UNIVERSITÄT BERN

Übung Open Data:

Daten einbinden in D3.js und Anpassen von bestehenden D3.js Code Snippets

Termin 5, 19. März 2015

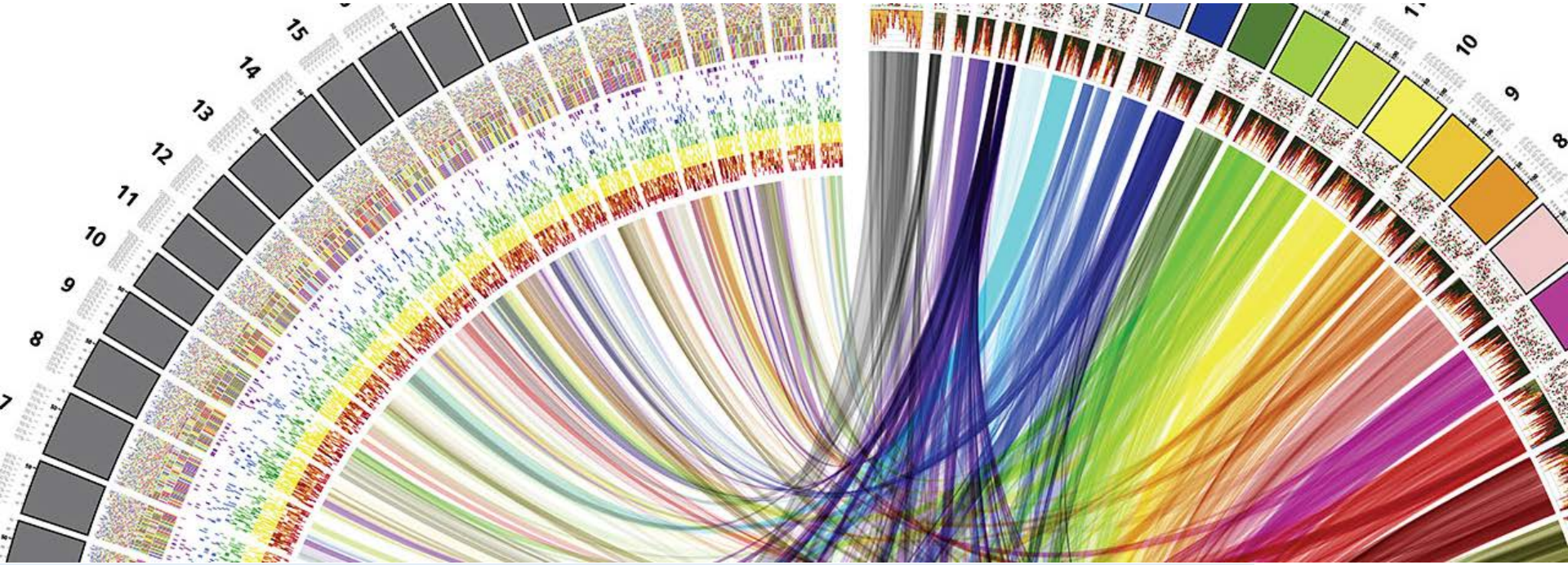
Dr. Matthias Stürmer und Prof. Dr. Thomas Myrach

Universität Bern, Institut für Wirtschaftsinformatik
Abteilung Informationsmanagement
Forschungsstelle Digitale Nachhaltigkeit

UNIVERSITÄT BERN

Agenda

1. **Theorie: Daten einbinden in D3.js (Kapitel 5)**
2. Praxis: Anpassen bestehender D3.js Snippets

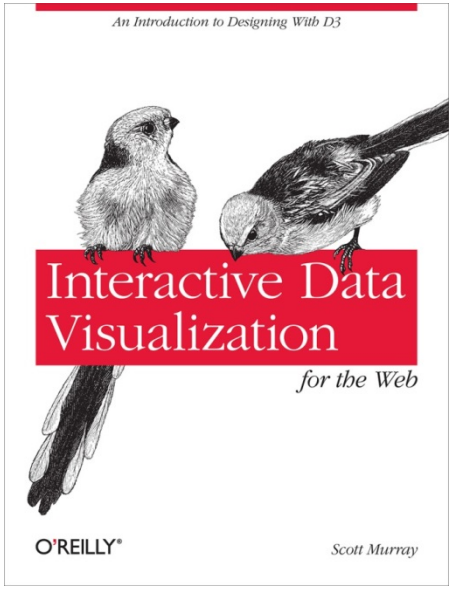


Interactive Data Visualization for the Web

Chapter 5. Data:

- *Data* is an extremely broad term, only slightly less vague than the nearly all-encompassing*information*. What is data? (What *isn't* data?) What kinds of data are there, and what can we use with D3?

- http://chimera.labs.oreilly.com/books/1230000000345/ch05.html



Chaining Methods

```
d3.select("body").append("p").text("New paragraph!");
```

ist der gleiche Code wie

```
d3.select("body")
    .append("p")
    .text("New paragraph!");
```

ist der gleiche Code wie

```
var body = d3.select("body");
var p = body.append("p");
p.text("New paragraph!");
```

Chaining Methods

```
d3.select("body").append("p").text("New paragraph!");
```

D3	References the D3 object, so we can access its methods. Our D3 adventure begins here.
.select("body")	Give the select() method a CSS selector as input, and it will return a reference to the first element in the DOM that matches.
.append("p")	append() creates whatever new DOM element you specify and appends it to the end (but just inside) of whatever selection it's acting on. Finally, append() hands off a reference to the new element it just created.
.text("New paragraph!")	text() takes a string and inserts it between the opening and closing tags of the current selection.
;	The all-important semicolon indicates the end of this line of code. Chain over.

Binding Data

```
d3.select("body").selectAll("p")
    .data(dataset)
    .enter()
    .append("p")
    .text("New paragraph!");
```

`d3.select("body")`	Finds the body in the DOM and hands off a reference to the next step in the chain.
`.selectAll("p")`	Selects all paragraphs in the DOM. Because none exist yet, this returns an empty selection. Think of this empty selection as representing the paragraphs that will soon exist.
`.data(dataset)`	Counts and parses our data values. There are five values in our array called dataset, so everything past this point is executed five times, once for each value.
`.enter()`	To create new, data-bound elements, you must use enter(). This method looks at the current DOM selection, and then at the data being handed to it. If there are more data values than corresponding DOM elements, then enter() creates a new placeholder element on which you can work your magic. It then hands off a reference to this new placeholder to the next step in the chain.
`.append("p")`	Takes the empty placeholder selection created by enter() and appends a p element into the DOM. Hooray! Then it hands off a reference to the element it just created to the next step in the chain.
`.text("New paragraph!")`	Takes the reference to the newly created p and inserts a text value.

Using Your Data

Die Testdaten:

```
var dataset = [ 5, 10, 15, 20, 25 ];
```

Anzeigen der Testdaten:

```
d3.select("body").selectAll("p")
     .data(dataset)
     .enter()
     .append("p")
     .text(function(d) {return "I can count up to " + d; });
```

Nur die letzte Zeile anpassen:

```
     .style("color", function(d) {
        if (d > 15) {   //Threshold of 15
            return "red";
        } else {
            return "black";
        }
     });
```

D3.js API Reference

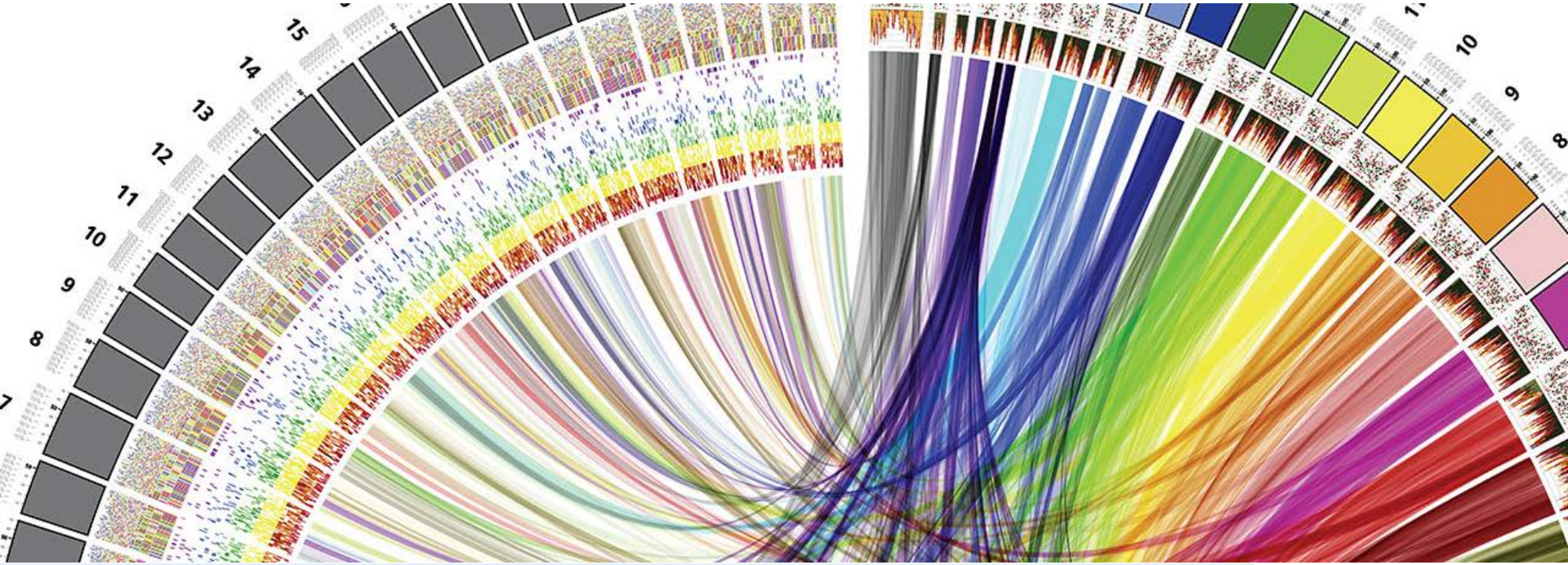
Application Programming Interface von D3.js:
https://github.com/mbostock/d3/wiki/API-Reference

- Selections - manipulate elements in the current document.
- Transitions - interpolate attributes and styles smoothly over time.
- Arrays - manipulate arrays and objects with ease.
- Requests - load external data.
- Formatting - convert numbers, dates and other objects to strings.
- Localization - control locale-specific behavior, such as number formatting.
- Colors - parse and manipulate colors; work with color spaces.
- Namespaces - extend D3's support for XML namespaces.
- Math - miscellaneous mathematical functions.
- Internals - sundry utilities for extending D3.

UNIVERSITÄT BERN

Agenda

1. Theorie: Daten einbinden in D3.js (Kapitel 5)
2. **Praxis: Anpassen bestehender D3.js Snippets**



Anpassen bestehender D3.js Snippets

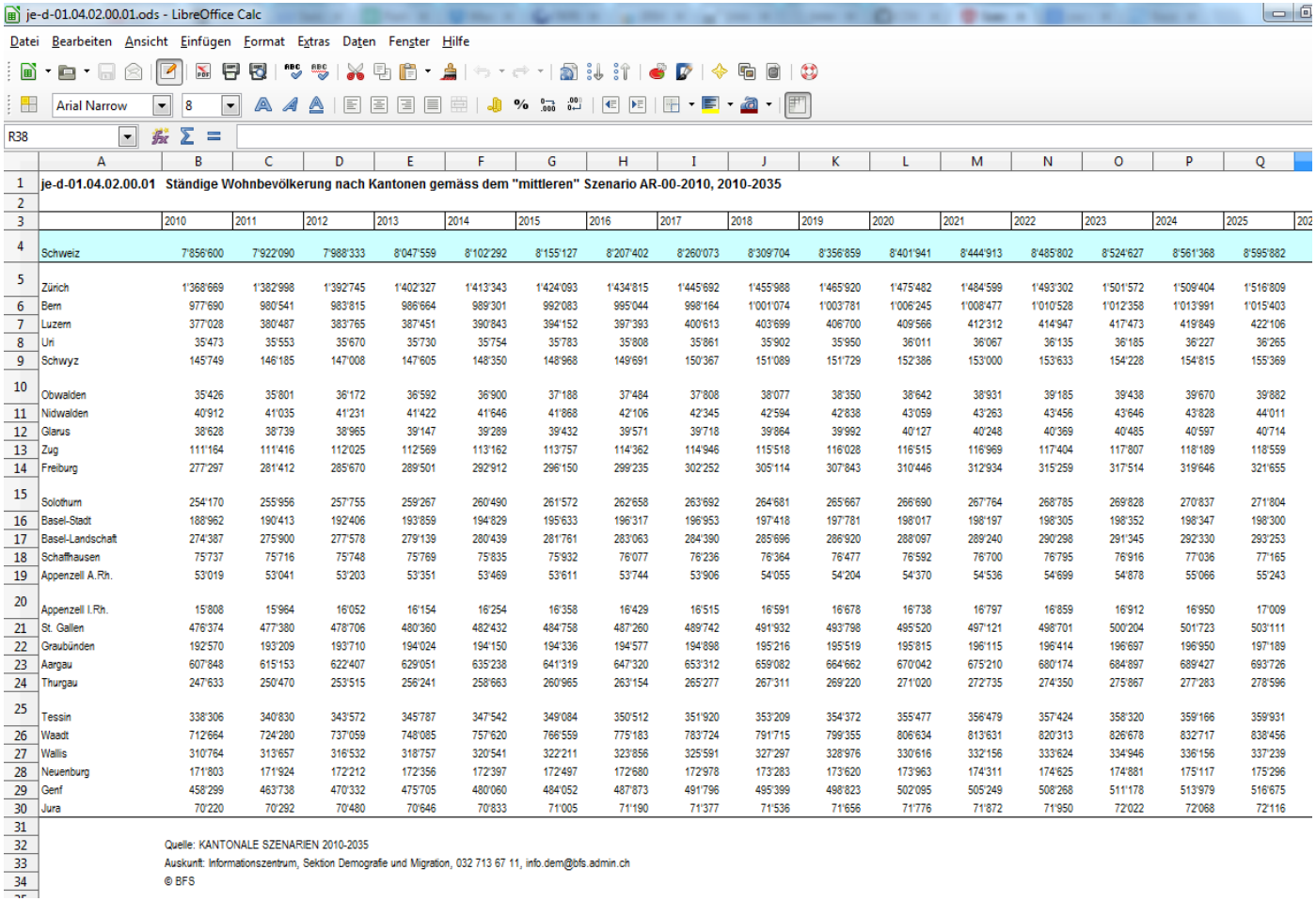
Vorgehen:

