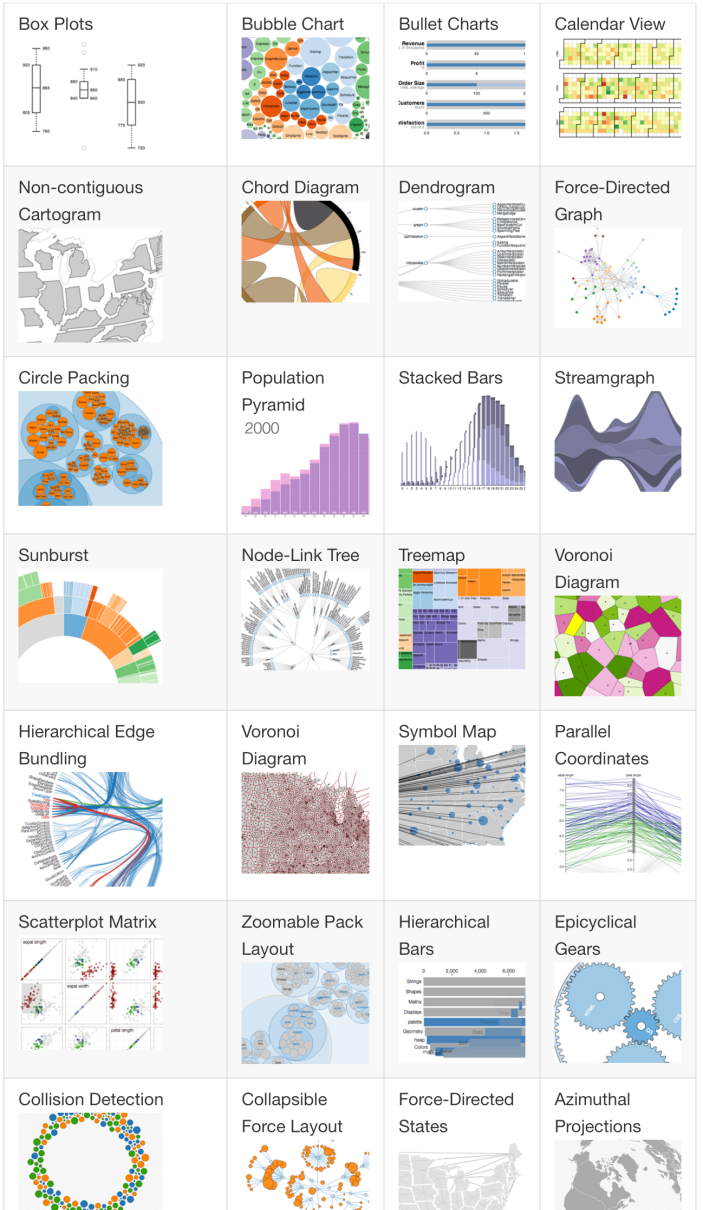
What is D3.js?

D3 is a small, free, powerful javascript library for manipulating documents based on data.

http://www.d3js.org/

What is D3.js?

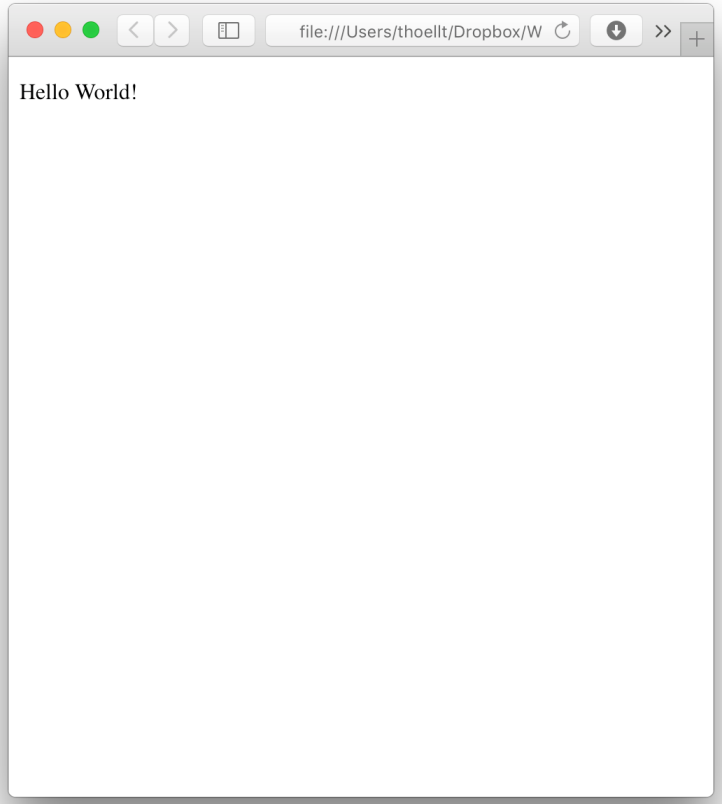
- Not a drawing library!
- allows binding arbitrary data to the Document Object Model (DOM)
  - Data Driven Documents
- D3 relies on web standards
  - HTML, CSS, Javascript, SVG
    http://www.w3schools.com



