D3 Foundations

Francesca Morini // Mark-Jan Bludau
21.05.2019

Hi!

Hi everyone, this is Francesca and Mark-Jan

Today we will introduce you to scrollytelling and D3!

D3?!

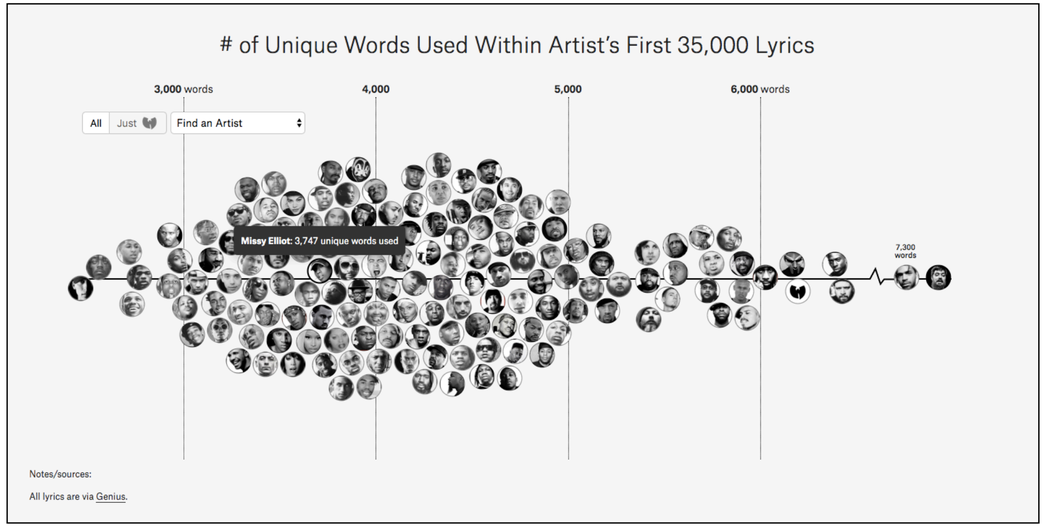
It’s a JS library for dataviz!

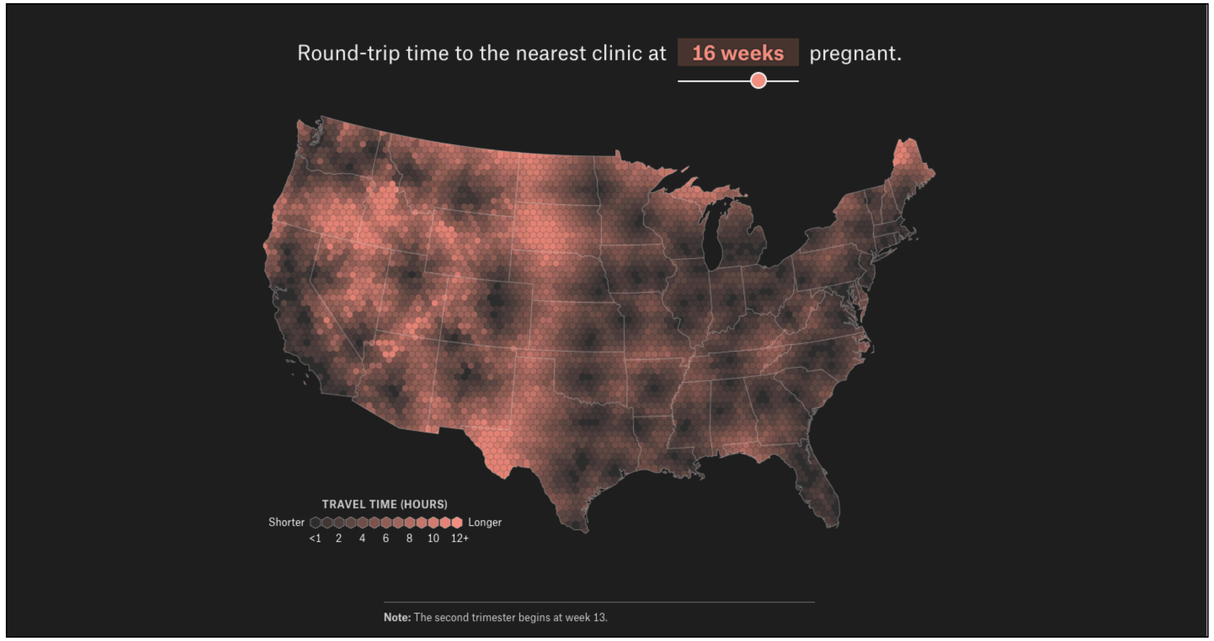
or in other words:

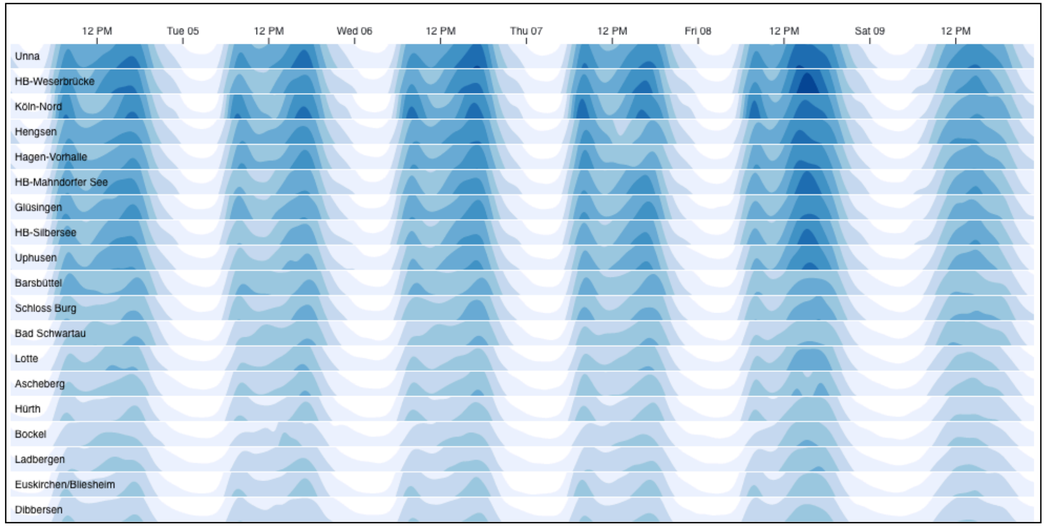
D3.js is a JavaScript library for manipulating documents based on data. D3 helps you bring data to life using HTML, SVG, and CSS.

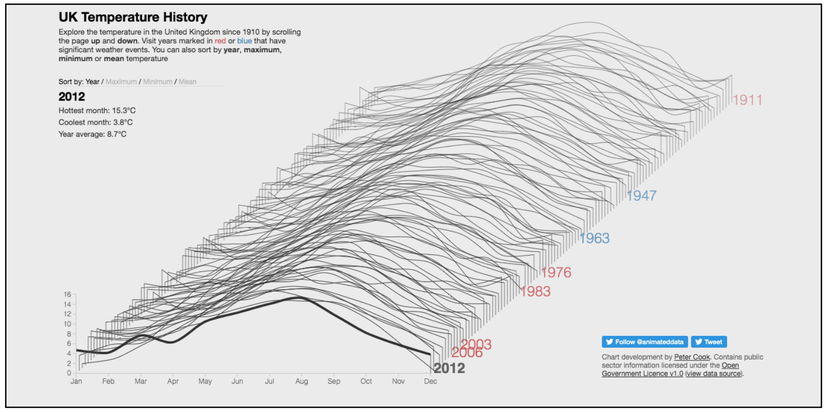
Mike Bostock

Some Examples:









more examples!

How does it work?

“D3 allows you to bind arbitrary data to a Document Object Model (DOM), and then apply data-driven transformations to the document.”

DATA

are attached to the

DOM

and they can transform elements!

HTML nodes

SVG

iso	sector	X1990	X1991	X1992
abw	1A3dii_Domestic-navigation	0.3830770125	0	0
abw	1A3eii_Other-transp	0.5391749893	0.5634653009	0.5138732368
abw	1A4a_Commercial-institutional	0	0	0
abw	1A4b_Residential	73.76106681	75.05089431	69.93287991
abw	1A4c_Agriculture-forestry-fishing	20.88468468	21.01007967	18.78403777
abw	1A5_Other-unspecified	524.2277051	515.0200206	413.0592833

But before diving in

A quick JS revision!