1. **Welche Daten werden visualisiert? -> Bevölkerungswachstum**
2. Welche Visualisierungsart macht Sinn? -> D3 Show Real
3. Wie muss ich die Daten anpassen? -> Transponieren
4. Was muss ich am Code Snippet anpassen -> Pfade
5. Was läuft schief? -> Debugging

Daten von opendata.admin.ch holen

Link: http://opendata.admin.ch/de/dataset/je-d-01-04-02-00-01

ODS Datei mit LibreOffice geöffnet



je-d-01.04.02.00.01.ods - LibreOffice Calc

Datei Bearbeiten Ansicht Einfügen Format Extras Daten Fenster Hilfe

Arial Narrow 8

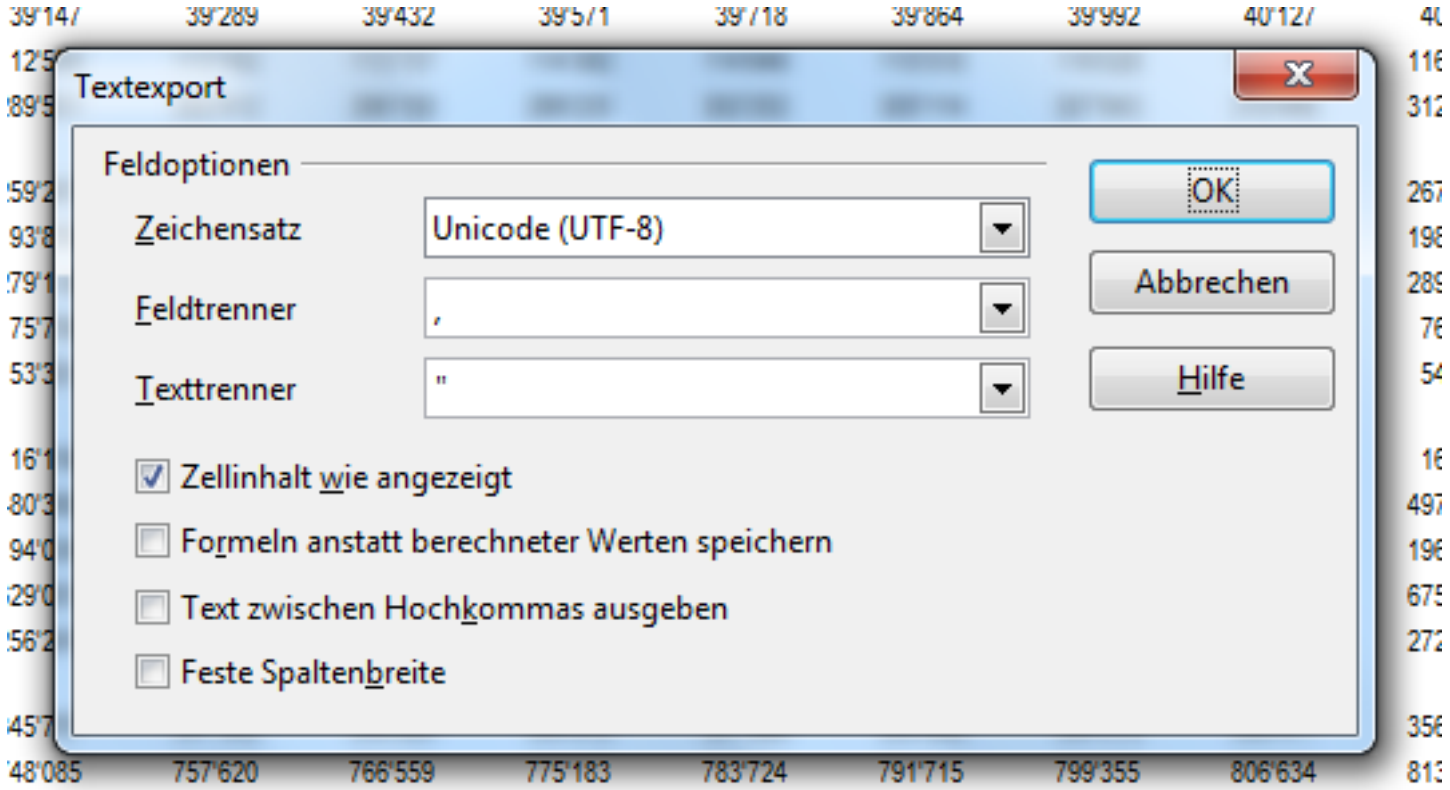
R38

je-d-01.04.02.00.01 Ständige Wohnbevölkerung nach Kantonen gemäss dem "mittleren" Szenario AR-00-2010, 2010-2035

	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	202
Schweiz	7'856'600	7'922'090	7'988'333	8'047'559	8'102'292	8'155'127	8'207'402	8'260'073	8'309'704	8'356'859	8'401'941	8'444'913	8'485'802	8'524'627	8'561'368	8'595'882	
Zürich	1'368'669	1'382'998	1'392'745	1'402'327	1'413'343	1'424'093	1'434'815	1'445'692	1'455'988	1'465'920	1'475'482	1'484'599	1'493'302	1'501'572	1'509'404	1'516'809	
Bern	977'690	980'541	983'815	986'664	989'301	992'083	995'044	998'164	1'001'074	1'003'781	1'006'245	1'008'477	1'010'528	1'012'358	1'013'991	1'015'403	
Luzern	377'028	380'487	383'765	387'451	390'843	394'152	397'393	400'613	403'699	406'700	409'566	412'312	414'947	417'473	419'849	422'106	
Uri	35'473	35'553	35'670	35'730	35'754	35'783	35'808	35'861	35'902	35'950	36'011	36'067	36'135	36'185	36'227	36'265	
Schwyz	145'749	146'185	147'008	147'605	148'350	148'968	149'691	150'367	151'089	151'729	152'386	153'000	153'633	154'228	154'815	155'369	
Obwalden	35'426	35'801	36'172	36'592	36'900	37'188	37'484	37'808	38'077	38'350	38'642	38'931	39'185	39'438	39'670	39'882	
Nidwalden	40'912	41'035	41'231	41'422	41'646	41'868	42'106	42'345	42'594	42'838	43'059	43'263	43'456	43'646	43'828	44'011	
Glarus	38'628	38'739	38'965	39'147	39'289	39'432	39'571	39'718	39'864	39'992	40'127	40'248	40'369	40'485	40'597	40'714	
Zug	111'164	111'416	112'025	112'569	113'162	113'757	114'362	114'946	115'518	116'028	116'515	116'969	117'404	117'807	118'189	118'559	
Freiburg	277'297	281'412	285'670	289'501	292'912	296'150	299'235	302'252	305'114	307'843	310'446	312'934	315'259	317'514	319'646	321'655	
Solothurn	254'170	255'956	257'755	259'267	260'490	261'572	262'658	263'692	264'681	265'667	266'690	267'764	268'785	269'828	270'837	271'804	
Basel-Stadt	188'962	190'413	192'406	193'859	194'829	195'633	196'317	196'953	197'418	197'781	198'017	198'197	198'305	198'352	198'347	198'300	
Basel-Landschaft	274'387	275'900	277'578	279'139	280'439	281'761	283'063	284'390	285'696	286'920	288'097	289'240	290'298	291'345	292'330	293'253	
Schaffhausen	75'737	75'716	75'748	75'769	75'835	75'932	76'077	76'236	76'364	76'477	76'592	76'700	76'795	76'916	77'036	77'165	
Appenzell A.Rh.	53'019	53'041	53'203	53'351	53'469	53'611	53'744	53'906	54'055	54'204	54'370	54'536	54'699	54'878	55'066	55'243	
Appenzell I.Rh.	15'808	15'964	16'052	16'154	16'254	16'358	16'429	16'515	16'591	16'678	16'738	16'797	16'859	16'912	16'950	17'009	
St. Gallen	476'374	477'380	478'706	480'360	482'432	484'758	487'260	489'742	491'932	493'798	495'520	497'121	498'701	500'204	501'723	503'111	
Graubünden	192'570	193'209	193'710	194'024	194'150	194'336	194'577	194'898	195'216	195'519	195'815	196'115	196'414	196'697	196'950	197'189	
Aargau	607'848	615'153	622'407	629'051	635'238	641'319	647'320	653'312	659'082	664'662	670'042	675'210	680'174	684'897	689'427	693'726	
Thurgau	247'633	250'470	253'515	256'241	258'663	260'965	263'154	265'277	267'311	269'220	271'020	272'735	274'350	275'867	277'283	278'596	
Tessin	338'306	340'830	343'572	345'787	347'542	349'084	350'512	351'920	353'209	354'372	355'477	356'479	357'424	358'320	359'166	359'931	
Waadt	712'664	724'280	737'059	748'085	757'620	766'559	775'183	783'724	791'715	799'355	806'634	813'631	820'313	826'678	832'717	838'456	
Wallis	310'764	313'657	316'532	318'757	320'541	322'211	323'856	325'591	327'297	328'976	330'616	332'156	333'624	334'946	336'156	337'239	
Neuenburg	171'803	171'924	172'212	172'356	172'397	172'497	172'680	172'978	173'283	173'620	173'963	174'311	174'625	174'881	175'117	175'296	
Genf	458'299	463'738	470'332	475'705	480'060	484'052	487'873	491'796	495'399	498'823	502'095	505'249	508'268	511'178	513'979	516'675	
Jura	70'220	70'292	70'480	70'646	70'833	71'005	71'190	71'377	71'536	71'656	71'776	71'872	71'950	72'022	72'068	72'116	

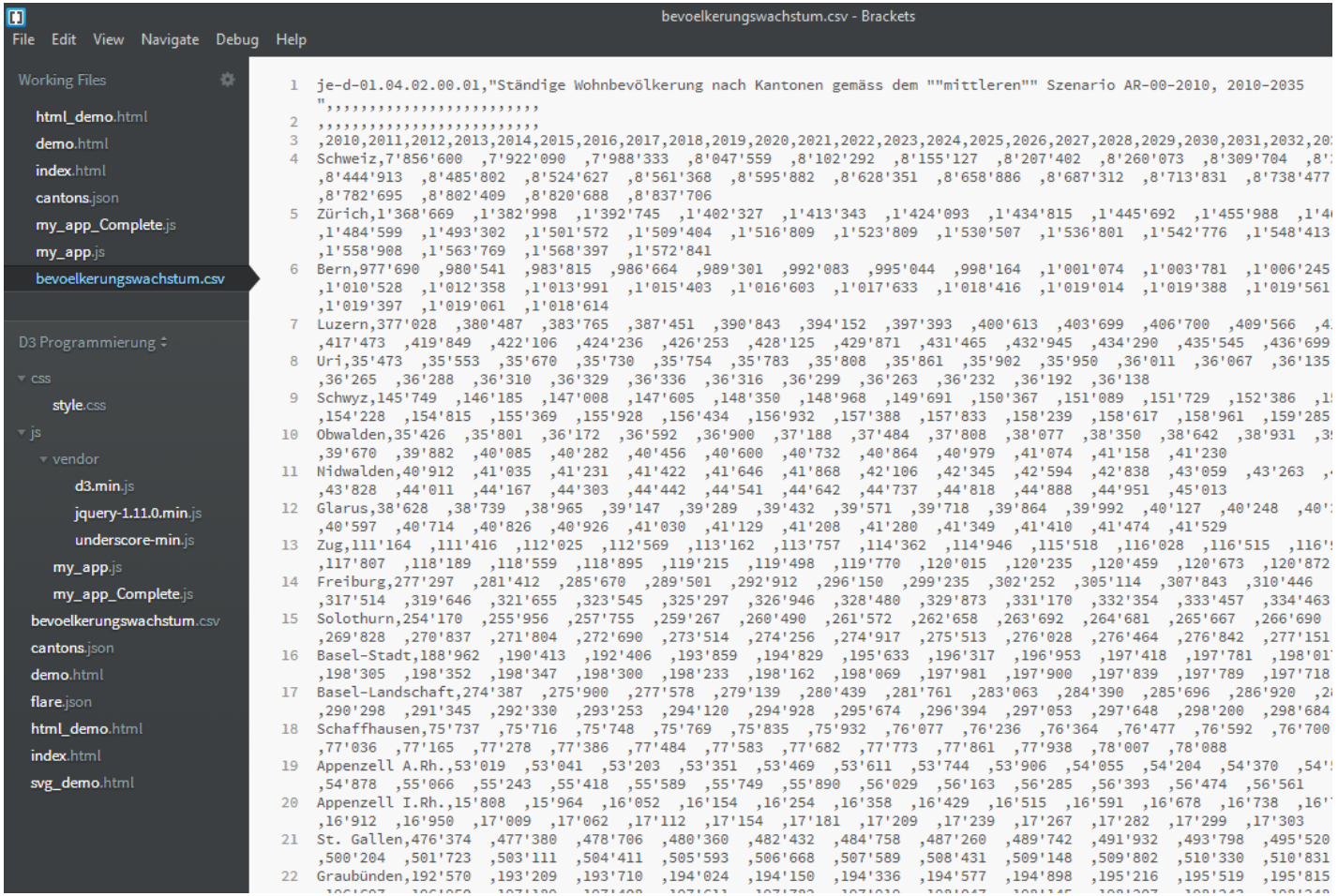
Quelle: KANTONALE SZENARIEN 2010-2035
Auskunft: Informationszentrum, Sektion Demografie und Migration, 032 713 67 11, info.dem@bfs.admin.ch
© BFS

