

**MOME Open
Adatvizualizáció**

Bevezetés a D3 használatába

Dévai Nándor
mail@nandordevai.net

Bevezetés

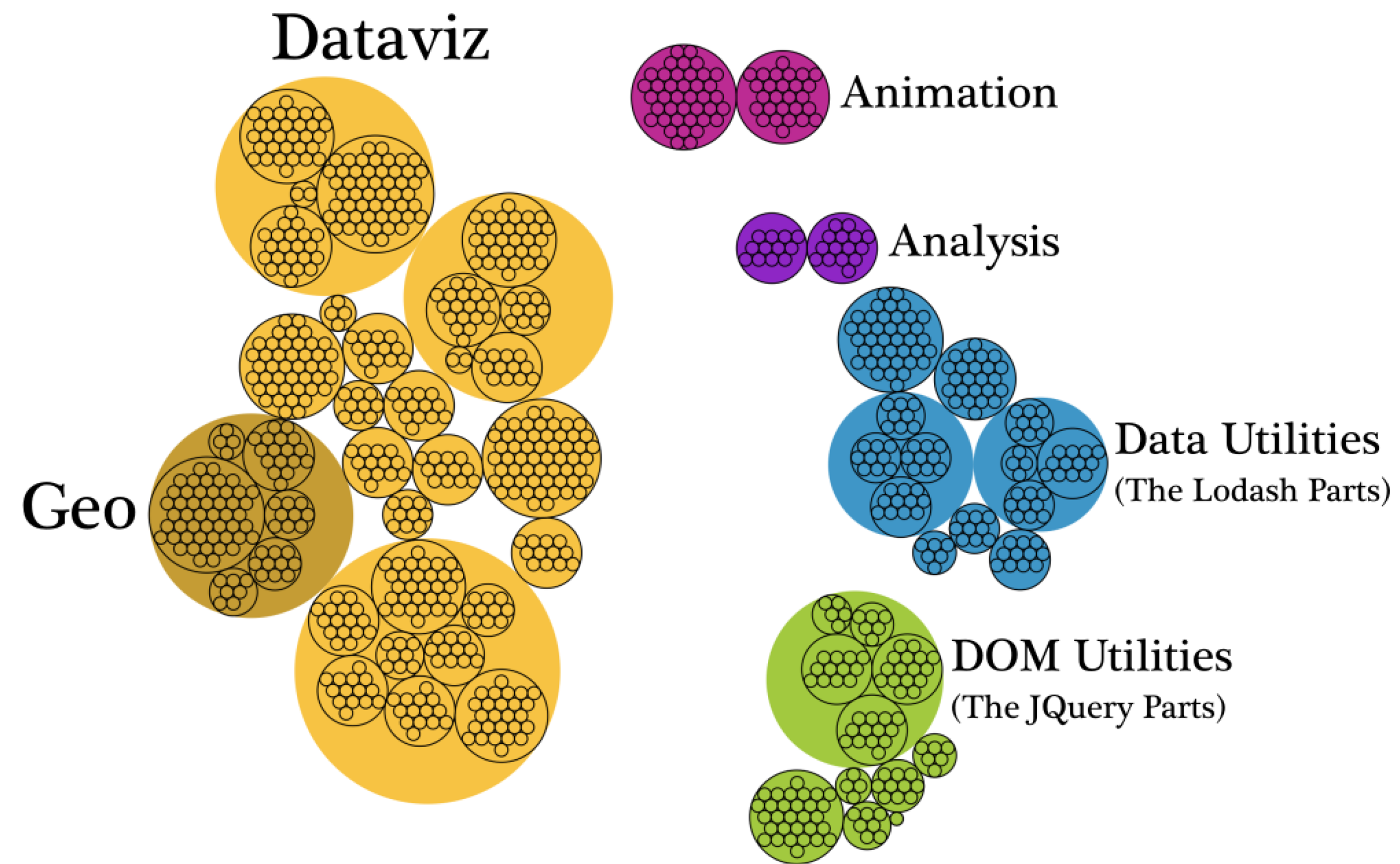
D3: Data Driven Documents

<https://d3js.org>

JavaScript programkönyvtár adatvezérelt weboldalak készítéséhez

Összekapcsolja az oldal elemeit az adatpontokkal

A D3 API



The D3 API

Előnyök

- rugalmas
- interaktív
- sokoldalú
- közvetlenül publikálható weben
- jól dokumentált
- sok példa, útmutató

Hátrányok

- viszonylag sok tanulást igényel
- mindent alapvető elemekből kell összerakni
- a dokumentáció API jellegű
- nem minden rendszerbe integrálható könnyen

Adatok

Adatforrások

- API (valós idejű grafikonok), pl. Airtable (<https://airtable.com>), Star Wars API (<https://swapi.co>)
- DSV (comma/tab separated values)
- Fetch API (képfájlok, JSON, szöveg, stb.)