Variables

```
var a = “Hallo Welt“          // String
var b = 5                     // Number
var c = 3                     // Number
var d = b+c =8                //Number

var d = true                  // Boolean
var e = false                 // Boolean
```

Arithmetic Operators

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus
++	Increment
--	Decrement

Arrays

```
var array = [2, 3, 8, 1, 9]
Index beginnt bei 0!
console.log(array[0]) // = 2
console.log(array[4]) // = 9
console.log(array.length) // = 5
```

Objects

```
var object = {name: “Max”, nachname:”Mustermann”, alter: 12}
console.log(object) // =  {name: “Max”, nachname:”Mustermann”, alter: 12}
console.log(object.name) // = Max
console.log(object[“nachname”]) // = Max
```

Functions

Example

Calculate the product of two numbers, and return the result:

```
var x = myFunction(4, 3);        // Function is called, return value will end up in x

function myFunction(a, b) {
    return a * b;                // Function returns the product of a and b
}
```

The result in x will be:

```
12
```

Logical Operators

Operator	Description	Example
&&	and	(x < 10 && y > 1) is true
||	or	(x == 5 || y == 5) is false
!	not	!(x == y) is true

Comparison Operators

Operator	Description	Comparing	Returns
==	equal to	x == 8	false
		x == 5	true
		x == "5"	true
===	equal value and equal type	x === 5	true
		x === "5"	false
!=	not equal	x != 8	true
!==	not equal value or not equal type	x !== 5	false
		x !== "5"	true
		x !== 8	true
>	greater than	x > 8	false
<	less than	x < 8	true
>=	greater than or equal to	x >= 8	false
<=	less than or equal to	x <= 8	true

Three fundamental concepts:

1

Selections

2

Joins

3

Scales

Selections?

Let’s modify this element color from red to blue!

Hello world!

<p>Hello world!< /p>

The classic way:

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

The D3 way:

```
d3.selectAll("p").style("color", "blue");
```

Those two pieces of code do the exact same thing, but look at how verbose is the first one!

Selections?

D3 employs a declarative approach, operating on arbitrary sets of nodes called selections.

This means you can select one or multiple elements by specifying their selector name (eg. ‘body’), class or id on the DOM and change their properties, for example color, size or position.

.selectAll( )

Select multiple elements

.select( )