Export der Daten als UTF-8 CSV-Datei

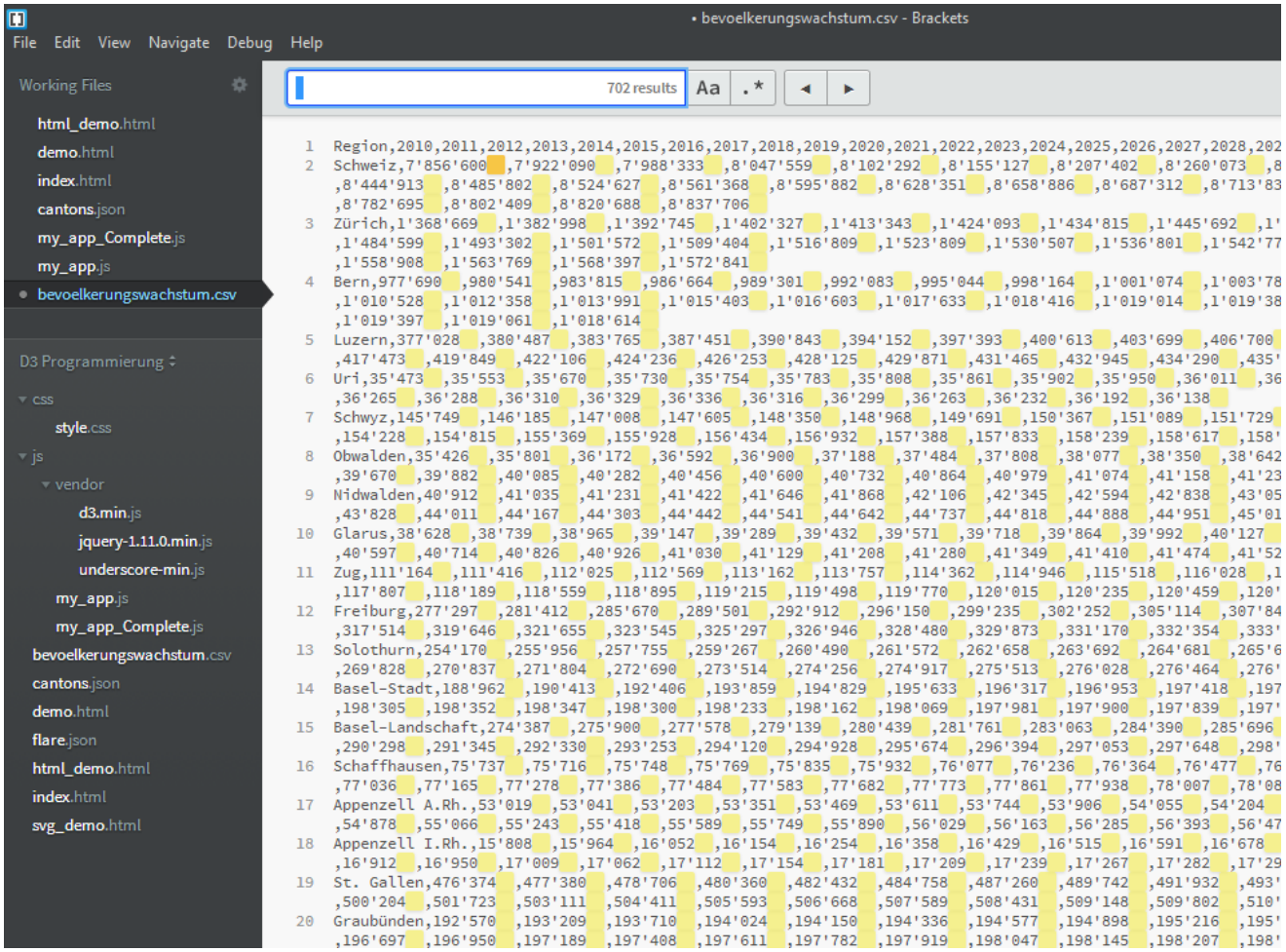


UNIVERSITÄT BERN

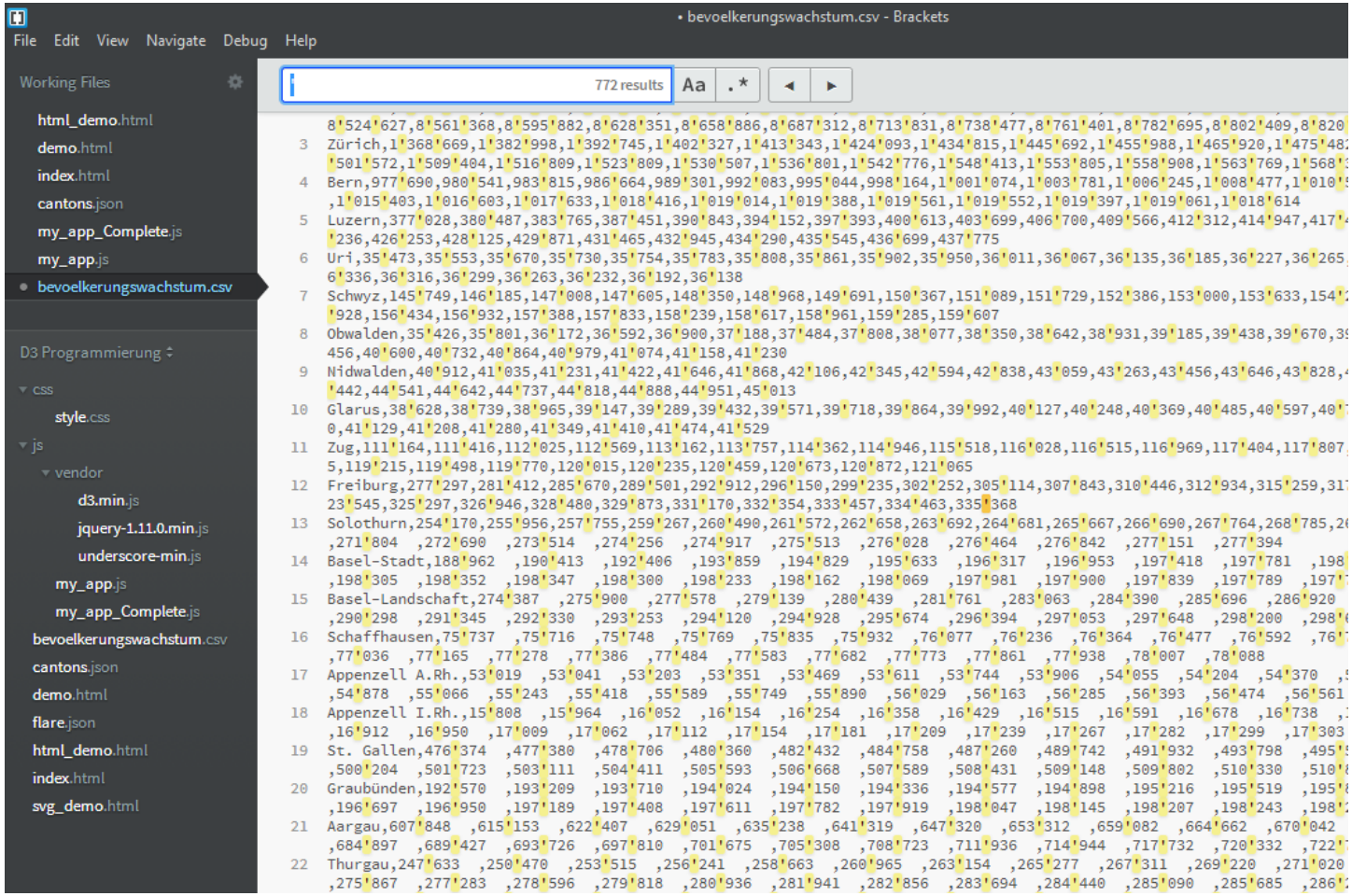
Daten als CSV in Brackets öffnen



Daten bereinigen: Kommentare und doppelte Zwischenräume löschen



Daten bereinigen: Hochkommas löschen

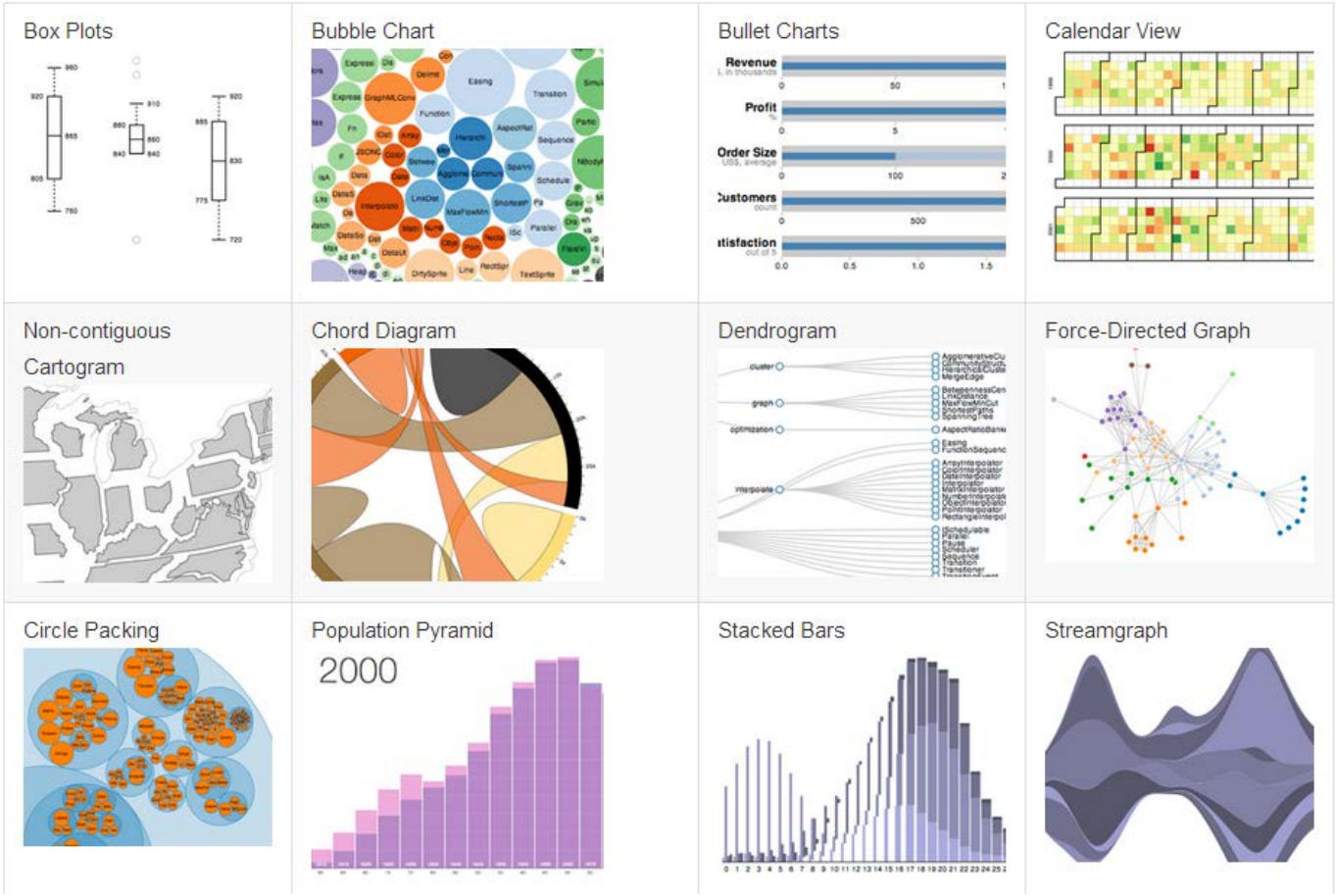


Anpassen bestehender D3.js Snippets

Vorgehen:

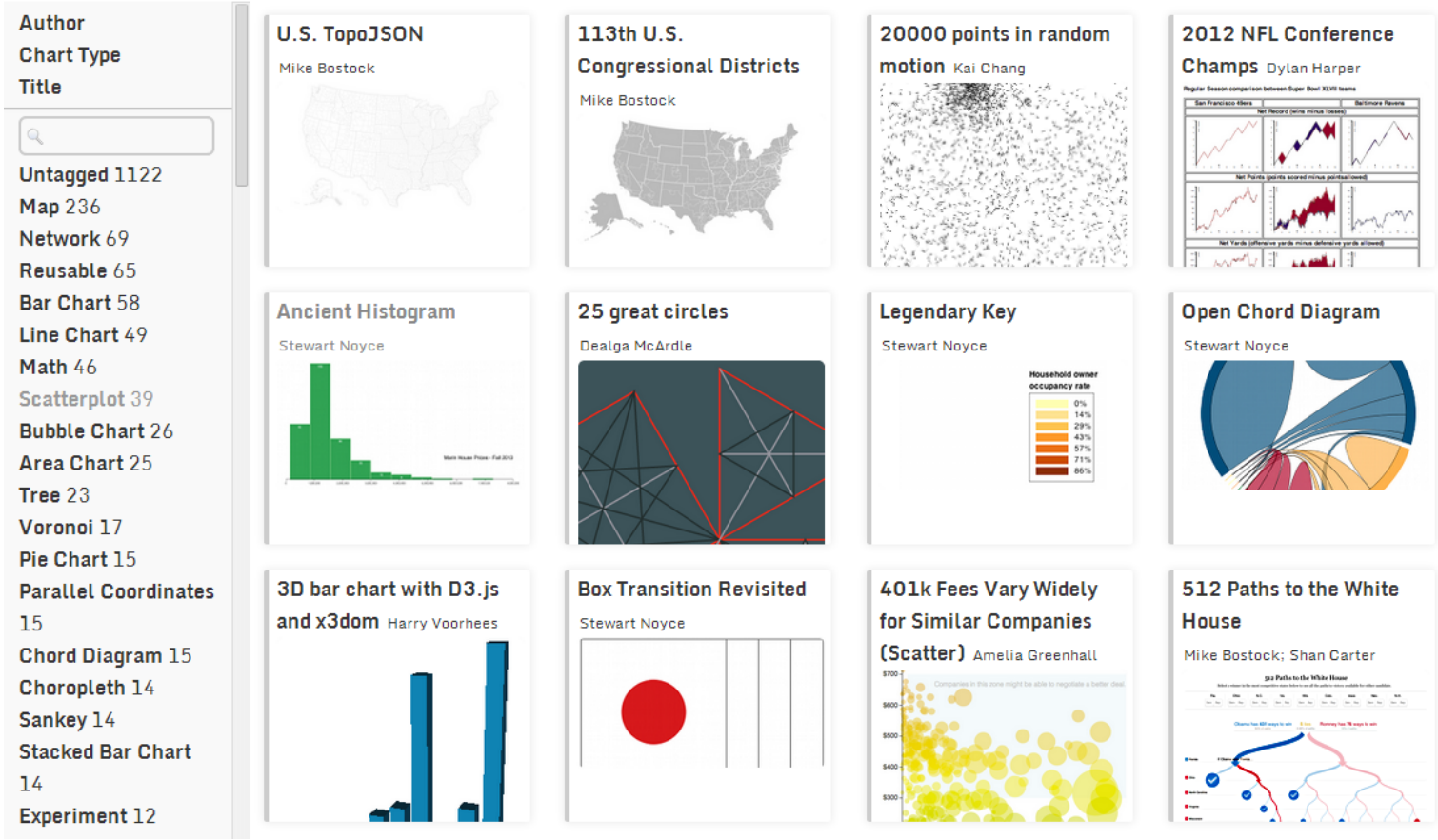
1. Welche Daten werden visualisiert? -> Bevölkerungswachstum
2. **Welche Visualisierungsart macht Sinn? -> D3 Show Real**
3. Wie muss ich die Daten anpassen? -> Transponieren
4. Was muss ich am Code Snippet anpassen -> Pfade
5. Was läuft schief? -> Debugging