HTML

- Basic building blocks for websites
  http://www.w3schools.com/html/

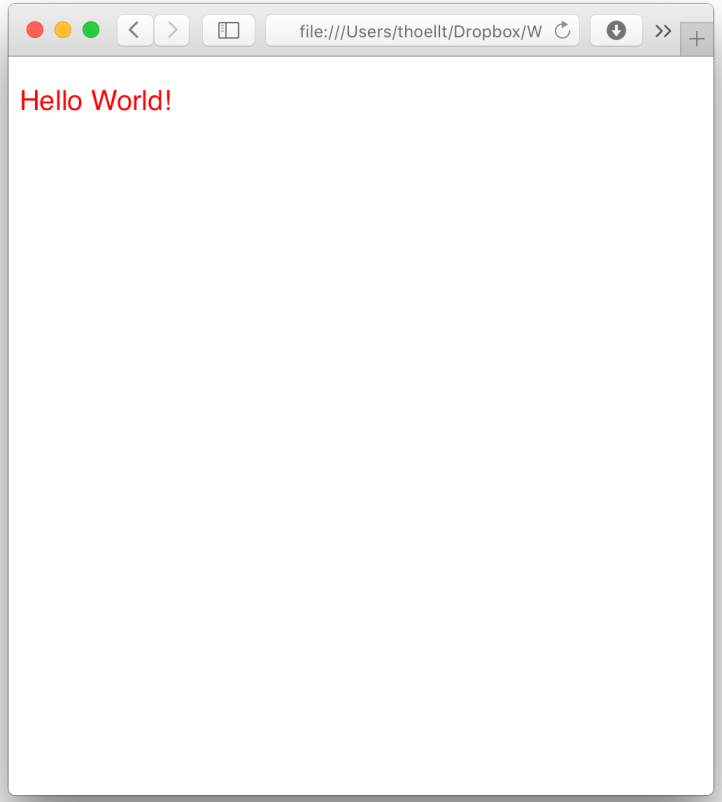
```
<!DOCTYPE html>
<html>
    <head>
        <title>Hello World!</title>
    </head>
    <body>
        <p>Hello World!</p>
    </body>
</html>
```



CSS

- Style HTML
  http://www.w3schools.com/css/

```
p {
    color: red;
    font-family: "Helvetica";
    font-size: 20px;
}
```



CSS

- Inline
  Hello World!

- Style Tag

  <style>
  p { color: red; }
  </style>

  goes into <head>

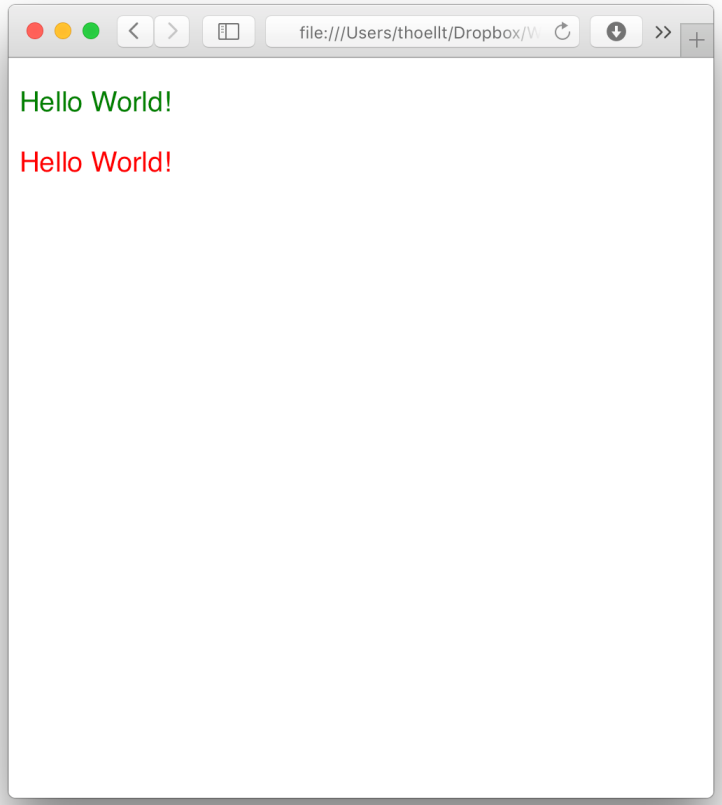
- Include
  <link href=style.css rel=stylesheet>

  goes into <head>

CSS

```
<body>
    <p class="green-par">Hello World!</p>
    <p id="red-par">Hello World!</p>
</body>
```

```
p {
    color: black;
    font-family: "Helvetica";
    font-size: 20px;
}
.green-par { color: green; }
#red-par { color: red; }
```



CSS

```
<body>
    <p class="green-par">Hello World!</p>
    <p id="red-par">Hello World!</p>
</body>
```

```
p {
    color: black;
    font-family: "Helvetica";
    font-size: 20px;
}
.green-par { color: green; }     <- class, use for multiple objects
#red-par { color: red; }         <- id, use once per doc
```

Javascript

- Modify HTML (client-side)
  http://www.w3schools.com/js/

- JS HTML DOM:
  - document.getElementsByClassName(name)

  - document.createElement(element)
  - document.removeChild(element)

  - element.innerHTML = new html content
  - element.setAttribute(attribute, value)