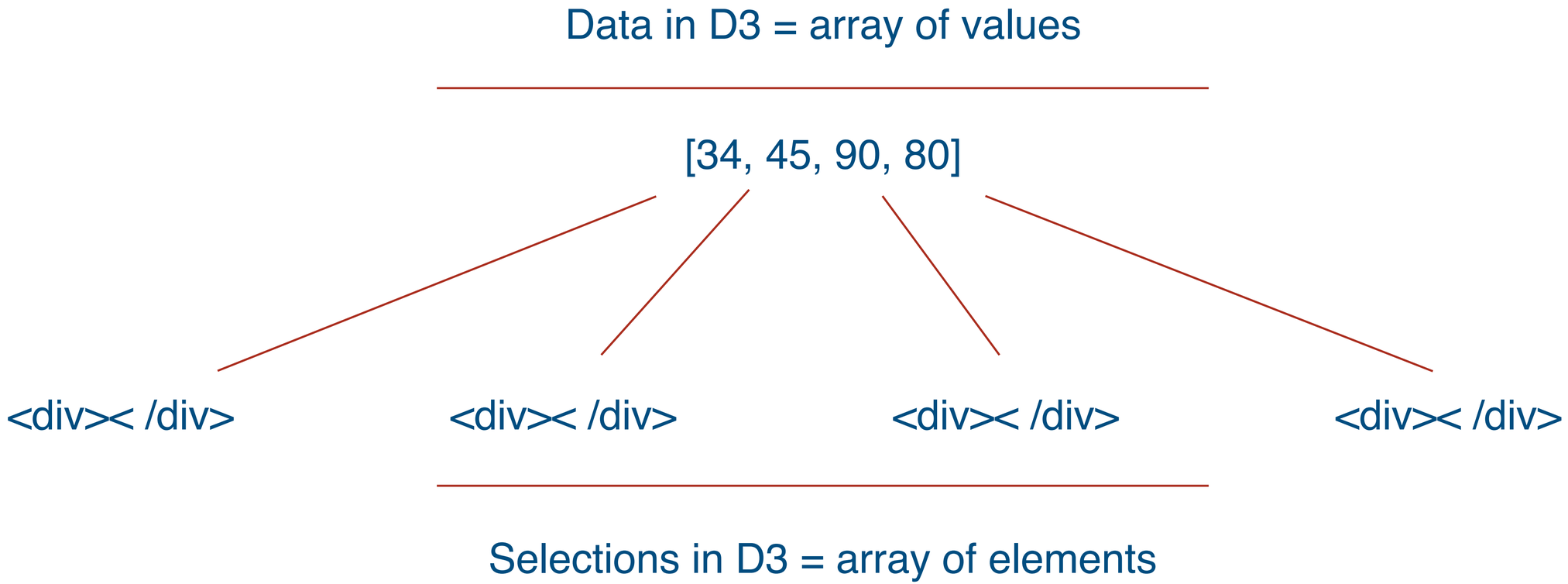
Select one single element

Selections?

Mutations on selections are often based on data

Data is specified as an array of values, and each value is passed as the first argument (d) to selection functions. With the default join-by-index, the first element in the data array is passed to the first node in the selection, the second element to the second node, and so on.

Joins?


Data in D3 = array of values
[34, 45, 90, 80]
< /div>
< /div>
< /div>
< /div>
Selections in D3 = array of elements

Joins?

Data are passed to selection through the use of the operator

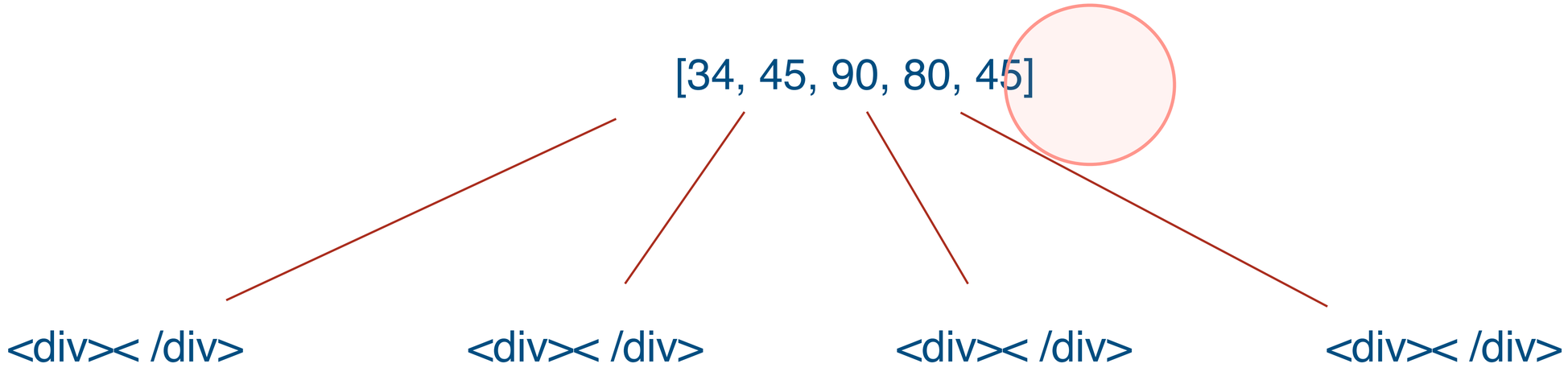
.data( )

Joins?

Passing data to exiting elements it’s not the only possibility, what if I want to create or delete new elements based on my data?