Promise

- jövőben befejeződő műveletet (és annak eredményét) reprezentáló objektum
- adatok betöltésénél használjuk
- összeköti az adatot létrehozó és az azt felhasználó kódot

A vizualizáció elemei

HTML

DOM (Document Object Model) fa:

HTML dokumentumot reprezentáló fa, amelyben minden csomópont a dokumentum egy-egy részét képező objektum

Általános (pl. **div**) elemek használhatók vizualizációhoz

SVG

Scalable Vector Graphics (<https://developer.mozilla.org/en-US/docs/Web/SVG>)

Fő elemtípusok: `rect`, `circle`, `text`, `line`, `g`

Koordinátarendszer

- bal felső sarok: 0, 0
- x és y pozitív

Stílusok: `fill`, `stroke`, `stroke-width`

Transzformáció: `translate`, `rotate`, `scale`

CSS osztályok és stílusok alkalmazhatók

Selection

A selection függvények határozzák meg, hogy mely elemekkel kötjük össze az adatot

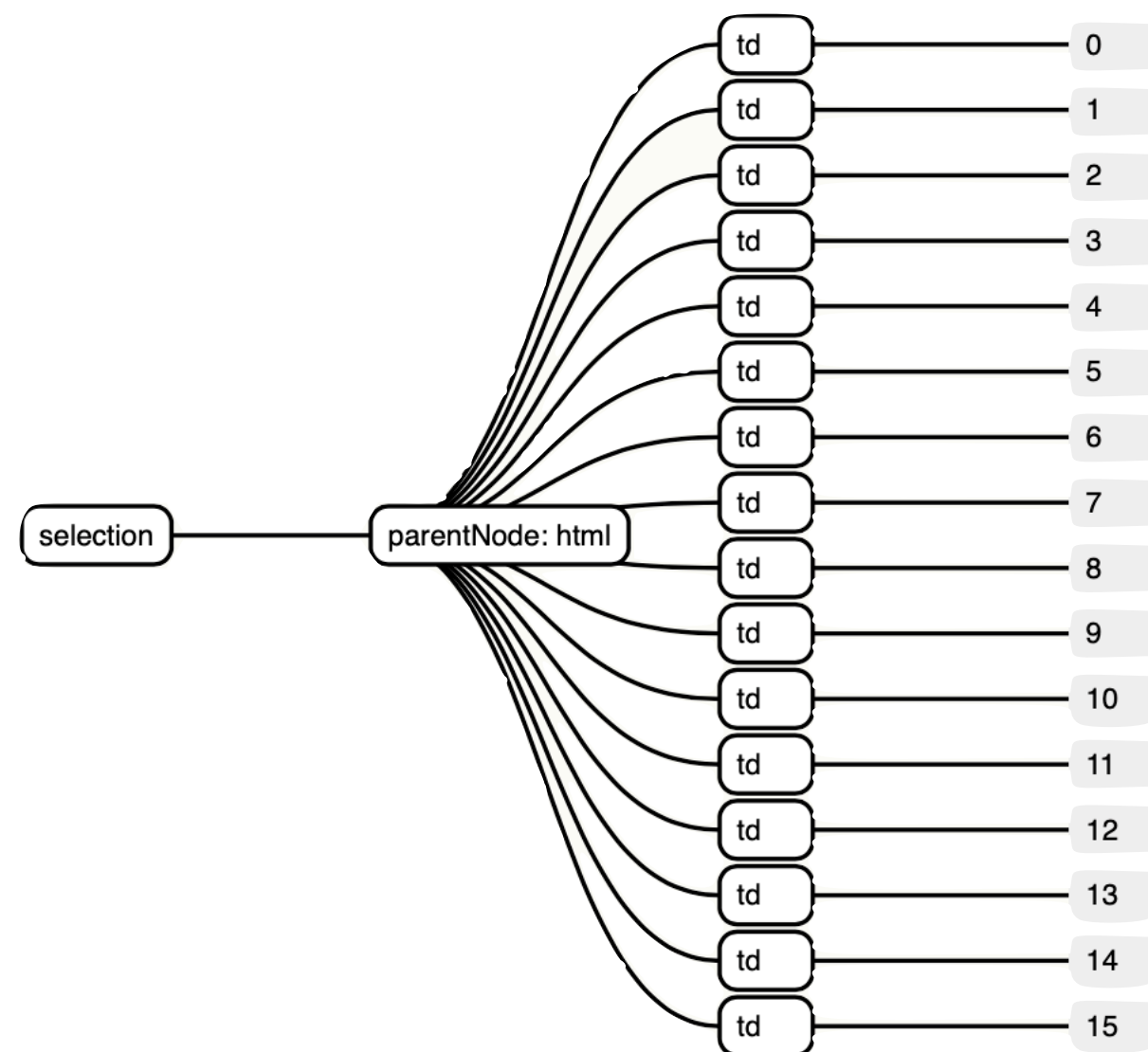
CSS selector: elem, v. elemek csoportjának kiválasztására szolgáló, egyszerű szöveges formában leírt szabály

`select()`, `selectAll()` függvényeket használjuk egy ill. több elem kiválasztására