Javascript

- Script Tag

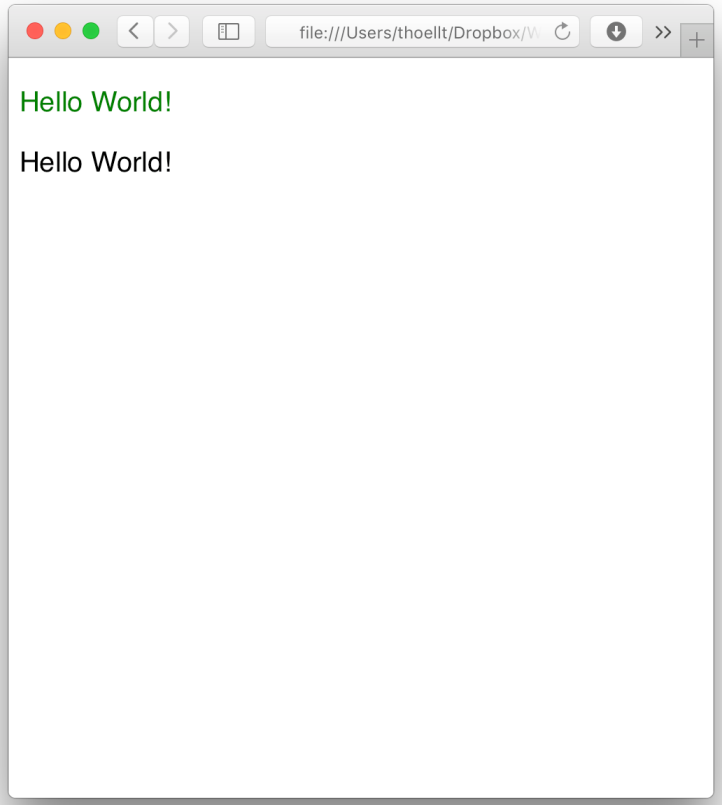
```
<script>
    p { color: red; }
</script>
```

- Include
  `<script src="scripts.js"></script>`

- Add scripts after your html code (i.e. just before </body>)
  to make sure html objects are available!

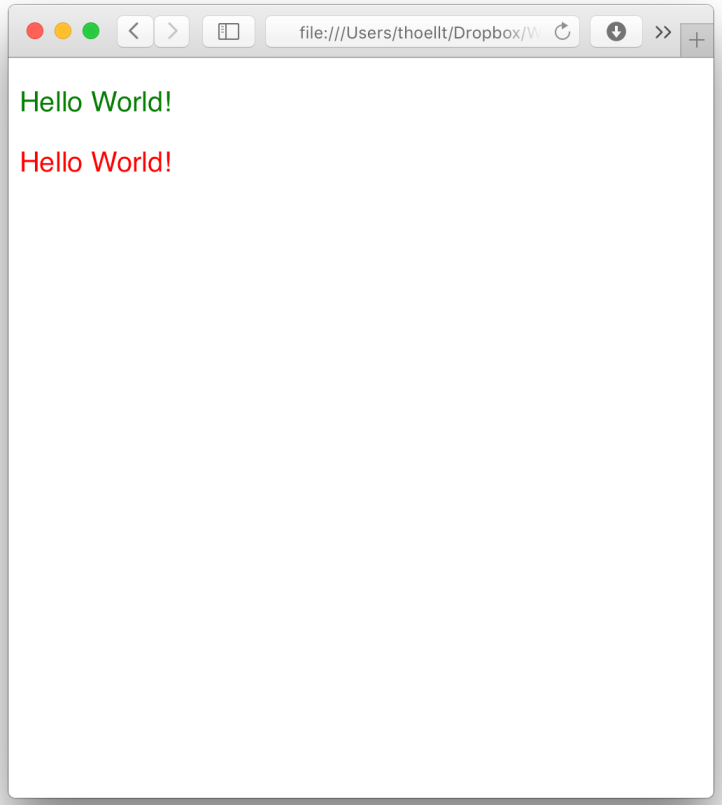
Javascript

```
<body>
    <p class="green-par">Hello World!</p>
    <p id="red-par">Hello World!</p>
</body>

.green-par { color: green; }
.red-par { color: red; }
```



Javascript

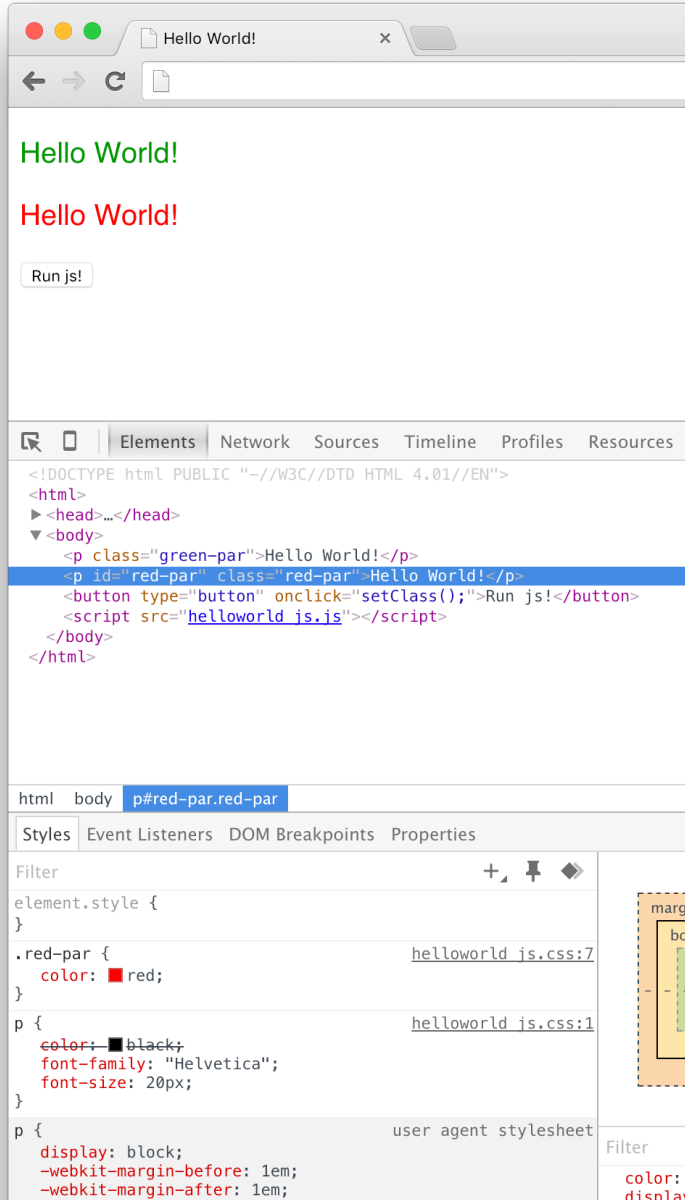
```
<body>
    <p class="green-par">Hello World!</p>
    <p id="red-par">Hello World!</p>
</body>

.green-par { color: green; }
.red-par { color: red; }

document.getElementById("red-par")
    .setAttribute("class", "red-par");
```