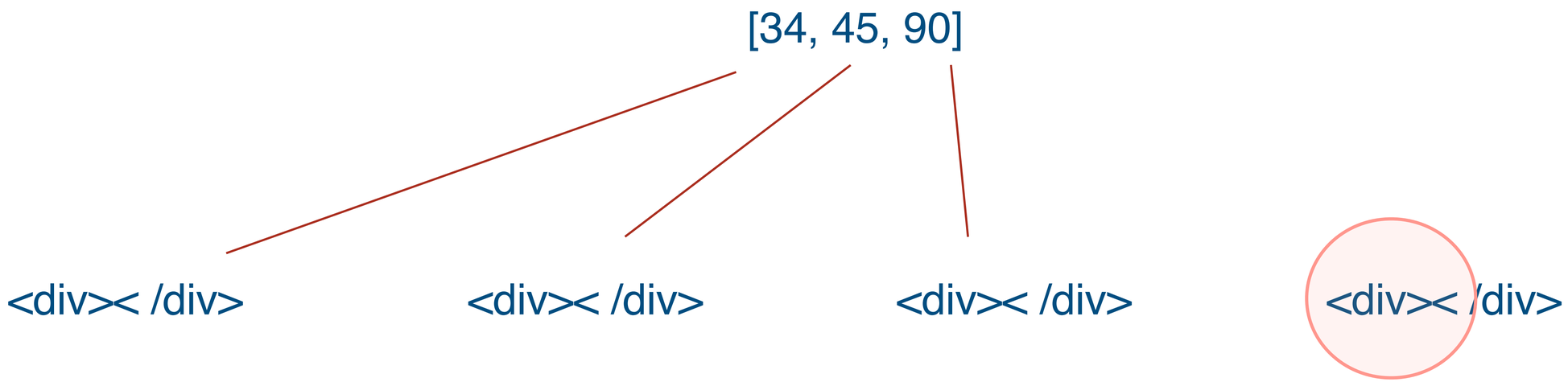
.enter( ) and .exit( ) will do the work

Joins?



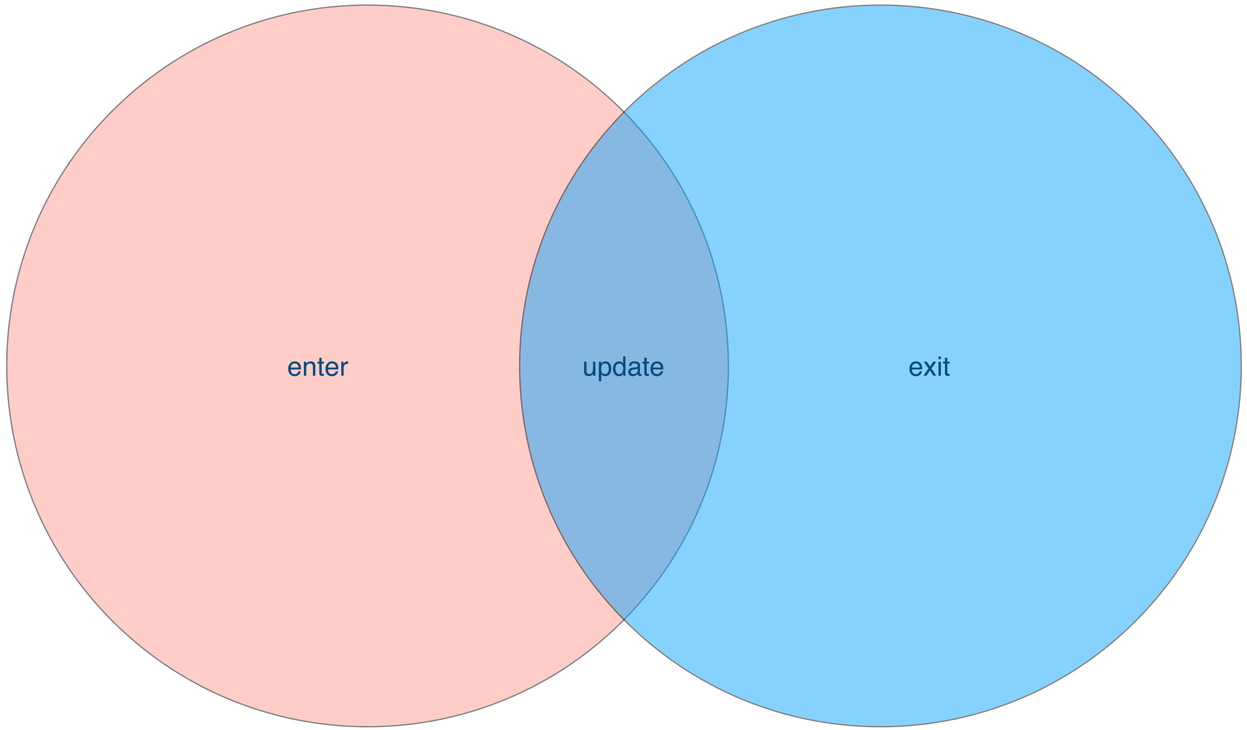
.enter ( ) will create more elements in the DOM if the number of values is bigger than the elements already present