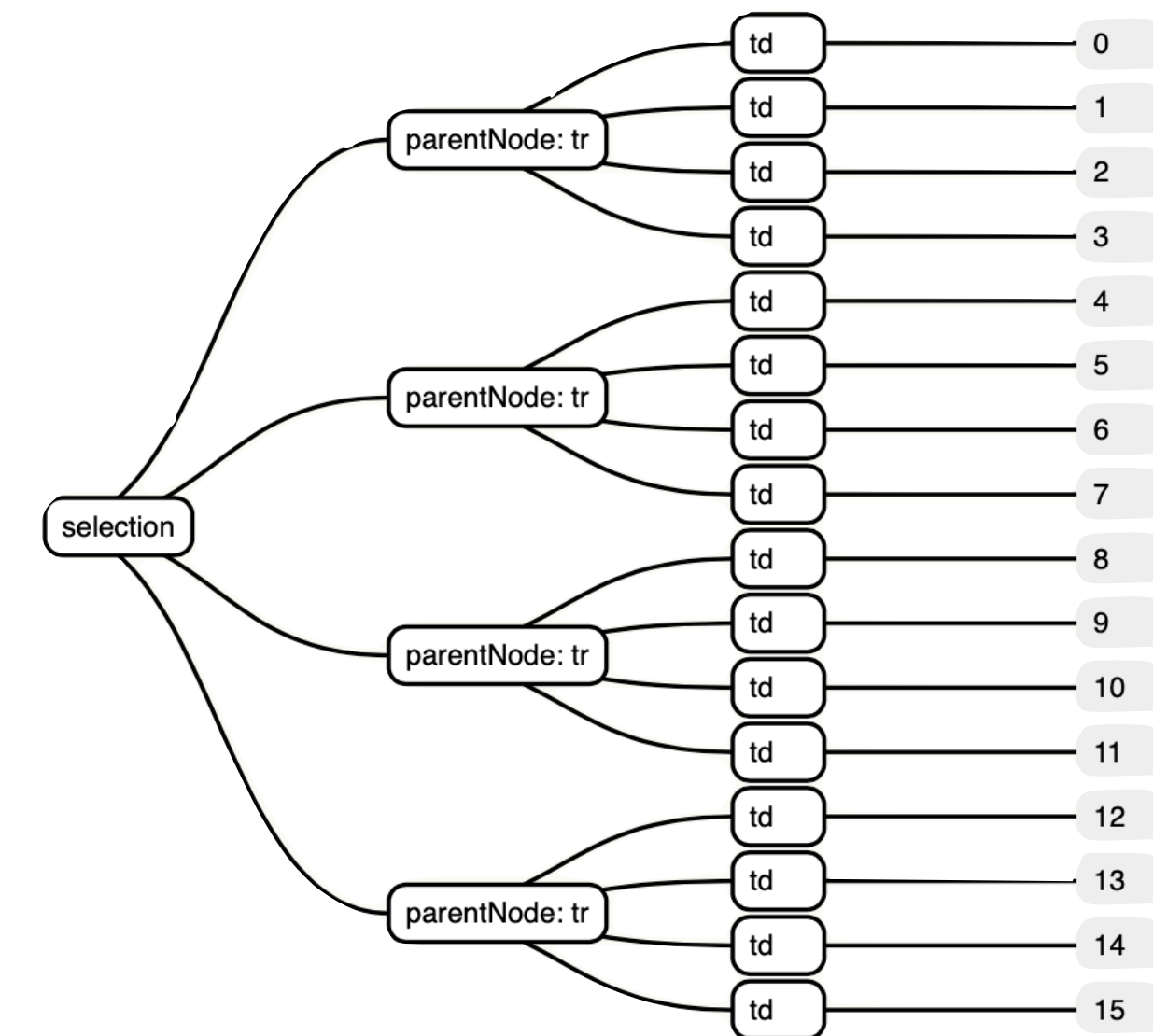
Flat vs. nested selection

Nested selection: megőrzi az adatstruktúrát,
pl. táblázatos adatoknál hasznos

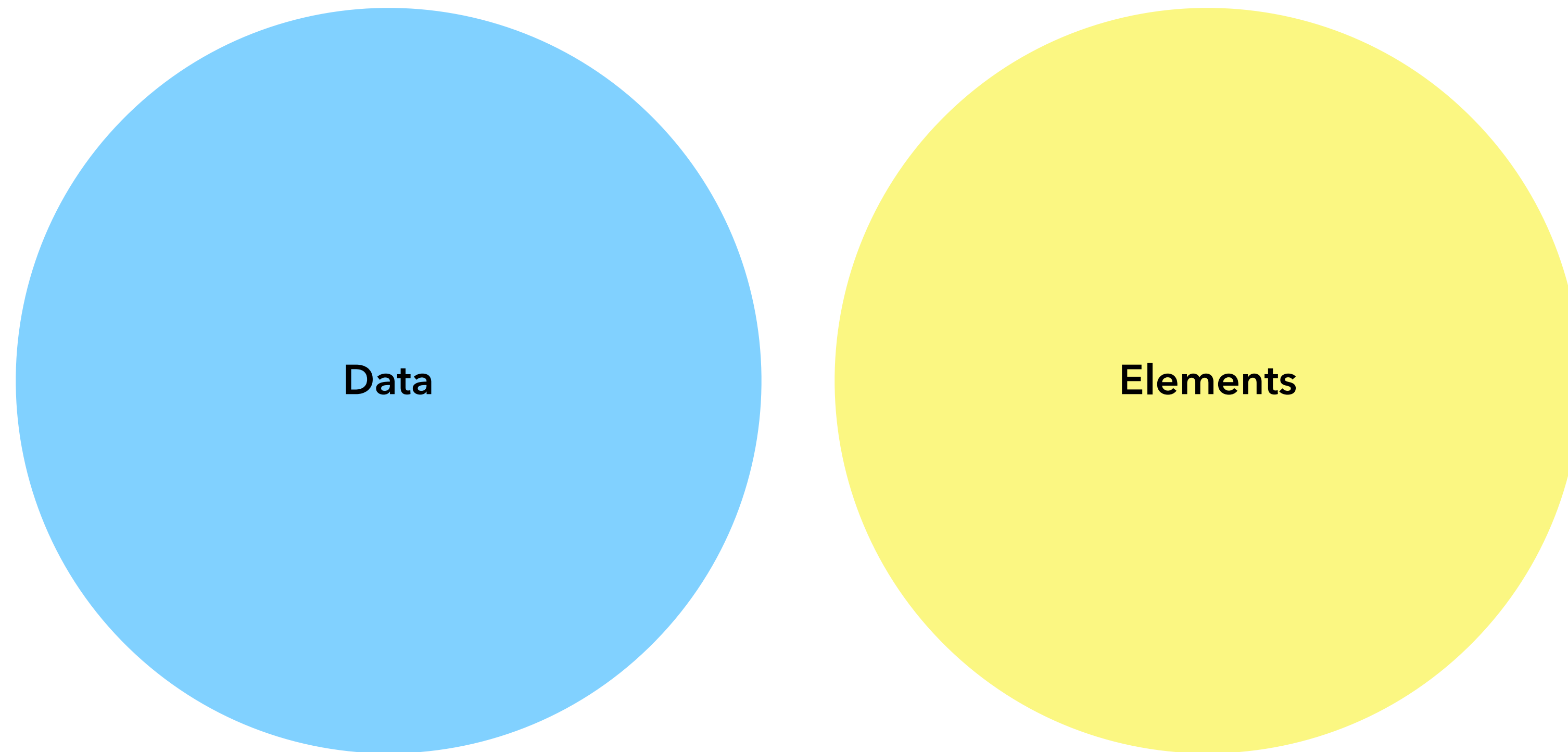
```
var td = d3.selectAll("tbody td");
```