Hands On

- Text Editor + Browser (your choice)
  - Notepad++
  - Chrome
    (ctrl + shift + i // cmd + alt + i)

- Online IDEs (e.g. http://www.jsfiddle.net)



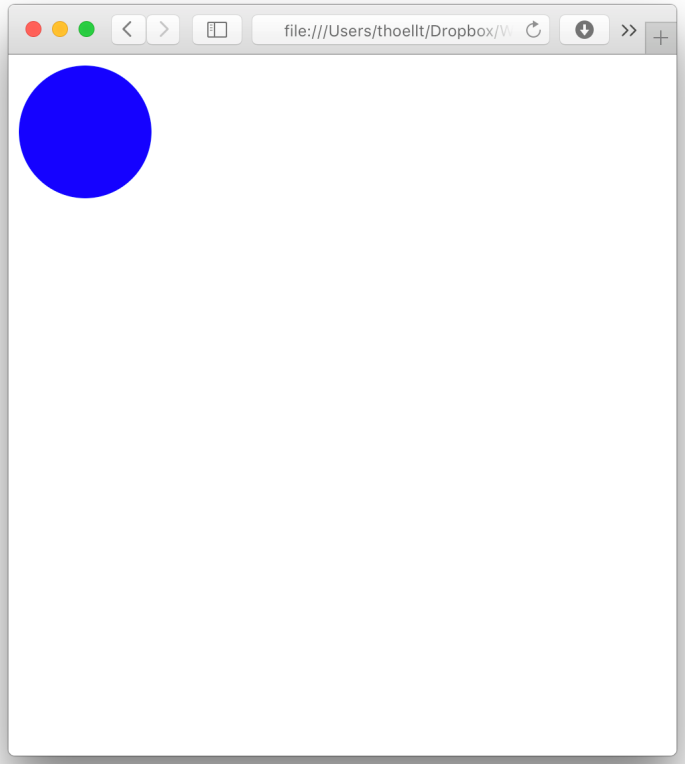
SVG

- Vector graphics standard based on XML
  http://www.w3schools.com/svg/

- Features
  - vector graphics
  - can be styled similar to html
  - can be edited interactively using js
- Disadvantages
  - creating xml structure by hand is tedious
  - relies on browser support

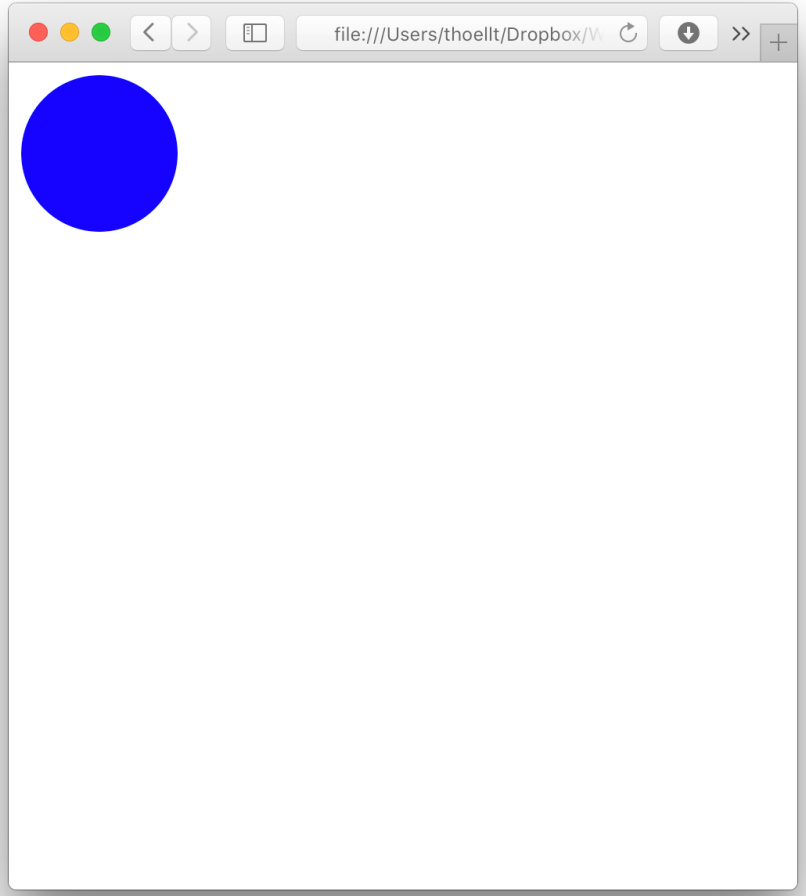
SVG

```
</body>
    <svg width="100" height="100">
        <circle cx="50" cy="50" r="50" style="fill: blue;"></circle>
    </svg>
</body>
```