D3.js Code Snippets



Link: https://github.com/mbostock/d3/wiki/Gallery

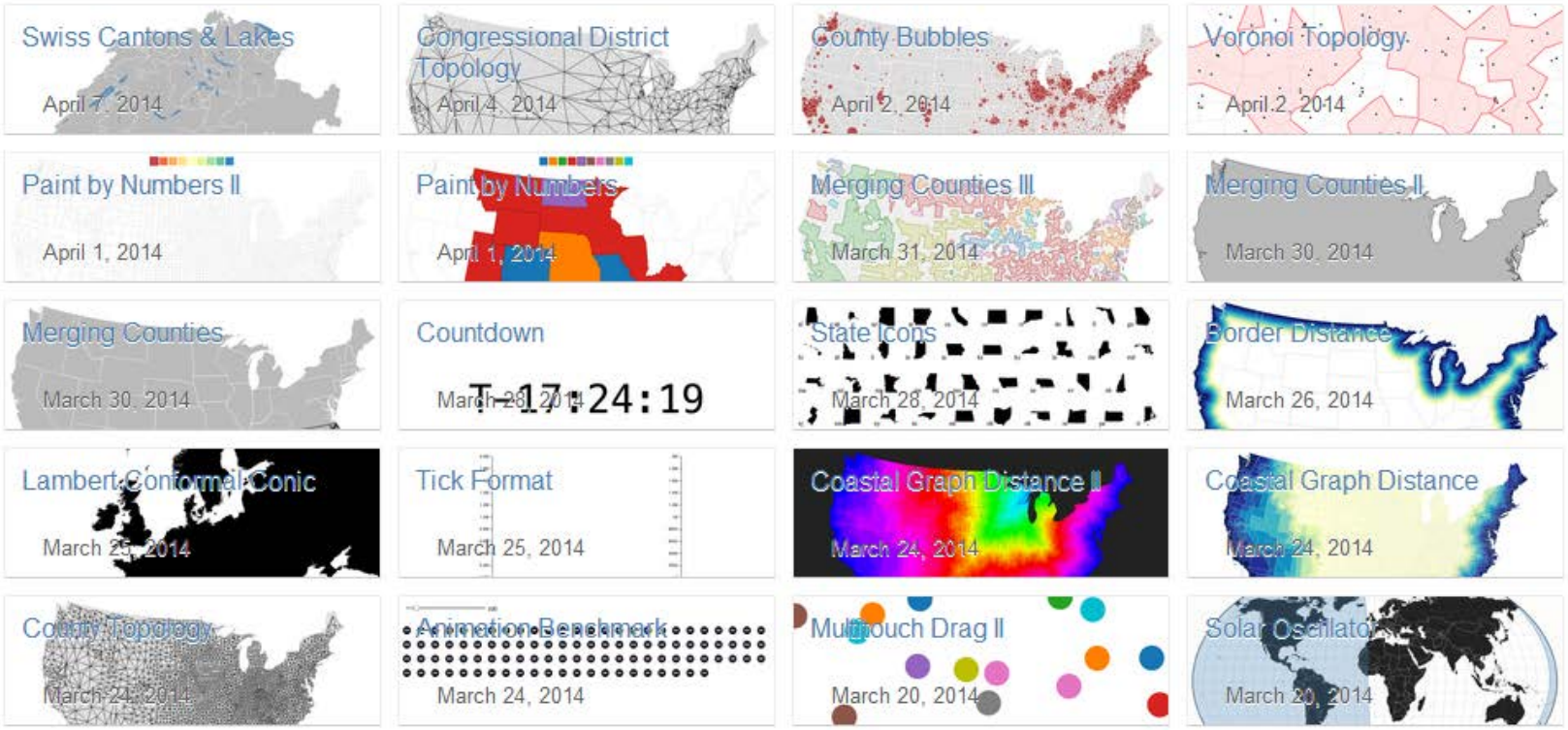
D3.js Code Snippets



Link: http://christopheviau.com/d3list/gallery.html

D3.js Code Snippets

mbostock's blocks

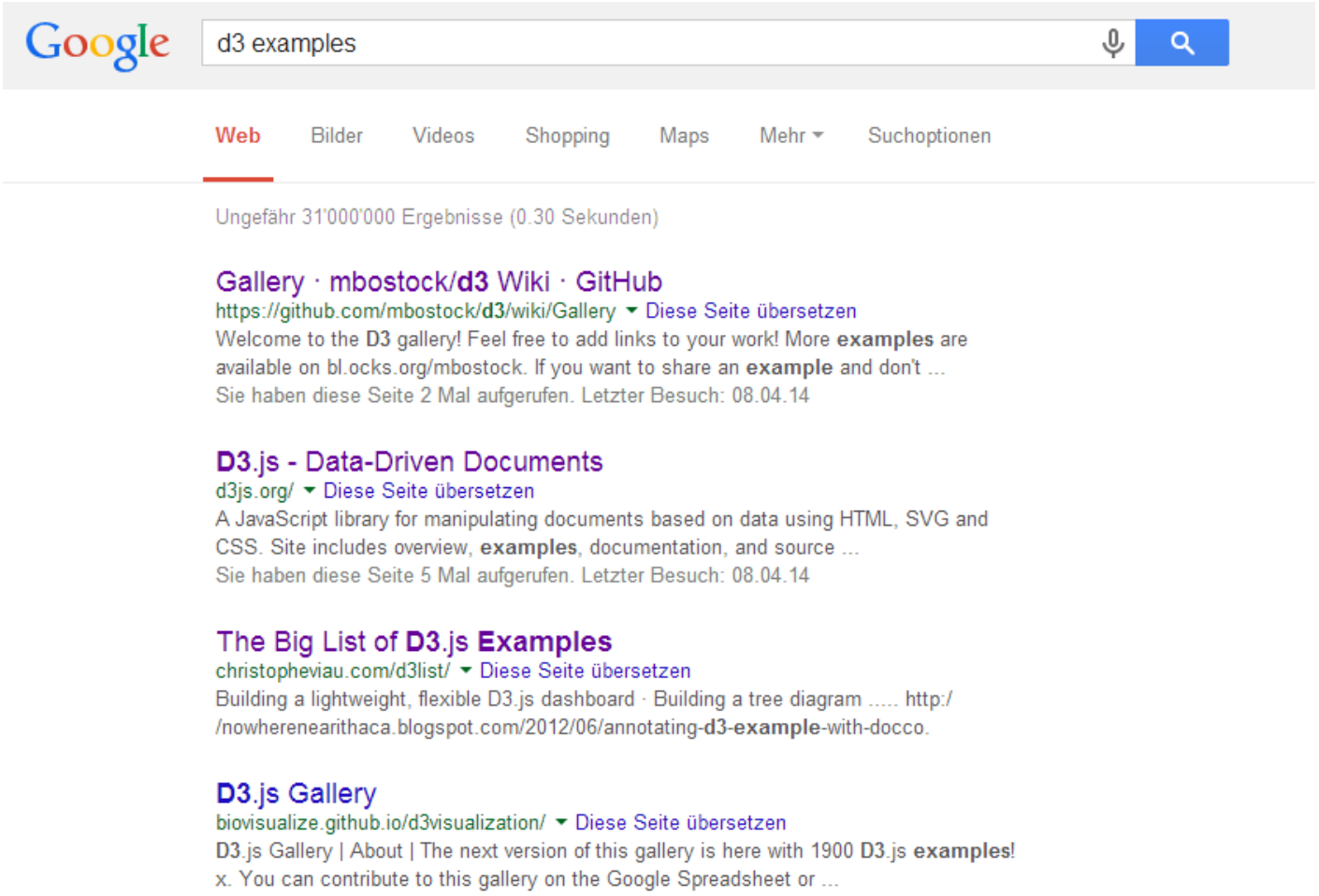


Link: http://bl.ocks.org/mbostock

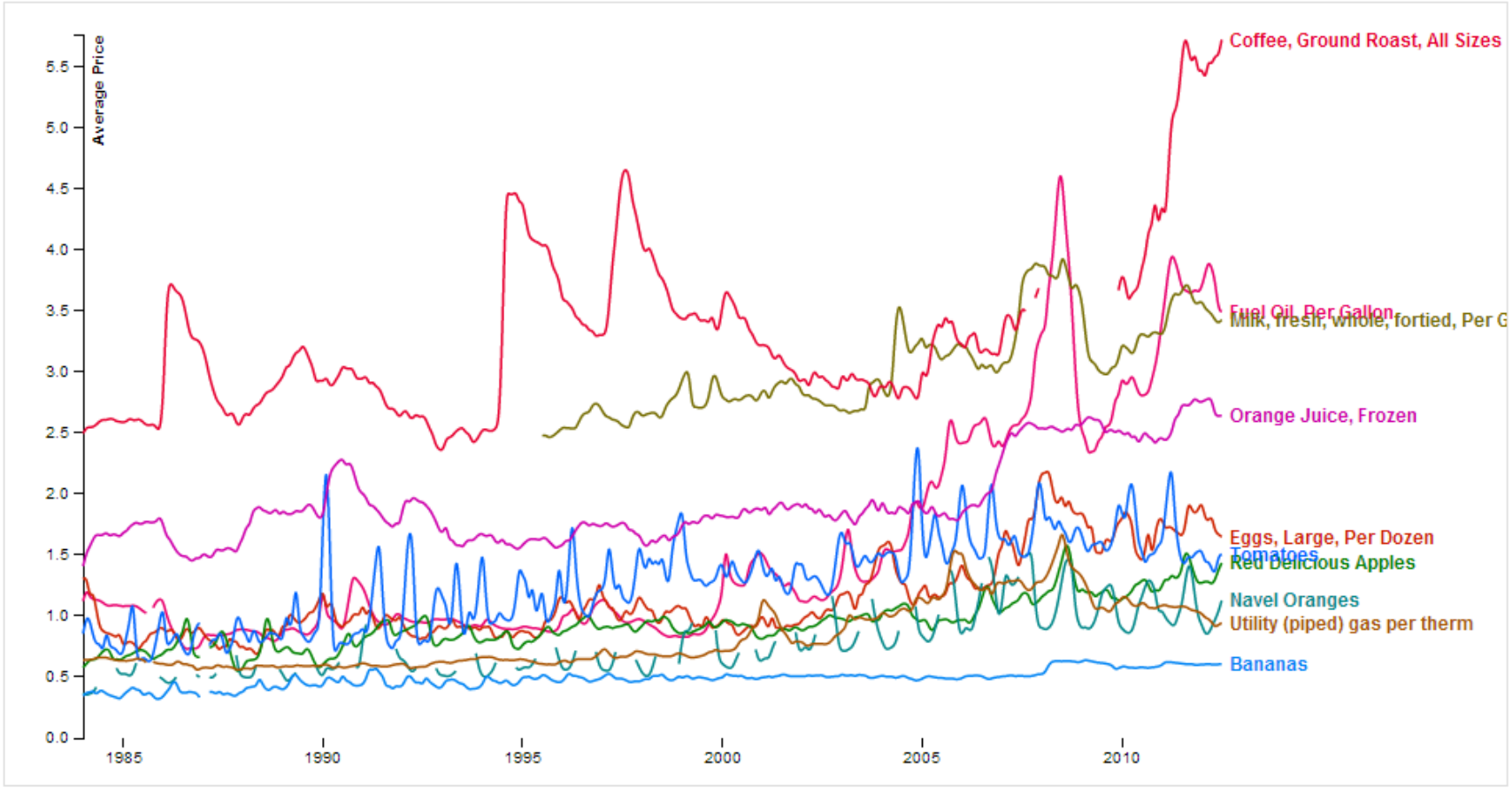
D3.js Code Snippets

Link: http://bost.ocks.org/mike/

Es gibt viele D3 Snippets im Internet...

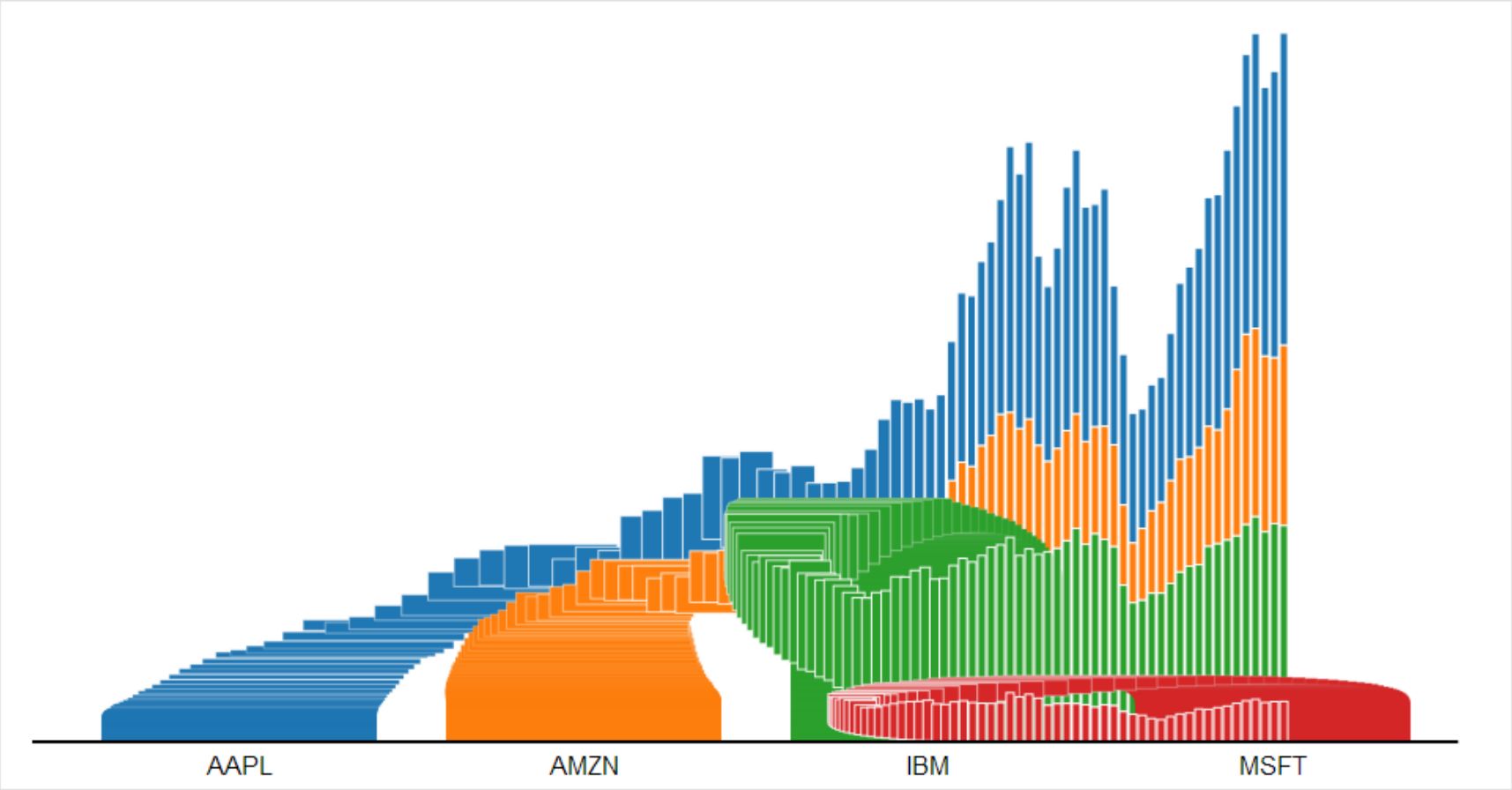


Auswahl einer passenden Visualisierung



Link: http://bl.ocks.org/syntagmatic/3891711

Auswahl einer passenden Visualisierung: D3 Show Reel



Link: http://bl.ocks.org/mbostock/1256572

Code und Datenformat von D3 Show Reel analysieren

index.html

```
<!DOCTYPE html>
<meta charset="utf-8">
<style type="text/css">

svg {
  font-family: "Helvetica Neue", Helvetica;
}

.line {
  fill: none;
  stroke: #000;
  stroke-width: 2px;
}

</style>
<body>
<script src="http://d3js.org/d3.v3.min.js"></script>
<script>

var m = [20, 20, 30, 20],
    w = 960 - m[1] - m[3],
    h = 500 - m[0] - m[2];

var x,
    y,
    duration = 1500,
    delay = 500;
```

stocks.csv

```
symbol,date,price
MSFT,Jan 2000,39.81
MSFT,Feb 2000,36.35
MSFT,Mar 2000,43.22
MSFT,Apr 2000,28.37
MSFT,May 2000,25.45
MSFT,Jun 2000,32.54
MSFT,Jul 2000,28.4
MSFT,Aug 2000,28.4
MSFT,Sep 2000,24.53
MSFT,Oct 2000,28.02
MSFT,Nov 2000,23.34
MSFT,Dec 2000,17.65
MSFT,Jan 2001,24.84
MSFT,Feb 2001,24
MSFT,Mar 2001,22.25
MSFT,Apr 2001,27.56
MSFT,May 2001,28.14
MSFT,Jun 2001,29.7
MSFT,Jul 2001,26.93
MSFT,Aug 2001,23.21
MSFT,Sep 2001,20.82
MSFT,Oct 2001,23.65
MSFT,Nov 2001,26.12
MSFT,Dec 2001,26.95
MSFT,Jan 2002,25.92
```

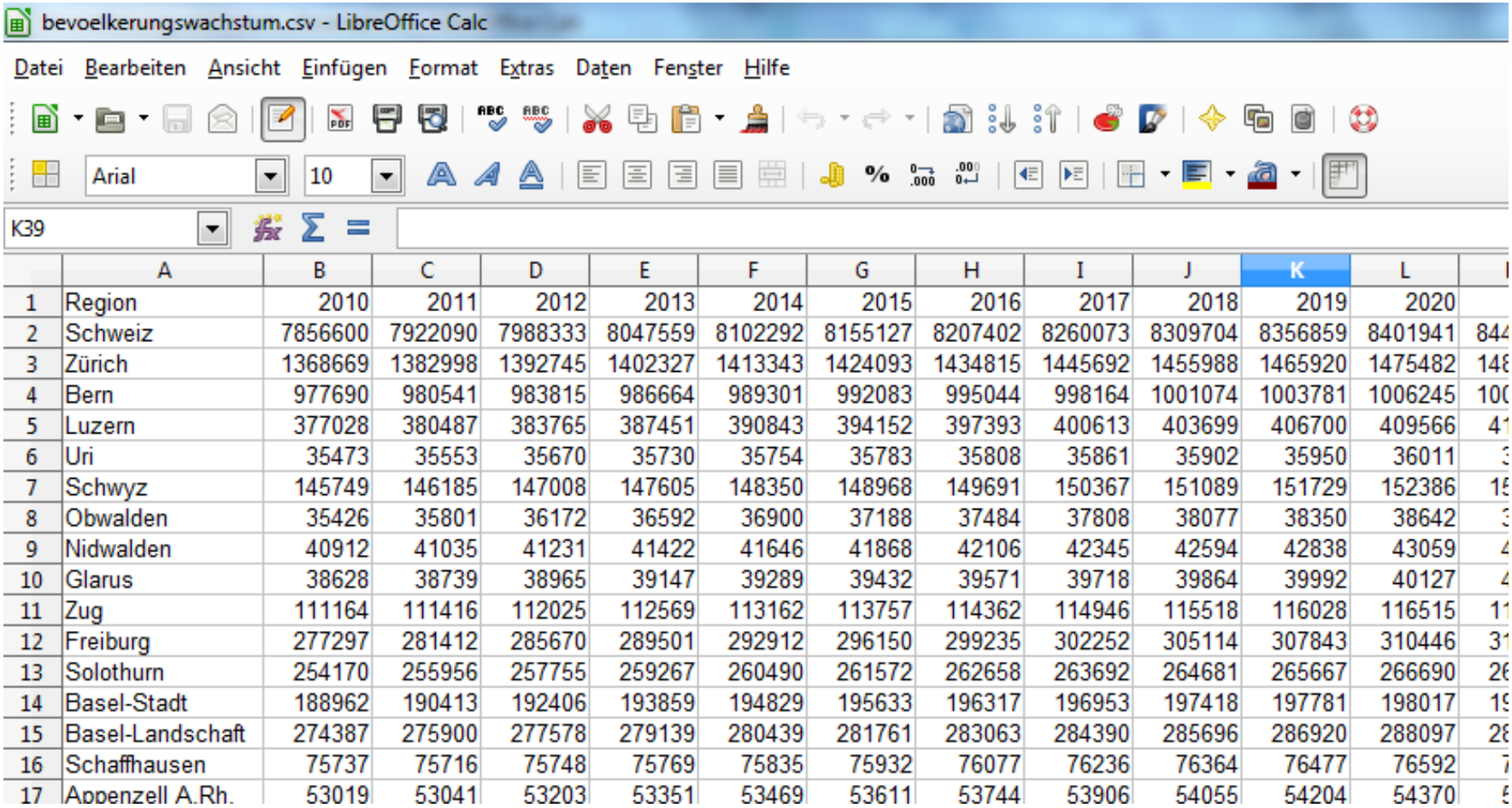
Link: http://bl.ocks.org/mbostock/1256572

Anpassen bestehender D3.js Snippets

Vorgehen:

1. Welche Daten werden visualisiert? -> Bevölkerungswachstum
2. Welche Visualisierungsart macht Sinn? -> D3 Show Real
3. **Wie muss ich die Daten anpassen? -> Transponieren**
4. Was muss ich am Code Snippet anpassen -> Pfade
5. Was läuft schief? -> Debugging

