ToiletPaper #6

jambit
WHERE INNOVATION WORKS

Datenvisualisierung im Web mit der JavaScript-Bibliothek D3

Von Philipp Hemmer

✖ Problem

Wir möchten Daten ansprechend im Browser darstellen.
Das Diagramm soll updatefähig, interaktiv und animierbar sein.

✔ Lösung

D3.js ist eine JavaScript Bibliothek, um dynamische, interaktive Datenvisualisierungen für den Browser zu erstellen.
D3.js ist nicht direkt eine Diagramm-Bibliothek, sondern vielmehr ein Toolkit, um vielseitige Datenvisualisierungen zu erstellen.
Wie funktioniert D3.js? Browser können seit einiger Zeit SVG-Dateien anzeigen, deren Datenmodel auf XML beruht. Dies macht sich D3.js zunutze und erzeugt oder verändert den DOM der SVG-Datei. Canvas und HTML werden ebenfalls unterstützt.

➔ Beispiel

SVG

Hier ein Beispiel-SVG für ein kleines Balkendiagramm:

```
<svg width="100" height="100"
style="background:orange;">
  <rect y="78" x="10" height="22"
width="20"></rect>
  <rect y="41" x="40" height="59"
width="20"></rect>
  <rect y="16" x="70" height="84"
width="20"></rect>
</svg>
```

Der SVG- und D3.js-Code erzeugen beide folgendes Diagramm:

Demo für dieses Beispiel:
https://codepen.io/anon/pen/MoNjmW

D3.js

JavaScript-Code, um das Diagramm zu erzeugen:

```
let height = 100, barWidth=20, padding=10;
let y = d3.scaleLinear() //      1) Wertprojektion
  .domain([1, 50])
  .rangeRound([height, 0]);
d3.select('svg')
  .attr('width', 100)
  .attr('height', 100)
  .attr('style', 'background:orange;')
  .selectAll('rect') //            2) Selection
  .data([12, 30, 42]) //           3) Data binding
  .enter()
  .append('rect') //               4) SVG DOM
  .attr('y', (d) => {return y(d);})
  .attr('x', (d,i) => {
    return i*(barWidth+padding)+padding;
  })
  .attr('height', (d) => {return height-y(d);})
  .attr('width',barWidth);
```

1) Es wird eine Scale-Funktion aus D3 genutzt um den Wertebereich der Daten (1-50) auf die SVG-Koordinaten abzubilden.
2) Auf dem SVG-Node werden alle *rect*-Nodes „ausgewählt“; diese müssen zum Zeitpunkt von *selectAll(…)* noch gar nicht existieren.
3) Hier werden die Daten an den SVG-Node gebunden. Das kann man auch in der Developer-Console am Node sehen.
4) Durch *enter(…)* definiert man, was für jeden Wert ausgeführt werden soll. In unserem Fall soll ein Rechteck hinzugefügt werden. Der zweite Parameter ist eine Funktion. Diese hat als ersten Parameter den Wert und als zweiten den Index. Damit berechnen wir Position und Maße des Rechtecks.

✚ Weiterführende Aspekte

D3.js Ressourcen:
- https://d3js.org/
- http://animateddata.co.uk/articles/d3/whatisd3/

Alternative Diagramm-Bibliotheken:
- http://nvd3.org/ (auf Basis von D3.js)
- https://www.highcharts.com/ (kommerziell)

Vergleich von JavaScript Diagramm-Bibliotheken: https://en.wikipedia.org/wiki/Comparison_of_JavaScript_charting_frameworks

Kostenloser Udacity Kurs: https://de.udacity.com/course/data-visualization-and-d3js–ud507/