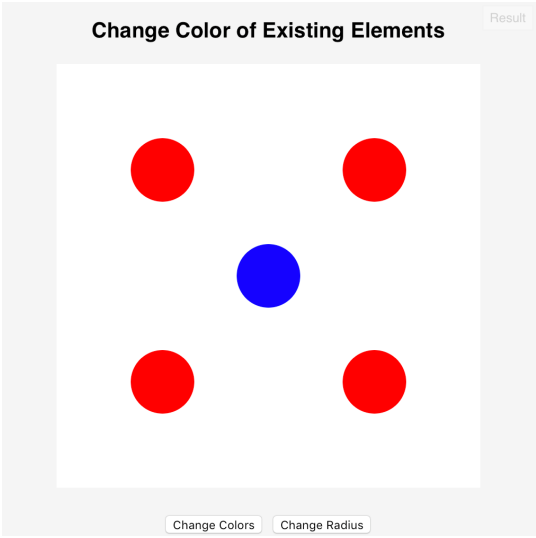
D3, Javascript & SVG

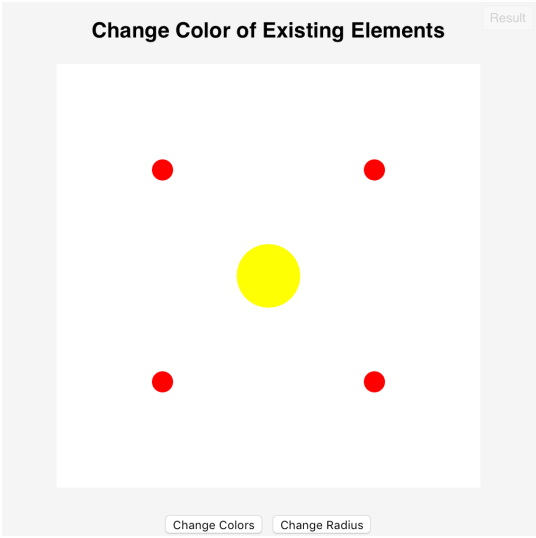
```
var size = 100;

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

svg.append("circle")
        .attr("cx", size/2)
        .attr("cy", size/2)
        .attr("r", size/2)
        .style("fill", "blue");
```



D3 - Select

```
d3.select("#blue")
      .style("fill", "yellow");

d3.selectAll(".red")
      .attr("r", 10);
```





http://jsfiddle.net/thoellt/6cvh0j3z/

Task 1

- What other shapes can you draw?
  - try drawing rectangles
- Look up the difference between style and attr
- What other styles/attributes can you change?
  - try adding a border

http://jsfiddle.net/thoellt/6cvh0j3z/

D3 - Enter & Exit

selection.enter()

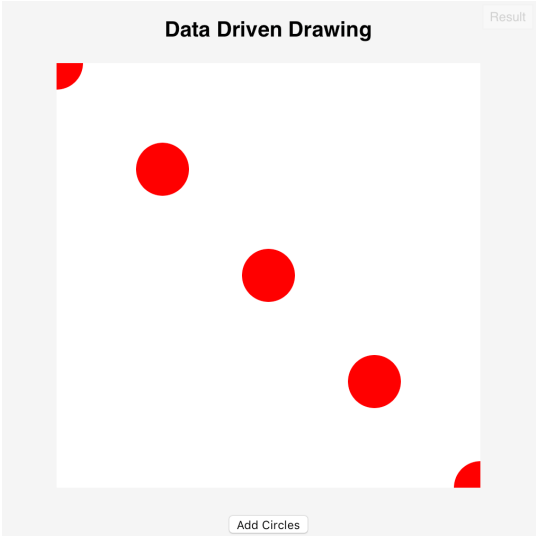
- Returns the entering selection:
  - placeholder nodes for each data element for which no corresponding existing DOM element was found in the current selection.
- Only defined on a selection returned by the data operator.
- Only defines append and insert operators

selection.exit()

- Returns the exiting selection:
  - existing DOM elements in the current selection for which no new data element was found.
- Only defined on a selection returned by the data operator.
- Returns a reference to the exiting selection.

D3 - Data Driven Drawing

```
var coords = [0, 100, 200, 300, 400];
var vis = d3.select("#vis");
var circles = vis.selectAll("circle")
                 .data(coords);

circles.enter()
       .append("circle")
       .attr("class","red")
       .attr("cx", function(d){return d;})
       .attr("cy", function(d){return d;})
       .attr("r", 25);
```