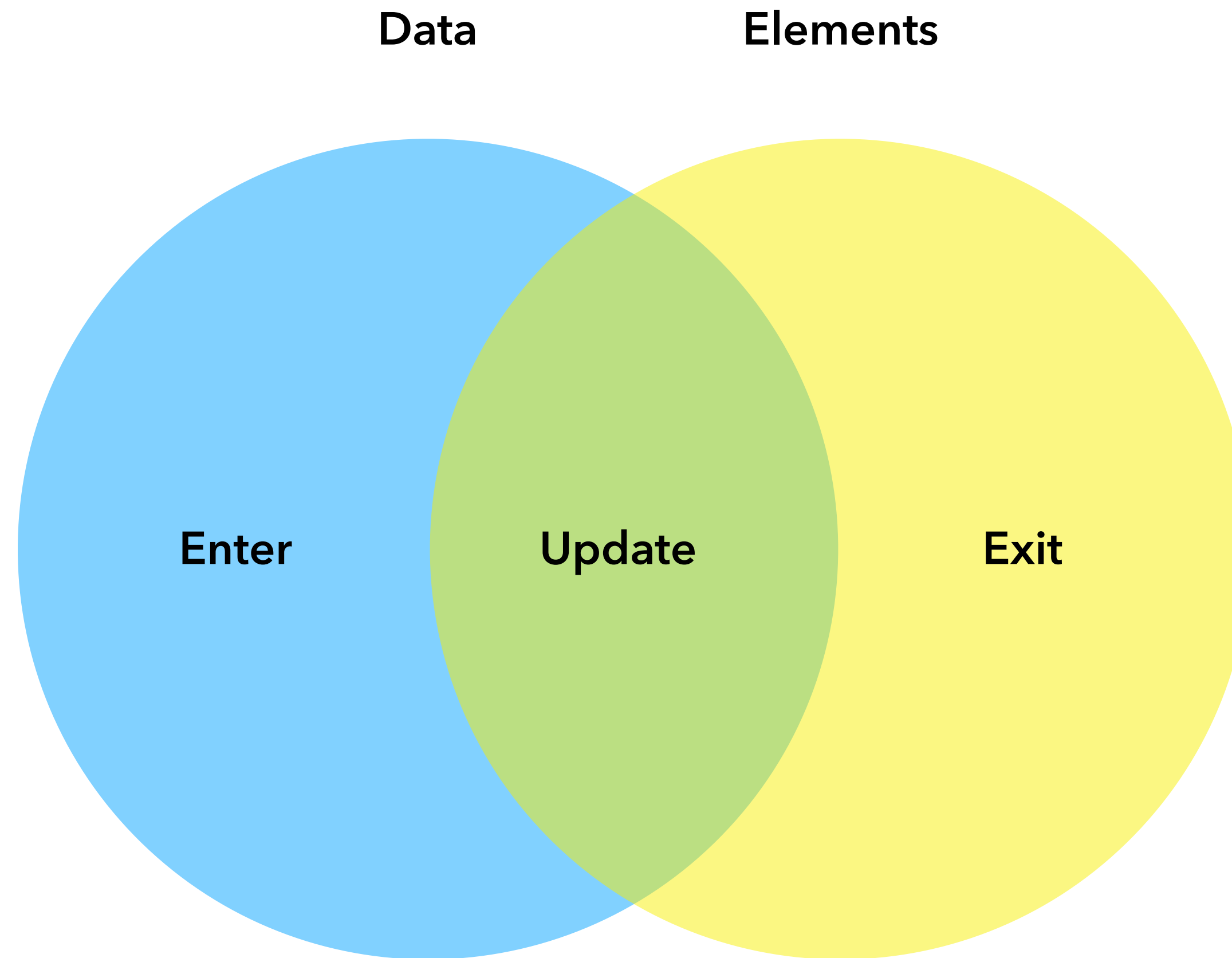
```
var td = d3.selectAll("tbody tr")  
    .selectAll("td");
```



Data join



Data join



Data join használata

- A `selection.data()` függvény visszaadja az update selectiont
- A `selection.enter()`, `selection.exit()` függvények létrehozzák a megfelelő selectiont
- Az adott függvényekben a kívánt módon kezeljük az egyes selectionöket (tulajdonságok módosítása, törlés/hozzáadás, áttűnések)