Daten in LibreOffice öffnen



bevoelkerungswachstum.csv - LibreOffice Calc

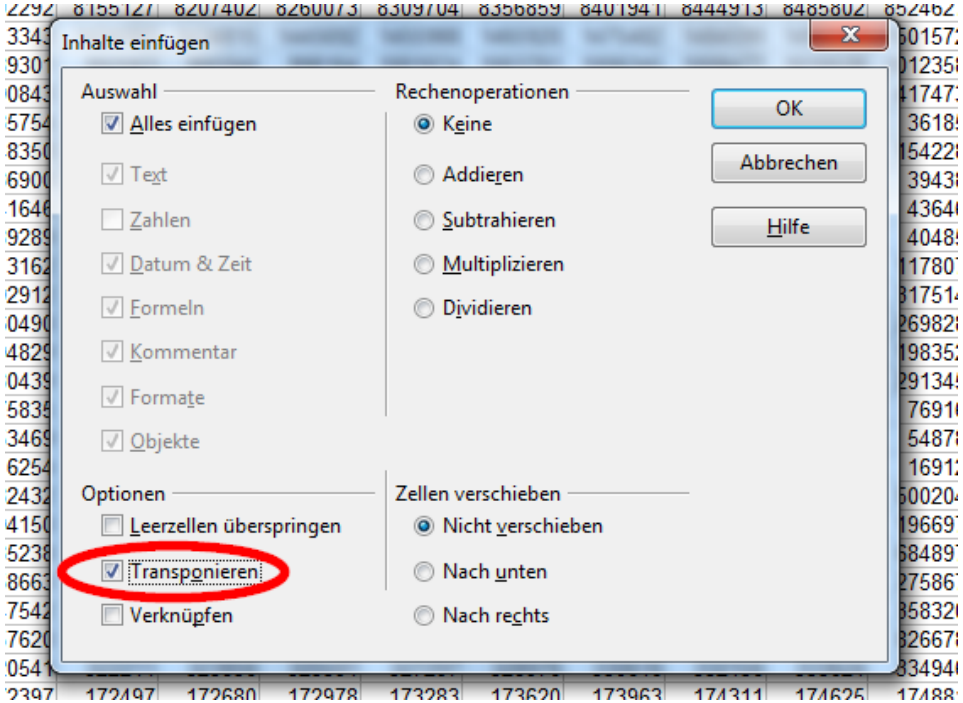
Datei Bearbeiten Ansicht Einfügen Format Extras Daten Fenster Hilfe

Arial 10

K39

	A	B	C	D	E	F	G	H	I	J	K	L
1	Region	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020
2	Schweiz	7856600	7922090	7988333	8047559	8102292	8155127	8207402	8260073	8309704	8356859	8401941
3	Zürich	1368669	1382998	1392745	1402327	1413343	1424093	1434815	1445692	1455988	1465920	1475482
4	Bern	977690	980541	983815	986664	989301	992083	995044	998164	1001074	1003781	1006245
5	Luzern	377028	380487	383765	387451	390843	394152	397393	400613	403699	406700	409566
6	Uri	35473	35553	35670	35730	35754	35783	35808	35861	35902	35950	36011
7	Schwyz	145749	146185	147008	147605	148350	148968	149691	150367	151089	151729	152386
8	Obwalden	35426	35801	36172	36592	36900	37188	37484	37808	38077	38350	38642
9	Nidwalden	40912	41035	41231	41422	41646	41868	42106	42345	42594	42838	43059
10	Glarus	38628	38739	38965	39147	39289	39432	39571	39718	39864	39992	40127
11	Zug	111164	111416	112025	112569	113162	113757	114362	114946	115518	116028	116515
12	Freiburg	277297	281412	285670	289501	292912	296150	299235	302252	305114	307843	310446
13	Solothurn	254170	255956	257755	259267	260490	261572	262658	263692	264681	265667	266690
14	Basel-Stadt	188962	190413	192406	193859	194829	195633	196317	196953	197418	197781	198017
15	Basel-Landschaft	274387	275900	277578	279139	280439	281761	283063	284390	285696	286920	288097
16	Schaffhausen	75737	75716	75748	75769	75835	75932	76077	76236	76364	76477	76592
17	Appenzell A.Rh.	53019	53041	53203	53351	53469	53611	53744	53906	54055	54204	54370

Daten kopieren und transponieren



bevoelkerungswachstum_showreal.csv -

Datei Bearbeiten Ansicht Einfügen Fo

Arial 10

F32

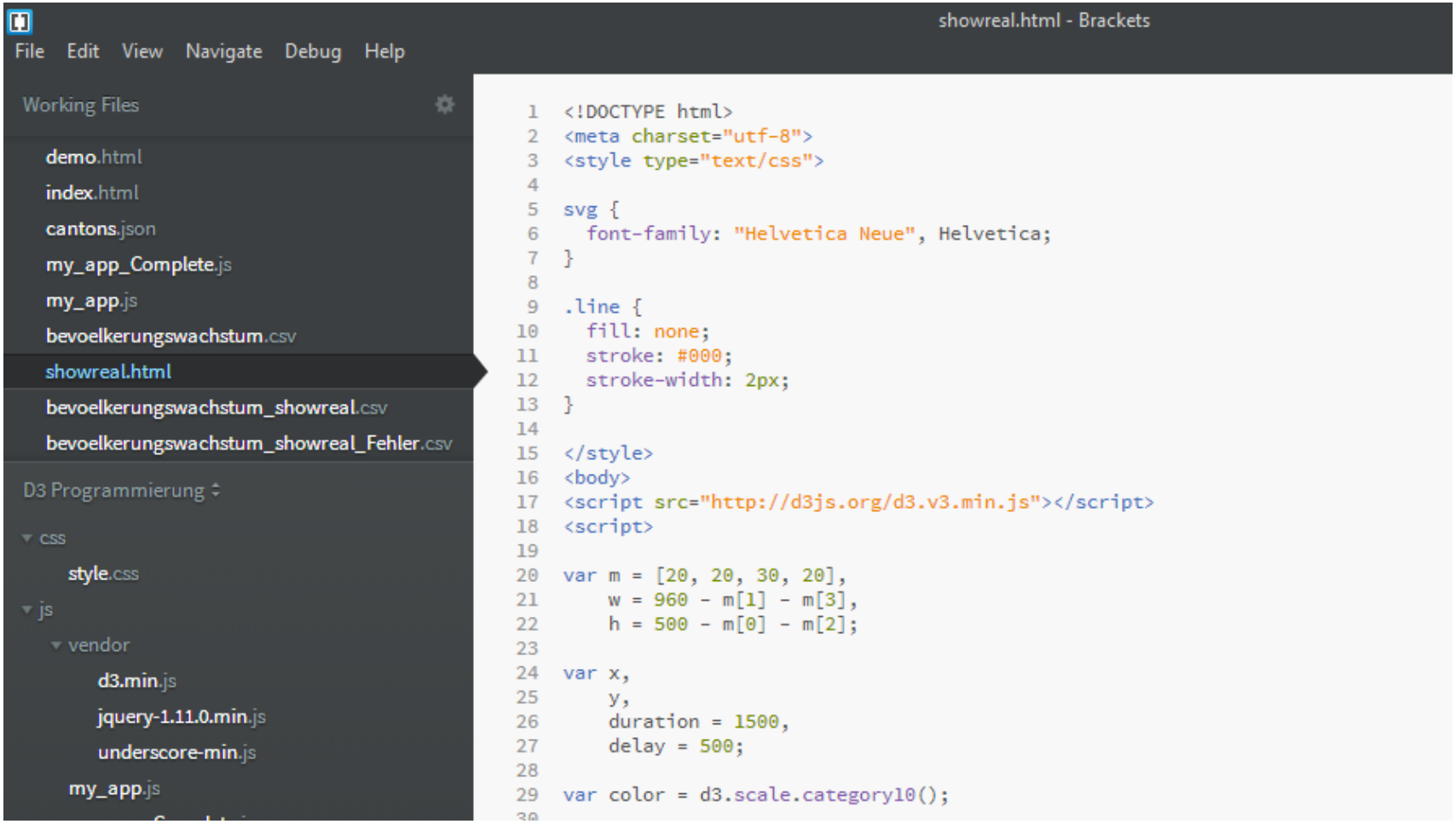
	A	B	C
1	Kanton	Jahr	Bevölkerung
2	Zürich	2010	1368669
3	Zürich	2011	1382998
4	Zürich	2012	1392745
5	Zürich	2013	1402327
6	Zürich	2014	1413343
7	Zürich	2015	1424093
8	Zürich	2016	1434815
9	Zürich	2017	1445692
10	Zürich	2018	1455988
11	Zürich	2019	1465920
12	Zürich	2020	1475482
13	Zürich	2021	1484599
14	Zürich	2022	1493302
15	Zürich	2023	1501572
16	Zürich	2024	1509404
17	Zürich	2025	1516809
18	Zürich	2026	1523809
19	Zürich	2027	1530507
20	Zürich	2028	1536801
21	Zürich	2029	1542776
22	Zürich	2030	1548413
23	Zürich	2031	1553805
24	Zürich	2032	1558908
25	Zürich	2033	1563769
26	Zürich	2034	1568397
27	Zürich	2035	1572841
28	Bern	2010	977690
29	Bern	2011	980541
30	Bern	2012	983815

Anpassen bestehender D3.js Snippets

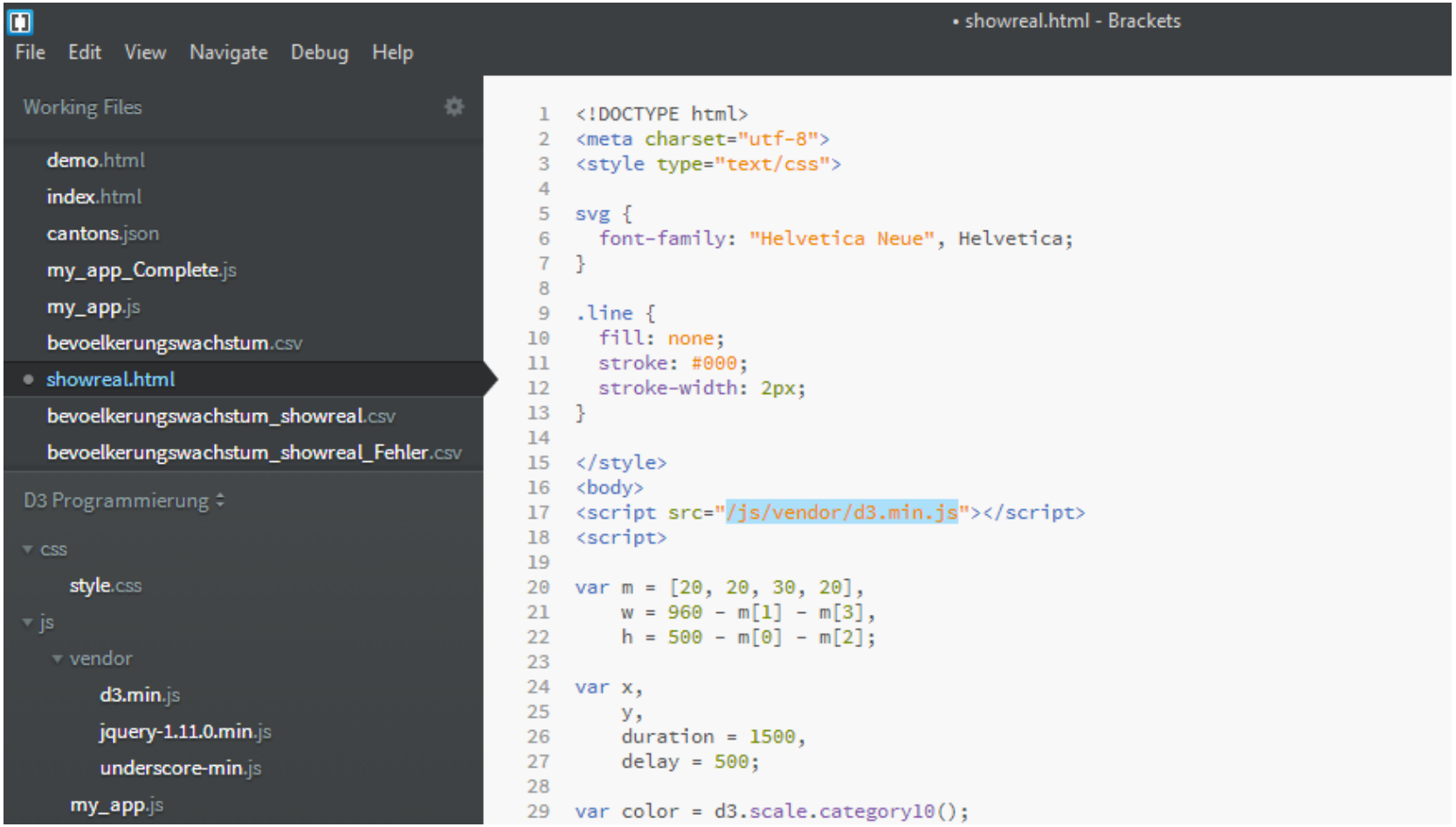
Vorgehen:

1. Welche Daten werden visualisiert? -> Bevölkerungswachstum
2. Welche Visualisierungsart macht Sinn? -> D3 Show Real
3. Wie muss ich die Daten anpassen? -> Transponieren
4. **Was muss ich am Code Snippet anpassen -> Pfade**
5. Was läuft schief? -> Debugging

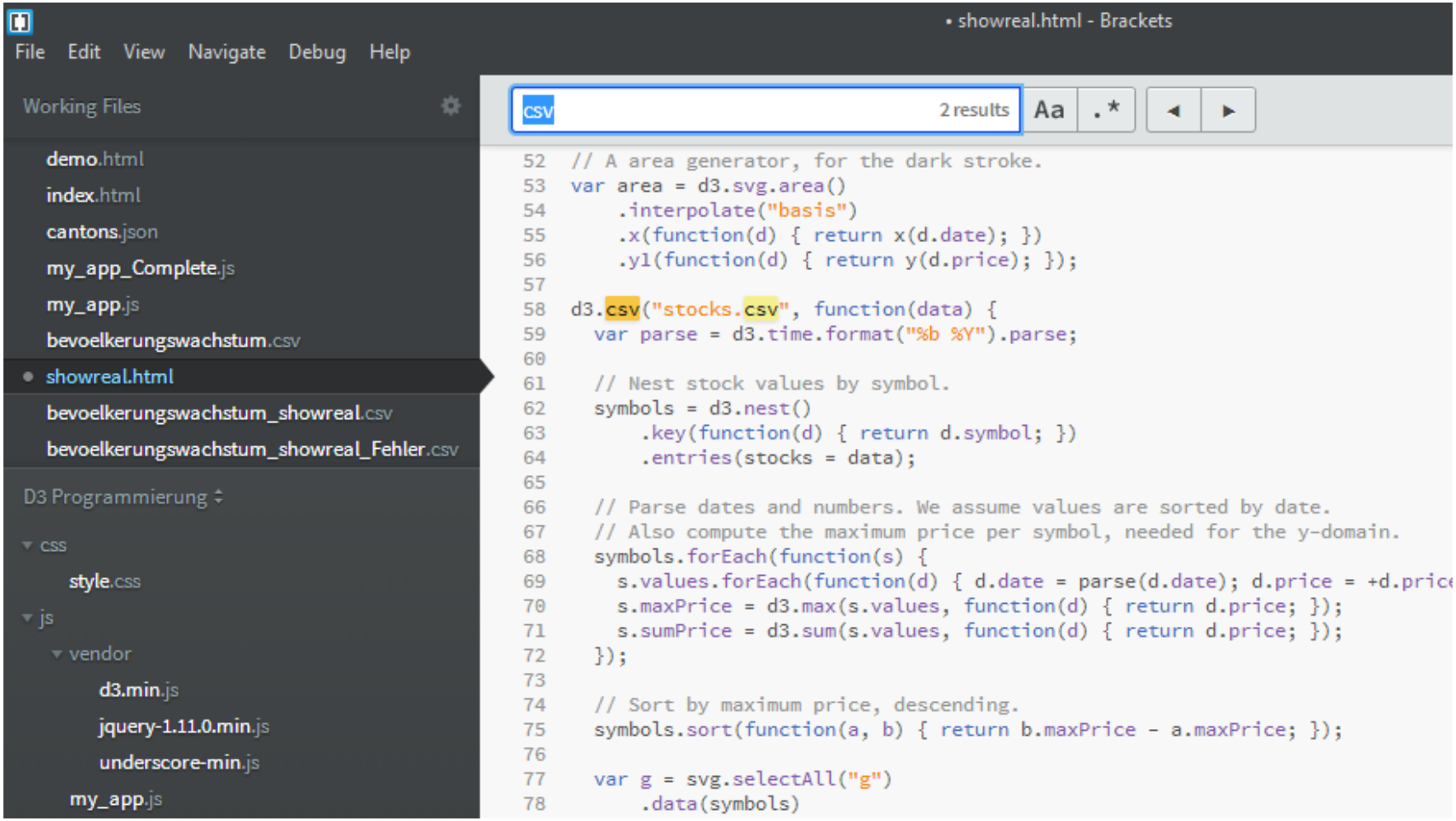
Code Snippet in Bracket kopieren



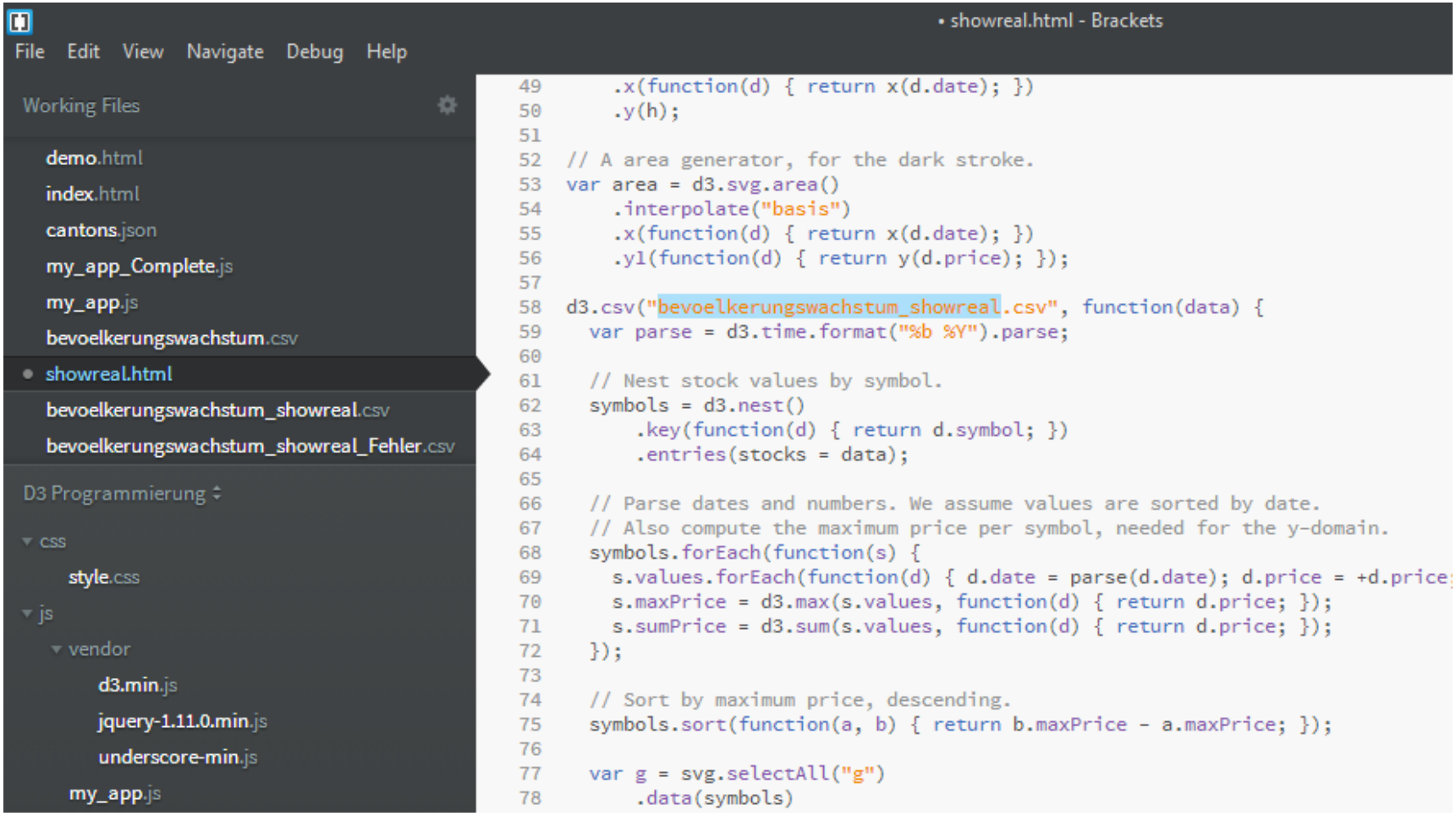
Pfad zu D3.js Bibliothek anpassen



Pfad zu CSV-Daten anpassen



Pfad zu CSV-Daten anpassen

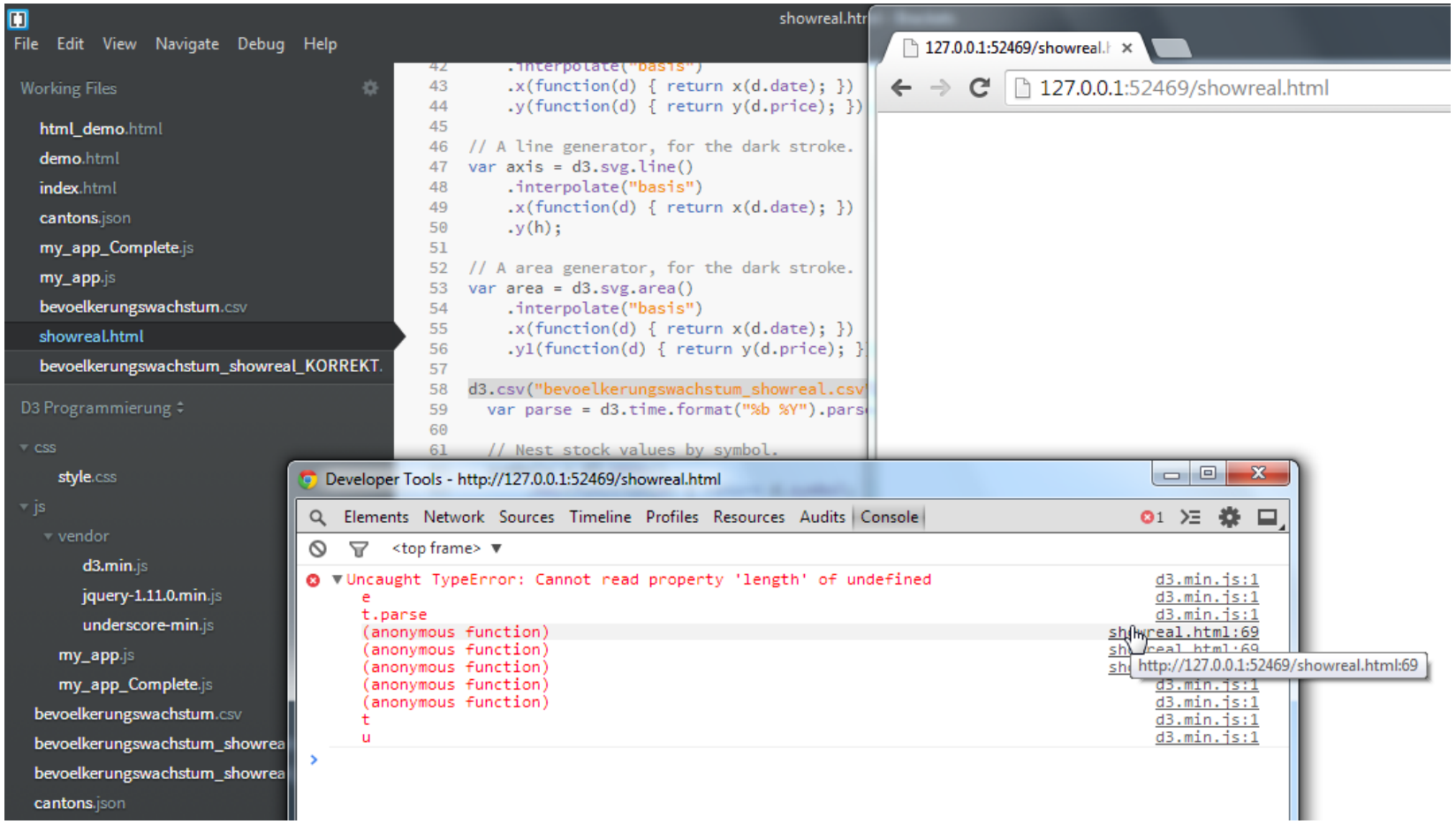


Anpassen bestehender D3.js Snippets

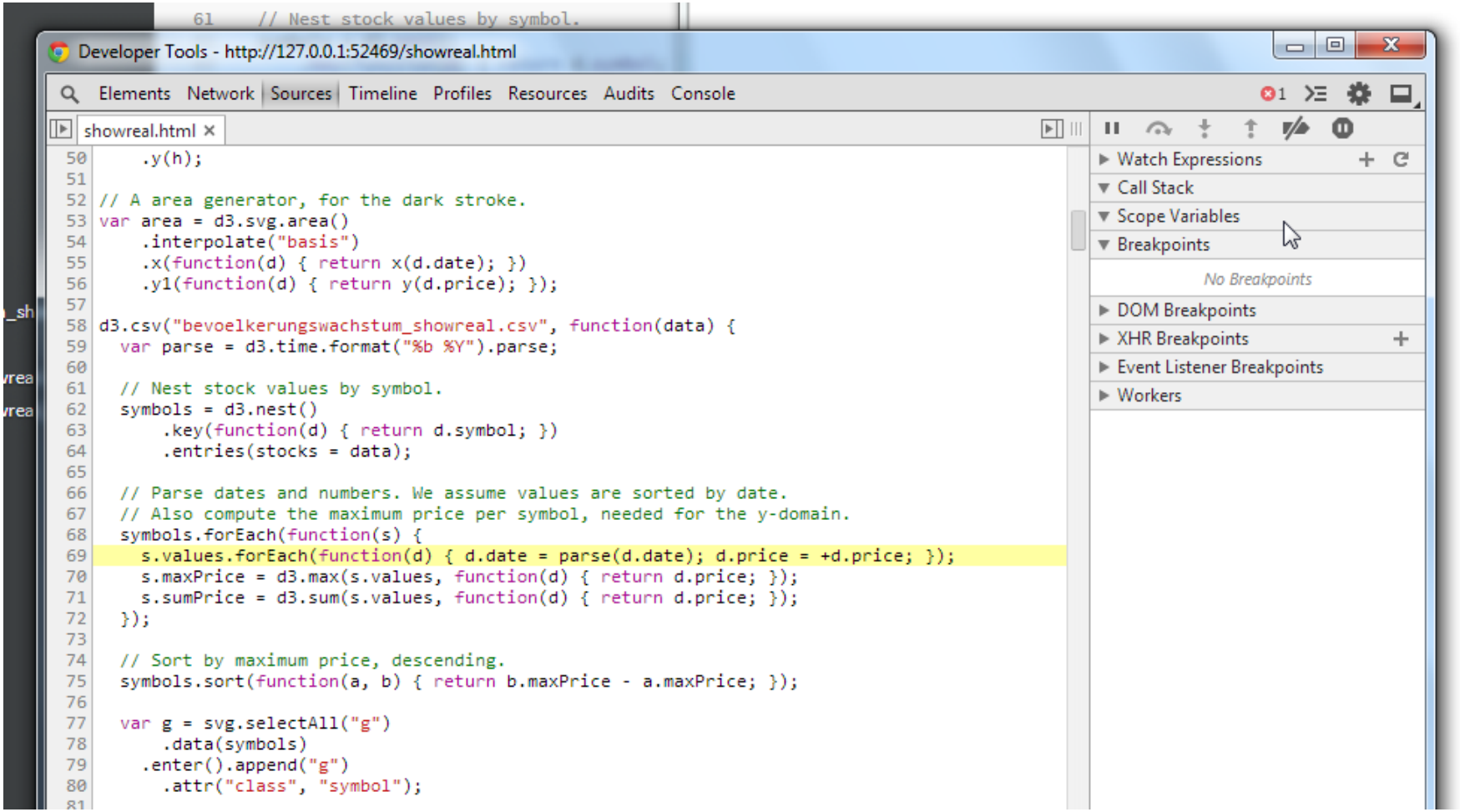
Vorgehen:

1. Welche Daten werden visualisiert? -> Bevölkerungswachstum
2. Welche Visualisierungsart macht Sinn? -> D3 Show Real
3. Wie muss ich die Daten anpassen? -> Transponieren
4. Was muss ich am Code Snippet anpassen -> Pfade
5. **Was läuft schief? -> Debugging**

A) Gar keine Darstellung -> Console

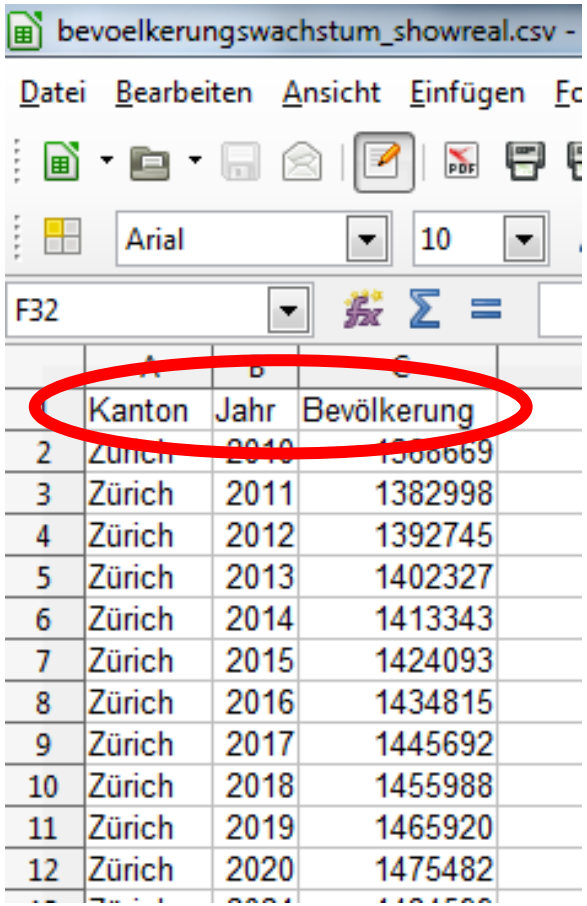


Fehler in der Beschriftung der Spalten

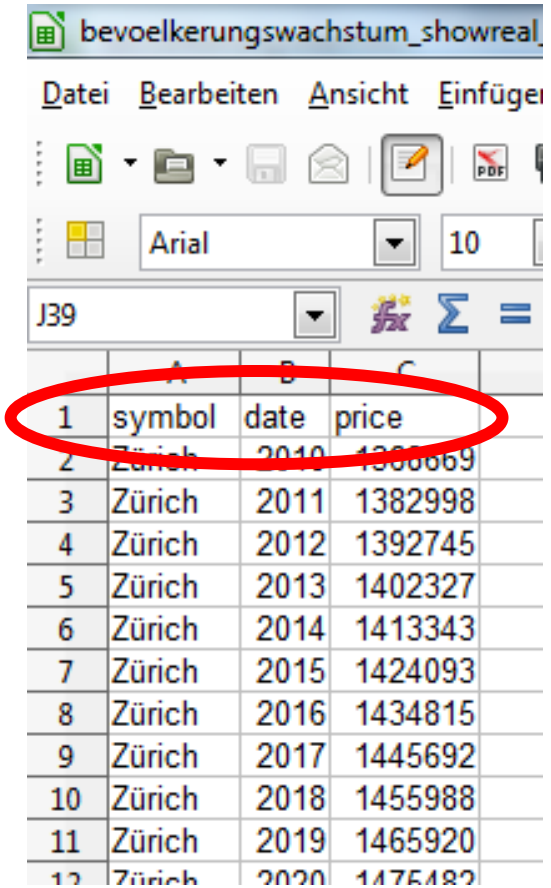



```
    .y(h);

// A area generator, for the dark stroke.
var area = d3.svg.area()
    .interpolate("basis")
    .x(function(d) { return x(d.date); })
    .y1(function(d) { return y(d.price); });

d3.csv("bevoelkerungswachstum_showreal.csv", function(data) {
  var parse = d3.time.format("%b %Y").parse;

  // Nest stock values by symbol.
  symbols = d3.nest()