http://jsfiddle.net/thoellt/aqfg988o/

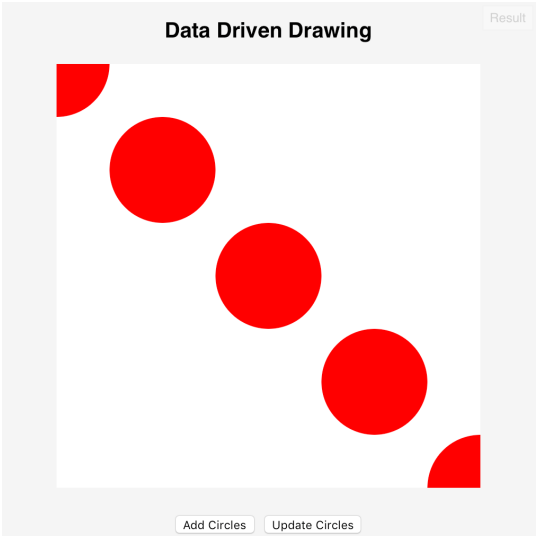
Task 2

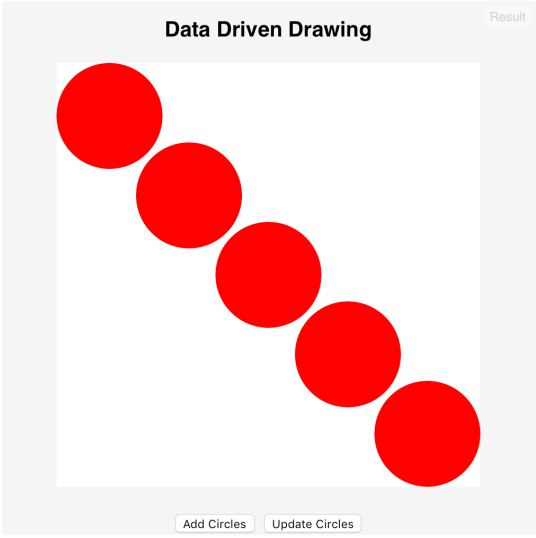
- Play around with the data functions
  - can you draw on a diagonal from top right to bottom left, without changing the coords array directly?
  - can you set the size using the coords array?
- The anonymous function() on an object with attached data can take more arguments: e.g. function(d, i).
  - Find out what these are used for and try to use them

http://jsfiddle.net/thoellt/aqfg988o/

D3 - Update

```
var coords = [50, 125, 200, 275, 350];
var vis = d3.select("#vis");

var circles = vis.selectAll("circle")
                 .data(coords);

circles.attr("cx", function(d){return d;})
       .attr("cy", function(d){return d;});
```

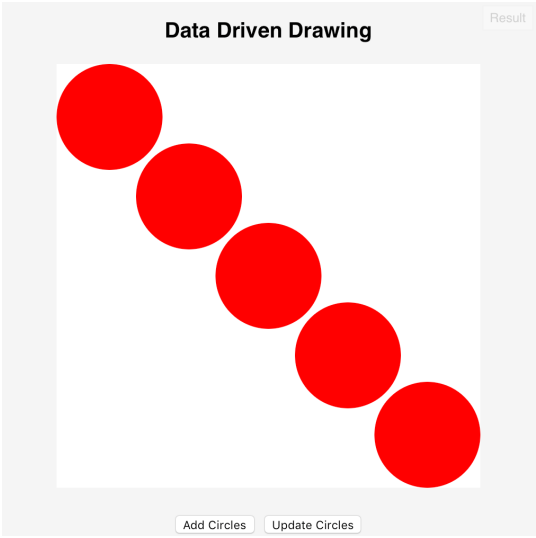


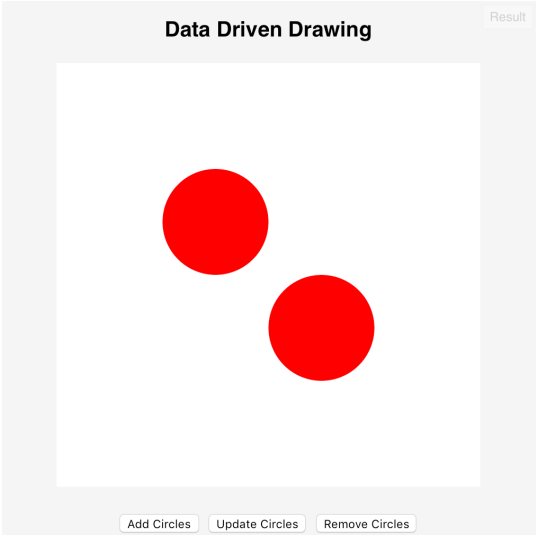


http://jsfiddle.net/thoellt/njLpvj90/

D3 - Exit

```
var circles = vis.selectAll("circle")
                 .data(coords);
circles.exit()
       .remove();

circles.attr("cx", function(d){return d;})
       .attr("cy", function(d){return d;});
```

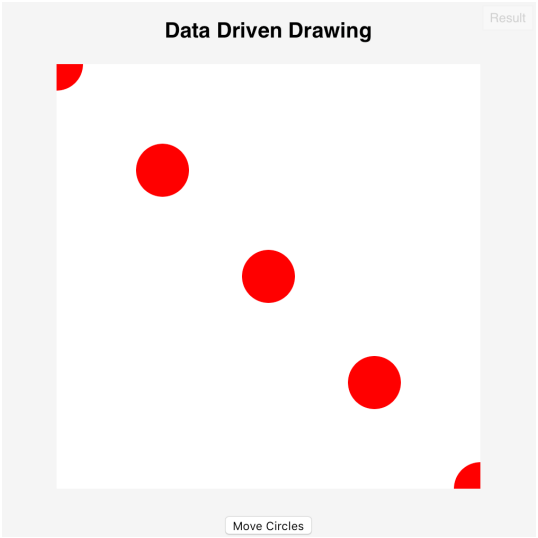


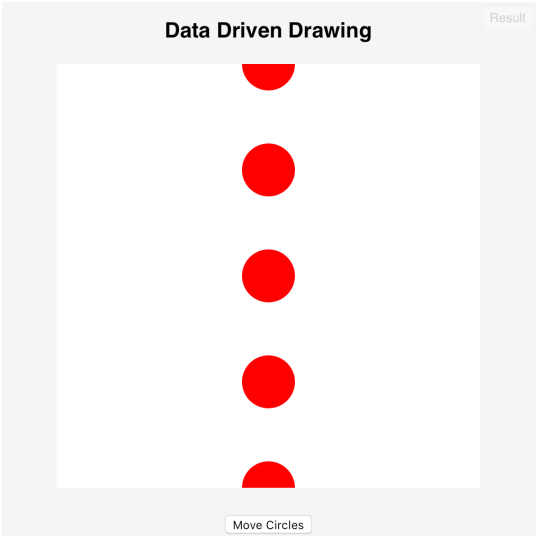


http://jsfiddle.net/thoellt/xv60v0rc/

D3 - Transition

```
circles.transition()
        .duration(1000)
        .attr("cx", 200);
```