Number of things in Star Wars movies

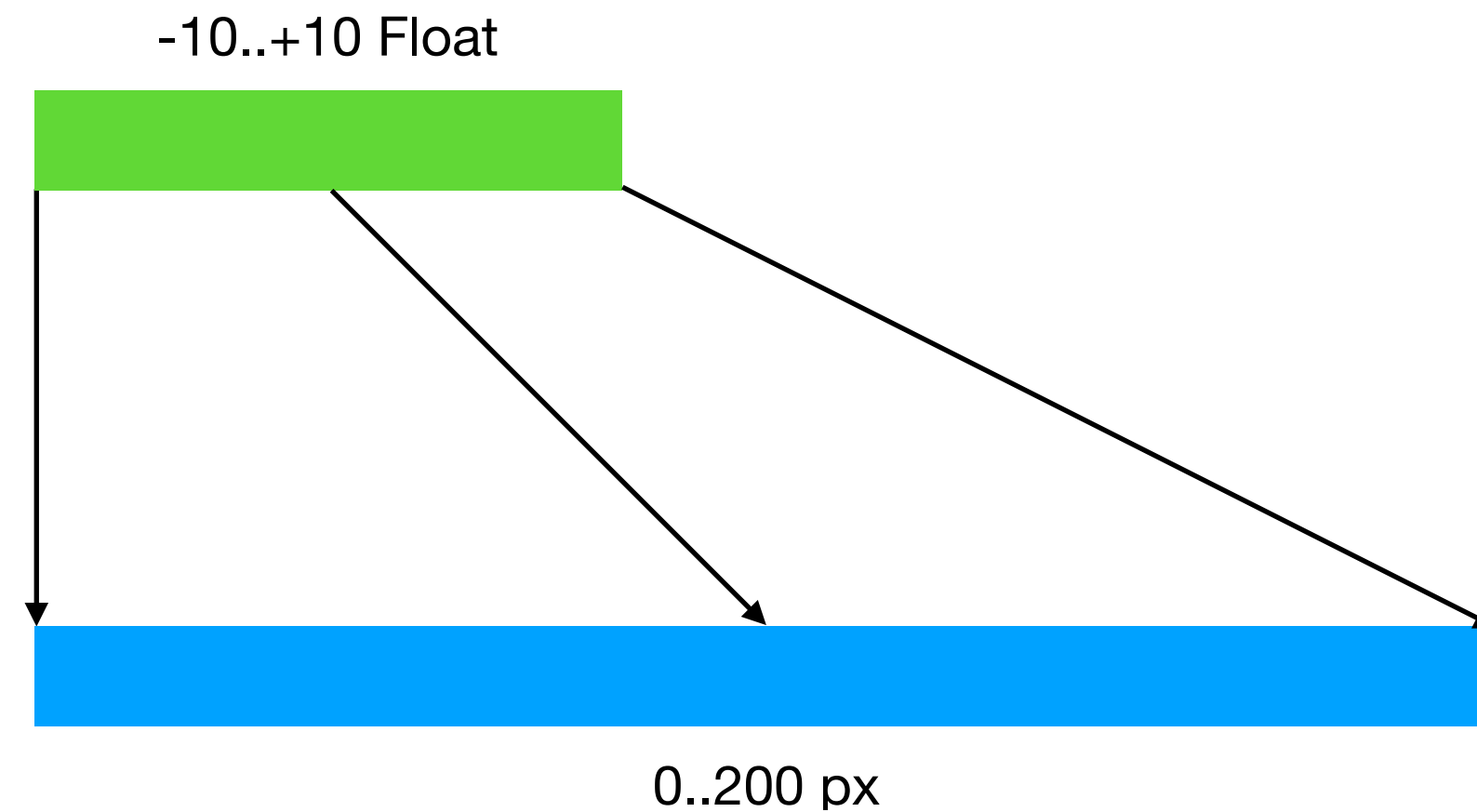


Source: [The Star Wars API](#)