Joins?



.exit ( ) will delete elements in the DOM if the number of values is smaller than the elements already present

Joins?



Scales?

When creating elements with data also the position on the page of those elements have to be considered and properly mapped.

D3 scales will help you mapping and representing your data as visual variables, such as position, color, length and so on.

Scales?

Scales have two important operator that you need to know about.

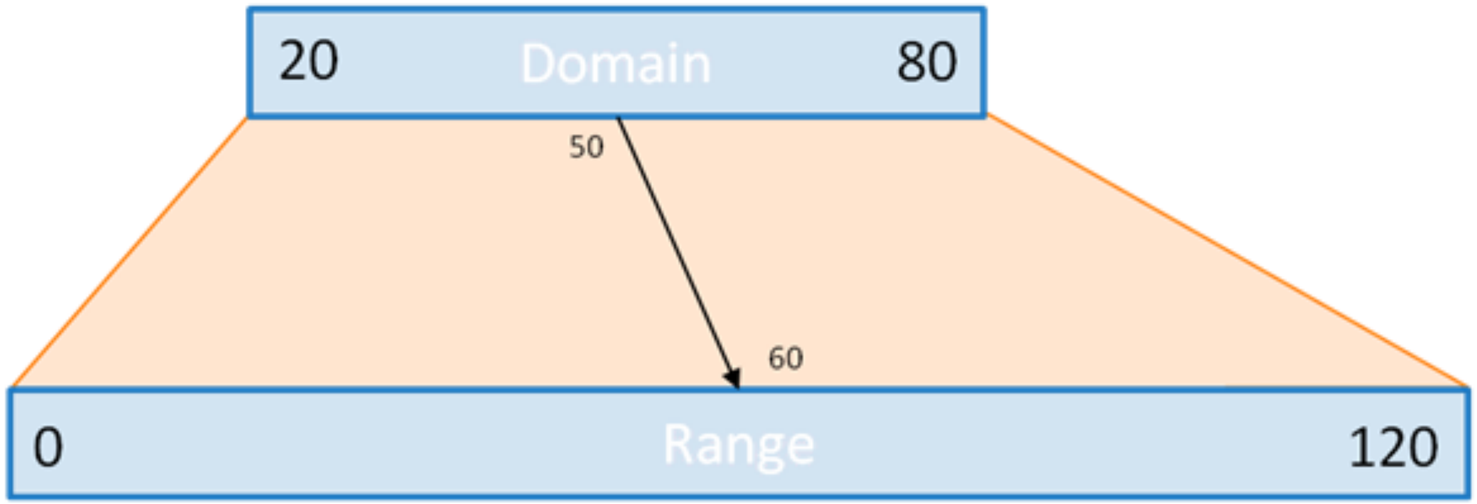
.domain( )

Domain represents the boundaries within which your data lies. e.g. If I had an array of numbers with no number smaller than 1 and no number larger than 100, my domain would be 1 to 100

.range( )

if you are plotting a graph and the one value is in tens of thousands, it is unlikely that you will be able to have a bar graph with the same pixel height as the data. In that case, you need to specify the boundaries within which your original data can be transformed. These boundaries are called the range.

Scales?



Let’s try it out!

Bibliography and further material:

Beherens C., “Enter, Update, Exit"

Bostock M., “Selections"

Bostock M., “Thinking with Joins"

Sukale R., “Understanding Domain, Ranges and Scales"

More code examples: Search the Blocks

More tutorials (some are old): D3 Tutorials index