http://jsfiddle.net/thoellt/kwqkb29r/

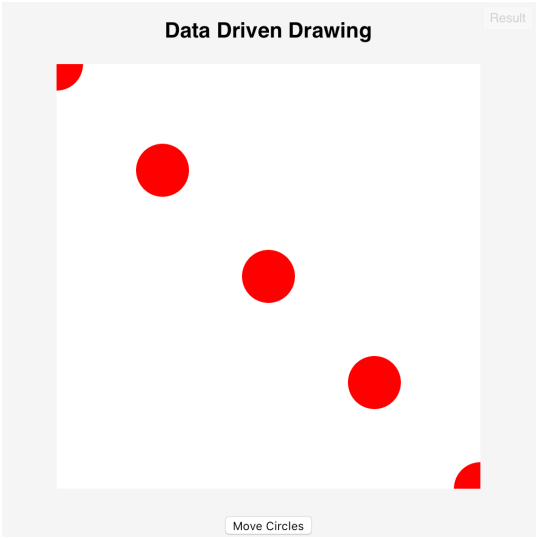
Task 3

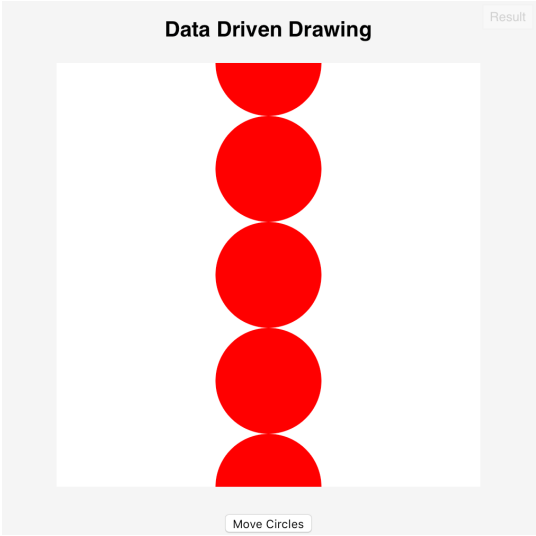
- Play around with the transitions
  - look up different transition effects
  - change the duration
  - add a delay
  - set an interpolation function

http://jsfiddle.net/thoellt/kwqkb29r/

D3 - Chained Transition

```
circles.transition()
        .duration(1000)
        .attr("cx", 200)
    .transition()
        .duration(1000)
        .attr("r", 50);
```

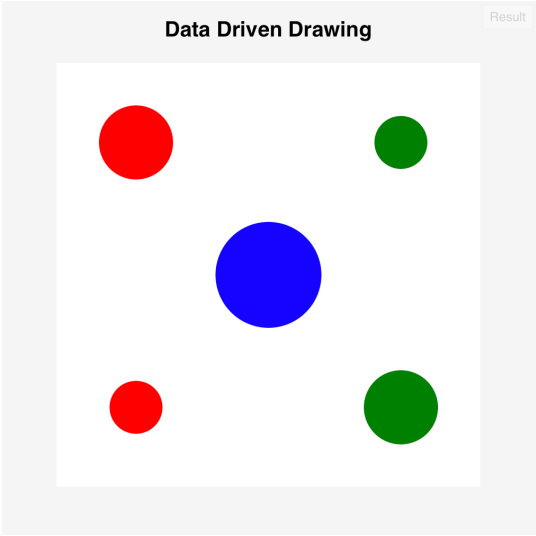




http://jsfiddle.net/thoellt/4keqg1ft/

D3 - Advanced Data

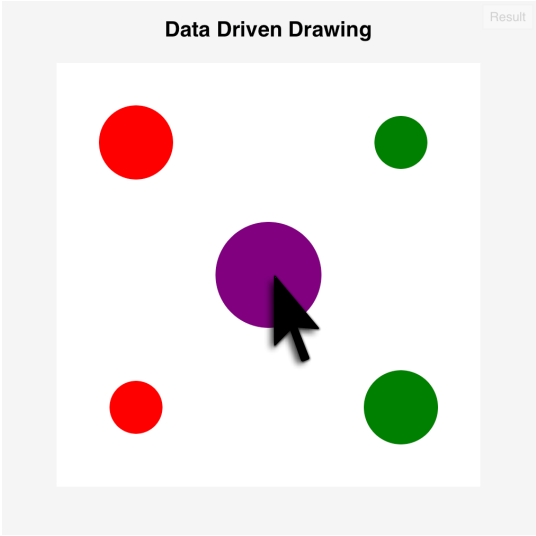
```
var data =
[ {x: 75, y:75, r:35, class: "red"},
  {x: 325, y:75, r:25, class: "green"},
  {x: 200, y:200, r:50, class: "blue"},
  {x: 75, y:325, r:25, class: "red"},
  {x: 325, y:325, r:35, class: "green"}];
```



http://jsfiddle.net/thoellt/xpq64ghs/

D3 - Interaction

```
circles
    .on("mouseover", function() {
        d3.select(this).attr("class", "purple")
    })
    .on("mouseout", function() {
        d3.select(this).attr("class", function(d){return d.class})
    });
```