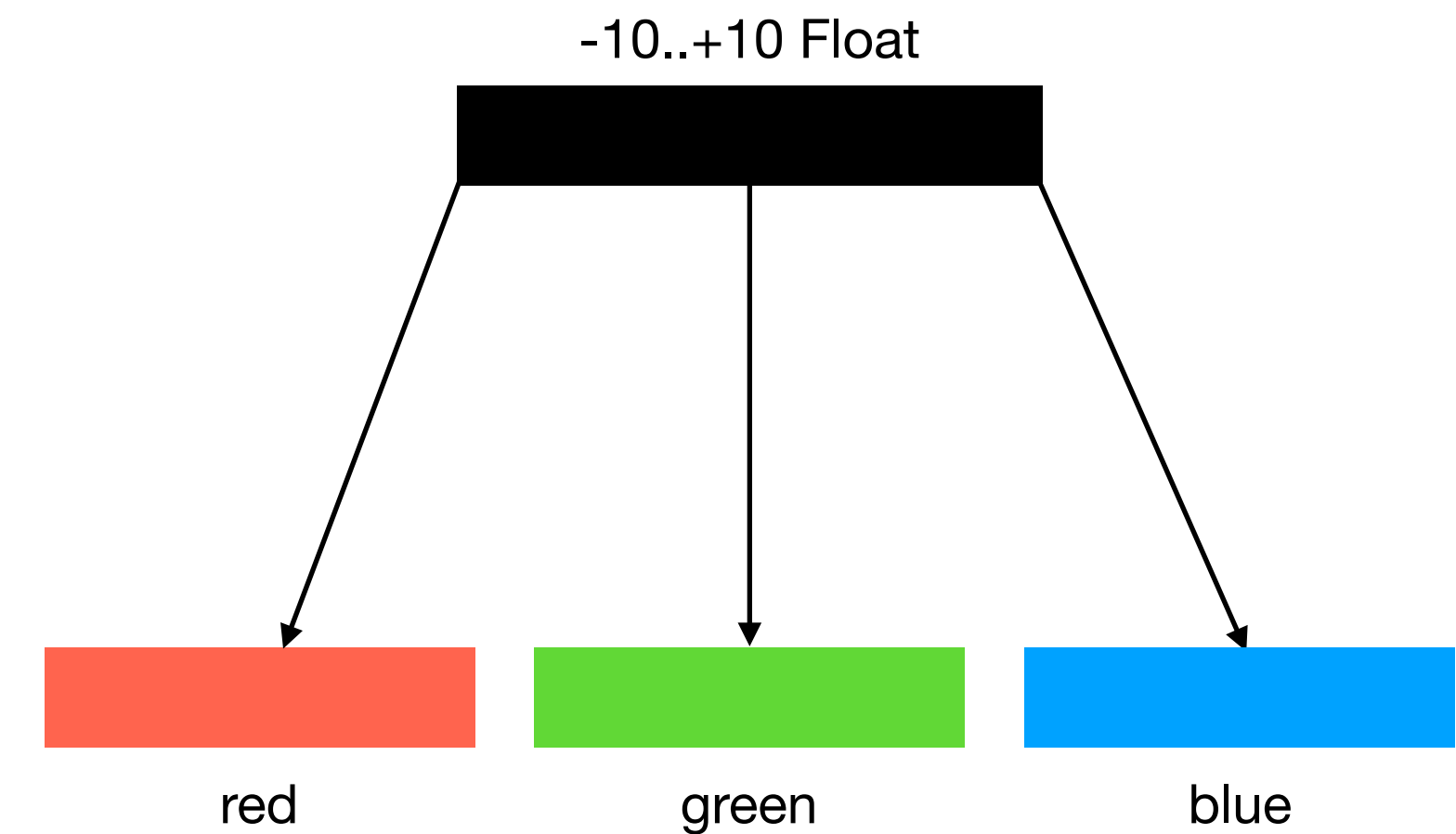
Skálák

absztrakt adat → vizuális reprezentáció

- lineáris
- kvantált
- ordinális
- stb.



```
d3.scaleLinear()  
  .domain([-10, 10])  
  .range([0, 200]);
```



```
d3.scaleQuantized()  
  .domain([-10, 10])  
  .range(['red', 'green', 'blue']);
```

Tengelyek

Skálák referenciaértékei, leolvasható formában

Elemei:

- scale – az adott skála, amit használ
- tick – jelölések, beállítható a darabszám, méret, címkeformátum, stb.

Címkék

Nem része a d3-nak, SVG text elemmel hozhatók létre

Szabadon pozicionálhatók, vegyük figyelembe a vízszintes és függőleges igazítást (**text-anchor**, **dominant-baseline**)

A **rotate** és **translate** transzformációk sorrendje fontos

Interaktivitás

Eseménykezelőkkel teremthető meg `on()` függvény

Standard DOM események használhatók

(<https://developer.mozilla.org/en-US/docs/Web/Events>)

Érdemes CSS osztályokkal megvalósítani az elemek vizuális tulajdonságainak módosítását -

`element.classList.add()`, `.remove()`, `.toggle()` függvények

`d3.event()`, `d3.mouse()` függvények megkönnyítik az esemény kezelését

Átmenetek

Animált állapotváltozások a kiválasztott elemeken → `selection`

`transition()` függvény használatával érhető el

Főbb jellemzők:

- `delay`
- `duration`
- `ease`
- `attr`
- `style`

Hierarchikus adatok és vizualizációk

Nesting: táblázatos adatok hierarchikus formába alakítása

- többszintű hierarchia alakítható ki a `key()` függvénnnyel
- a kulcsok és az értékek sorba rendezhetők (`sortKeys()`, `sortValues()`)

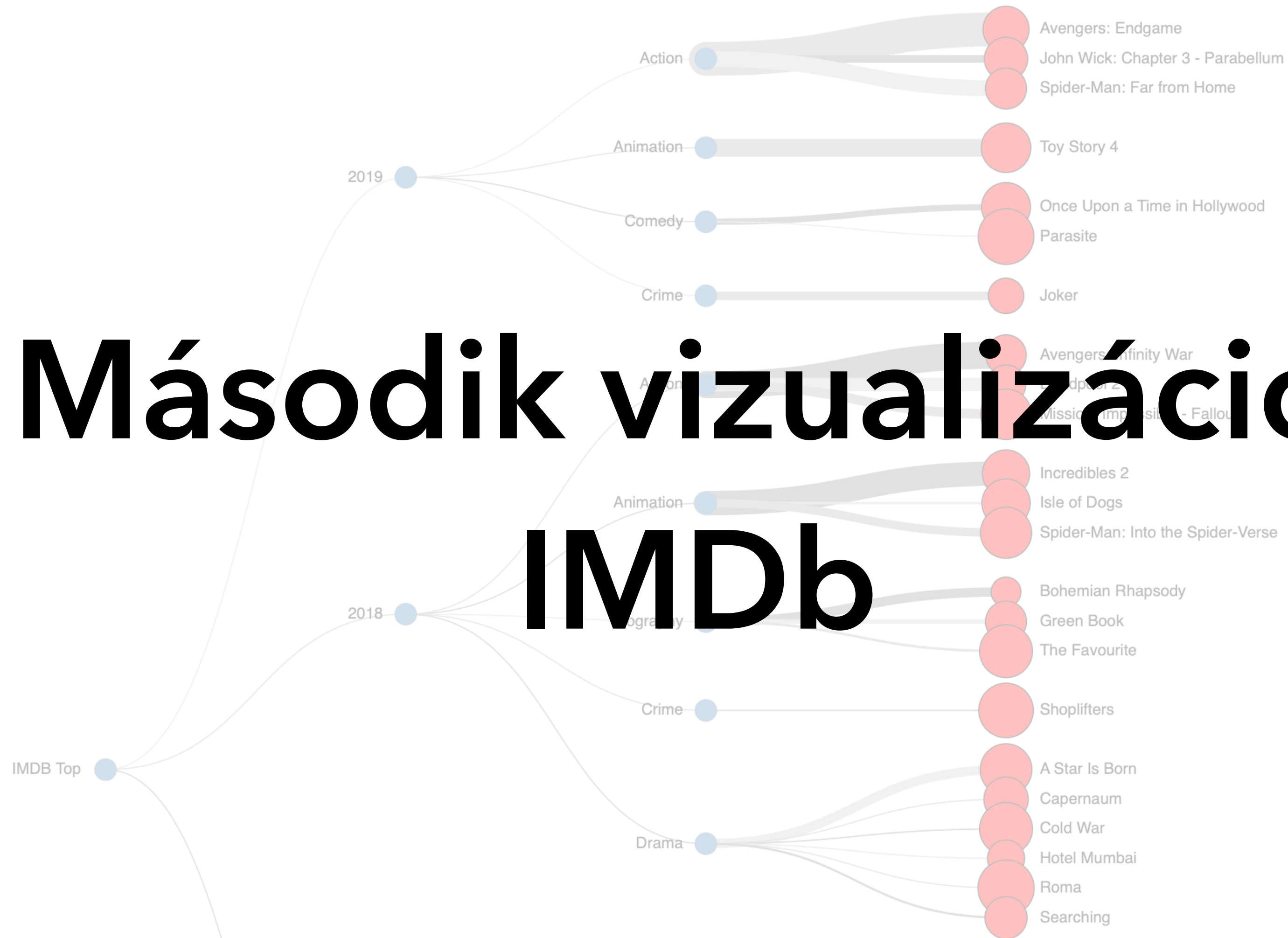
Rollup: a hierarchia alján lévő adatsorok összegzésére szolgál

A `d3.hierarchy()` függvénnnyel hozzuk létre a root elemet

- második paraméterként opcionálisan megadható a children accessor függvény

Második vizualizáció:

IMDb



Eszközök és könyvtárak

Observable

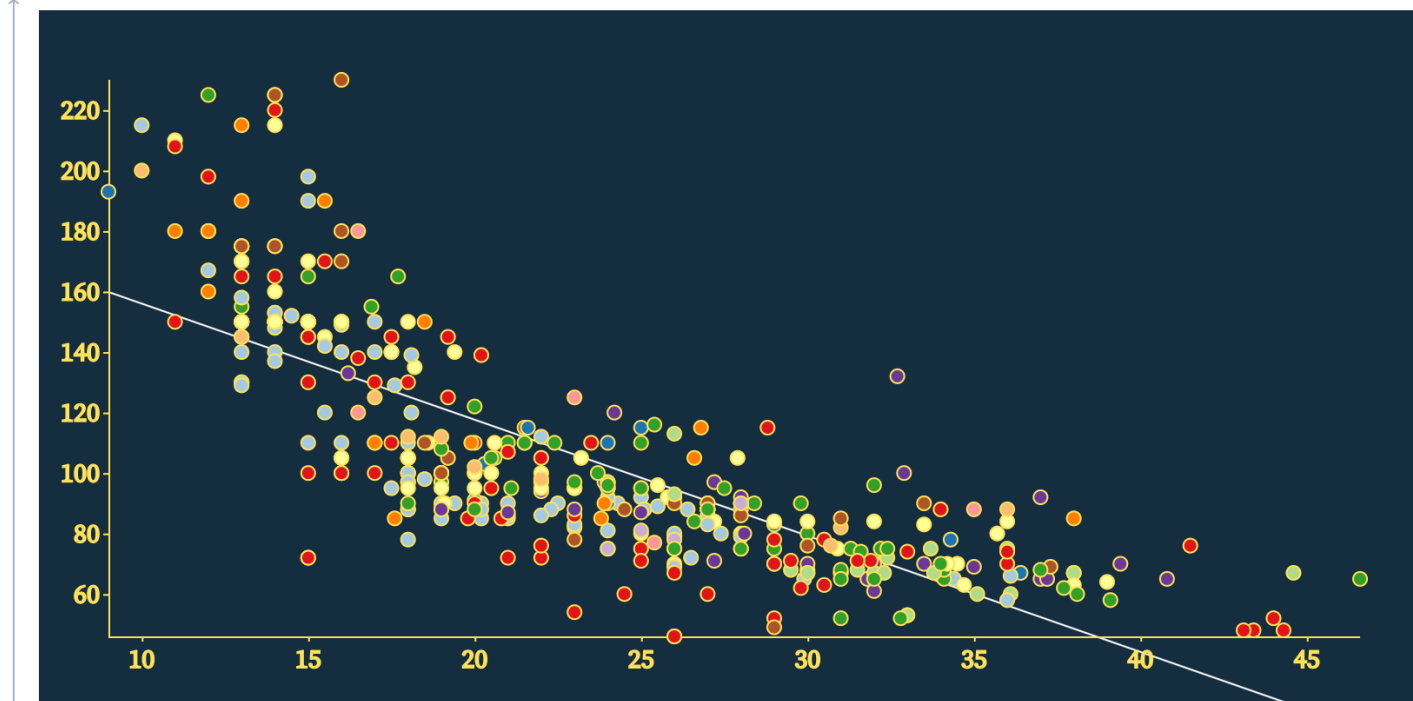
<https://observablehq.com>
data visualization notebook