http://jsfiddle.net/thoellt/ya21nxnc/

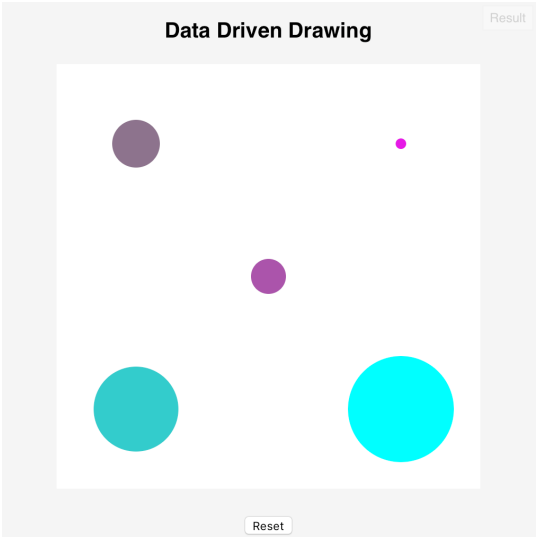
Task 4

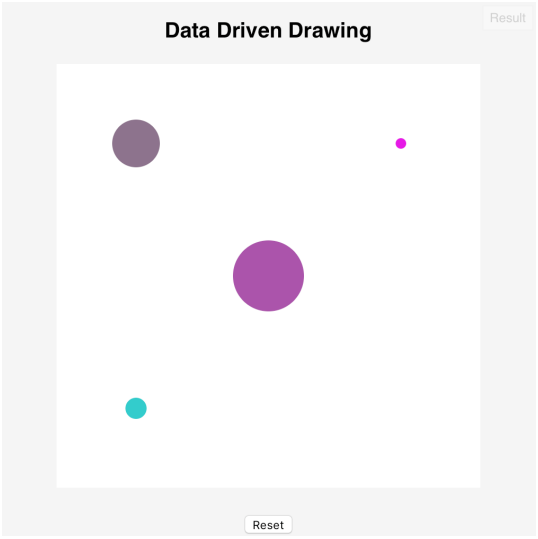
- Play around with interactions
  - look up different interaction methods
  - mouse press?
  - keyboard?

http://jsfiddle.net/thoellt/ya21nxnc/

D3 - Scales

```
var radiusScale = d3.scale.linear()
        .domain([0.0, 1.0])
        .range([0, 50]);

var colorScale = d3.scale.linear()
        .domain([0.0, 0.5, 1.0])
        .range(["magenta", "grey", "cyan"]);
```





http://jsfiddle.net/thoellt/qvfow985/

Task 5

- Play around with scales
  - can you invert the colour scale?
  - can you add a colour to the colour scale?

- Look up other scales than linear

http://jsfiddle.net/thoellt/qvfow985/

JSON Data

object: {members}

members: pair | pair, members

pair: string : value

array: [elements]

elements: value | value , elements

value:
- string
- number
- object
- array
- true
- false
- null

```
{ vectors: [ { x: 0.0, y: 0.0, u: 1.0, v: 1.0 },
             { x: 0.0, y: 0.0, u: 1.0, v: 1.0 },
             { x: 0.0, y: 0.0, u: 1.0, v: 1.0 } ] }
```

JSON Data

```
var data;

d3.json("path/to/file.json", function(error, json) {
  if (error) return console.warn(error);
  data = json;
});
```

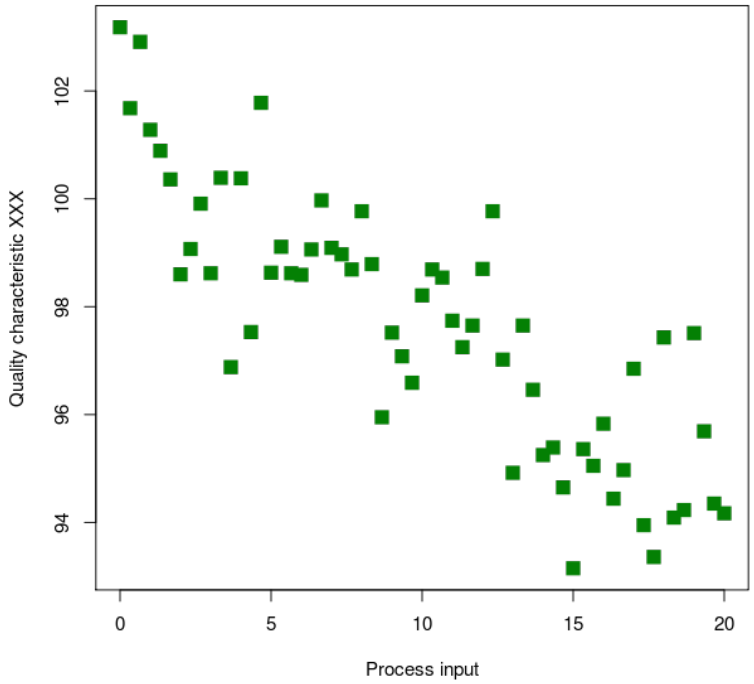
By default not allowed by most browsers

- We use data in a js file already declared as a variable

Exercise 1

- A scatterplot should have
  - data (see next slide)
  - axes/labels (description)

- Too easy?
  - allow selection of the variable per axis (x, y, u, v, m, a)
  - color code a variable
  - adapt chart to the window size



Exercise 1

- Build a Scatterplot

```
var boat_data = {  boats:[
    {
      x: 170.83,  // Lat
      y:  282.34, // Lon
      u: -14.56,  // Lat velocity
      v:  9.90,   // Lon velocity
      m: 6.39,    // Mag error
      a:  41.13   // Angular error
    }, {
      x: 170.83,
      y:  282.34,
      ... } ] };
```

```
var boats = boat_data.boats;

var boat_0 = boats[0];
var x_coord = boat_0.x;

on DOM element with data
function(d){ return d.x };
```