Bl.ocks

<https://bl.ocks.org/>
GitHub Gist viewer

Scatterplot with trend

md`# Scatterplot with trend`



`render(chart({ data: tidyCars, width: 800, height: 400}))`

`line = f(...)`

```
line = function({ points , color = "white" }) {  
  const [[x1, y1], [x2, y2]] = points;  
  const lineGen = d3.line();  
  return svg`<path d="${lineGen(points)}" stroke="${color}" />`;  
}
```

`circle = f(...)`

```
circle = function ({ x, y, size = 4, color = '#ffe45b' }) {  
  return svg`<circle cx=${x} cy=${y} r=${size} fill=${color}></circle>`;  
}
```

Plotly

<https://plot.ly/javascript/>

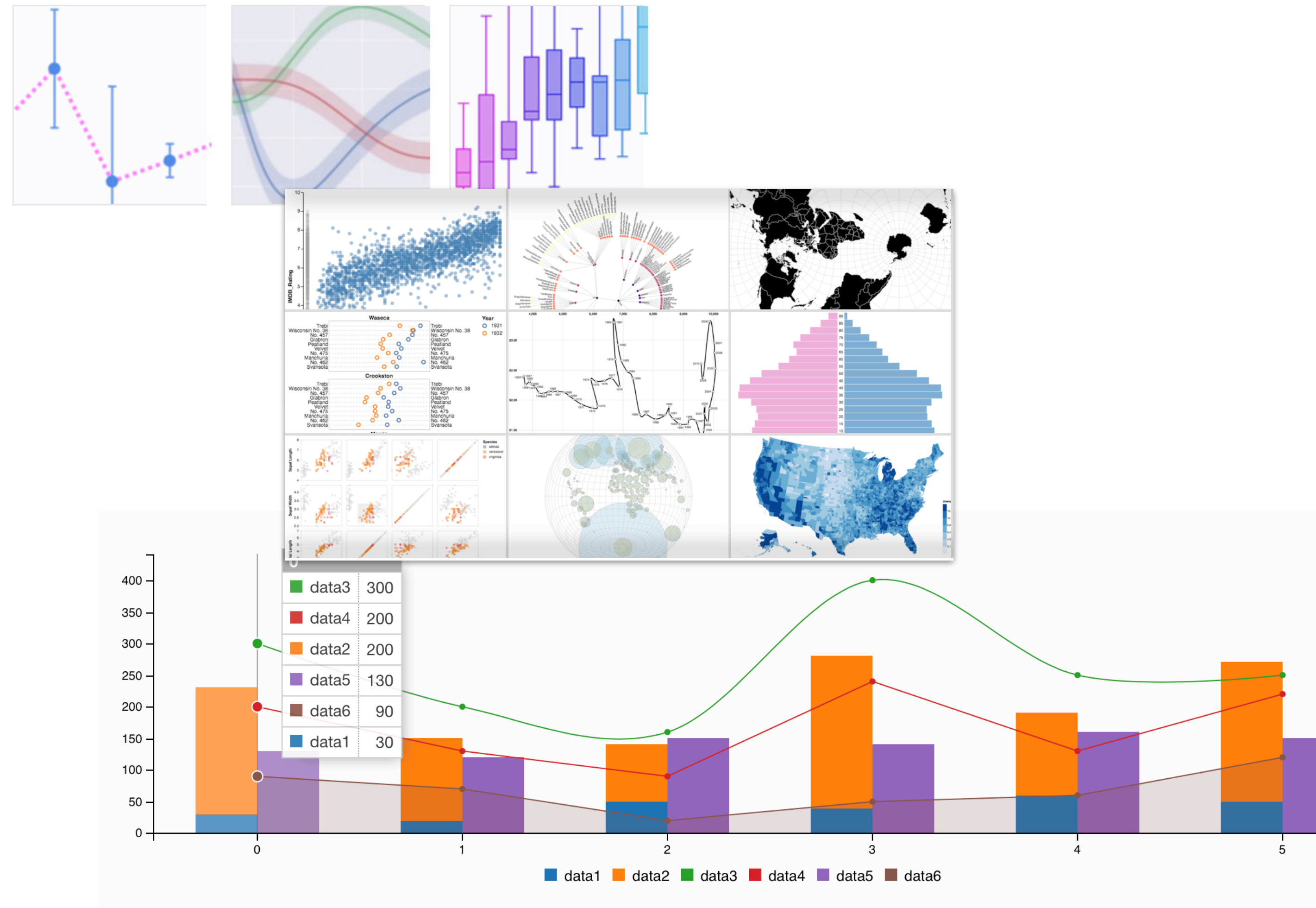
Vega & Vega-lite

<https://vega.github.io/vega/>

<https://vega.github.io/vega-lite/>

C3.js

<https://c3js.org>



Források

Blocks

<https://bl.ocks.org>

Mike Bostock's Blocks

<https://bl.ocks.org/mbostock>

Awesome D3

<https://github.com/wbkd/awesome-d3>

D3.js Graph Gallery

<https://www.d3-graph-gallery.com>

D3 Tips and Tricks

<https://leanpub.com/D3-Tips-and-Tricks>

(3-as verzióhoz készült, de jól
használható)