      .key(function(d) { return d.symbol; })
      .entries(stocks = data);

  // Parse dates and numbers. We assume values are sorted by date.
  // Also compute the maximum price per symbol, needed for the y-domain.
  symbols.forEach(function(s) {
    s.values.forEach(function(d) { d.date = parse(d.date); d.price = +d.price; });
    s.maxPrice = d3.max(s.values, function(d) { return d.price; });
    s.sumPrice = d3.sum(s.values, function(d) { return d.price; });
  });

  // Sort by maximum price, descending.
  symbols.sort(function(a, b) { return b.maxPrice - a.maxPrice; });

  var g = svg.selectAll("g")
      .data(symbols)
    .enter().append("g")
      .attr("class", "symbol");
```

Fehler in der Beschriftung der Spalten

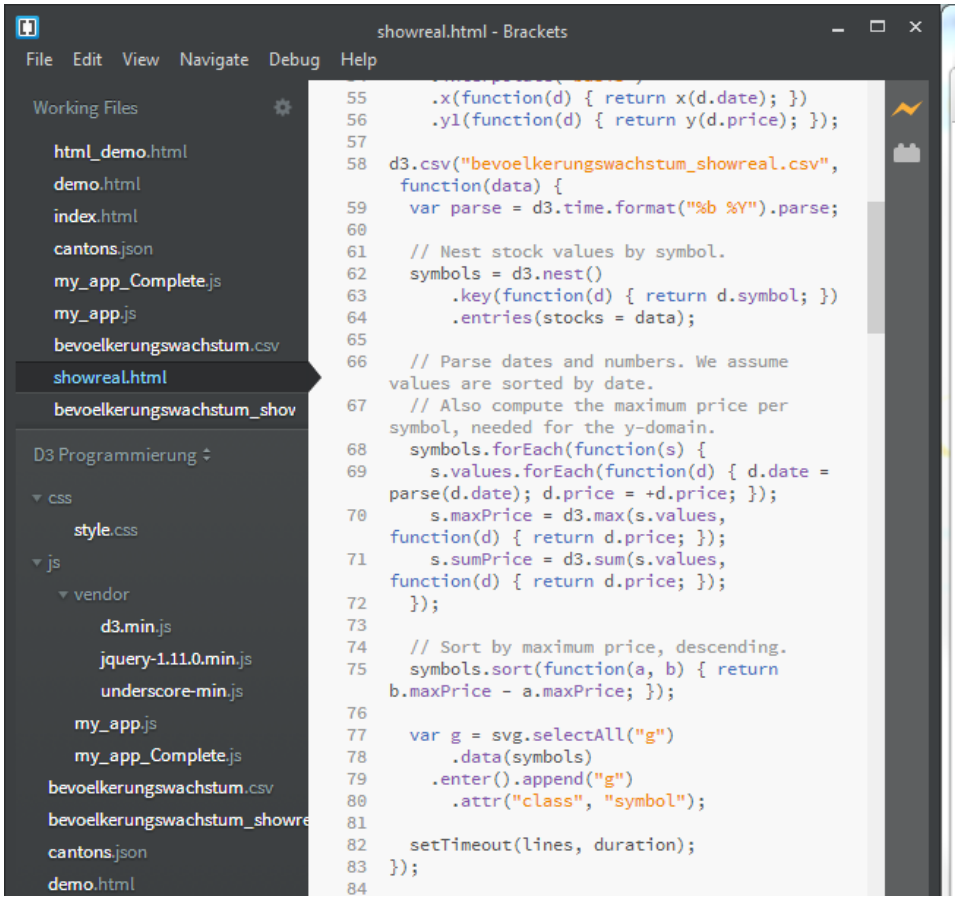


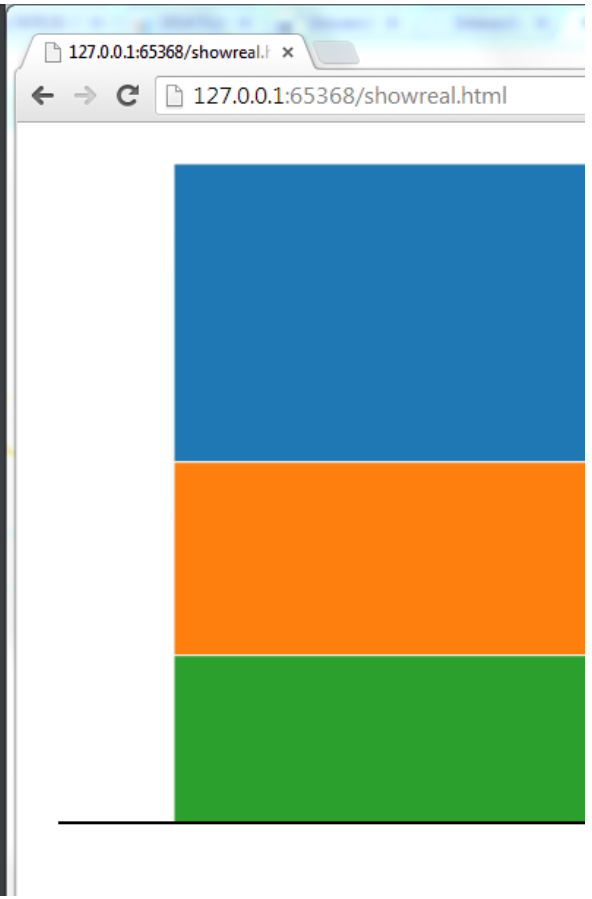
	Kanton	Jahr	Bevölkerung
2	Zürich	2010	1388669
3	Zürich	2011	1382998
4	Zürich	2012	1392745
5	Zürich	2013	1402327
6	Zürich	2014	1413343
7	Zürich	2015	1424093
8	Zürich	2016	1434815
9	Zürich	2017	1445692
10	Zürich	2018	1455988
11	Zürich	2019	1465920
12	Zürich	2020	1475482



	symbol	date	price
2	Zürich	2010	1388669
3	Zürich	2011	1382998
4	Zürich	2012	1392745
5	Zürich	2013	1402327
6	Zürich	2014	1413343
7	Zürich	2015	1424093
8	Zürich	2016	1434815
9	Zürich	2017	1445692
10	Zürich	2018	1455988
11	Zürich	2019	1465920
12	Zürich	2020	1475482

B) Fehlerhafte Darstellung

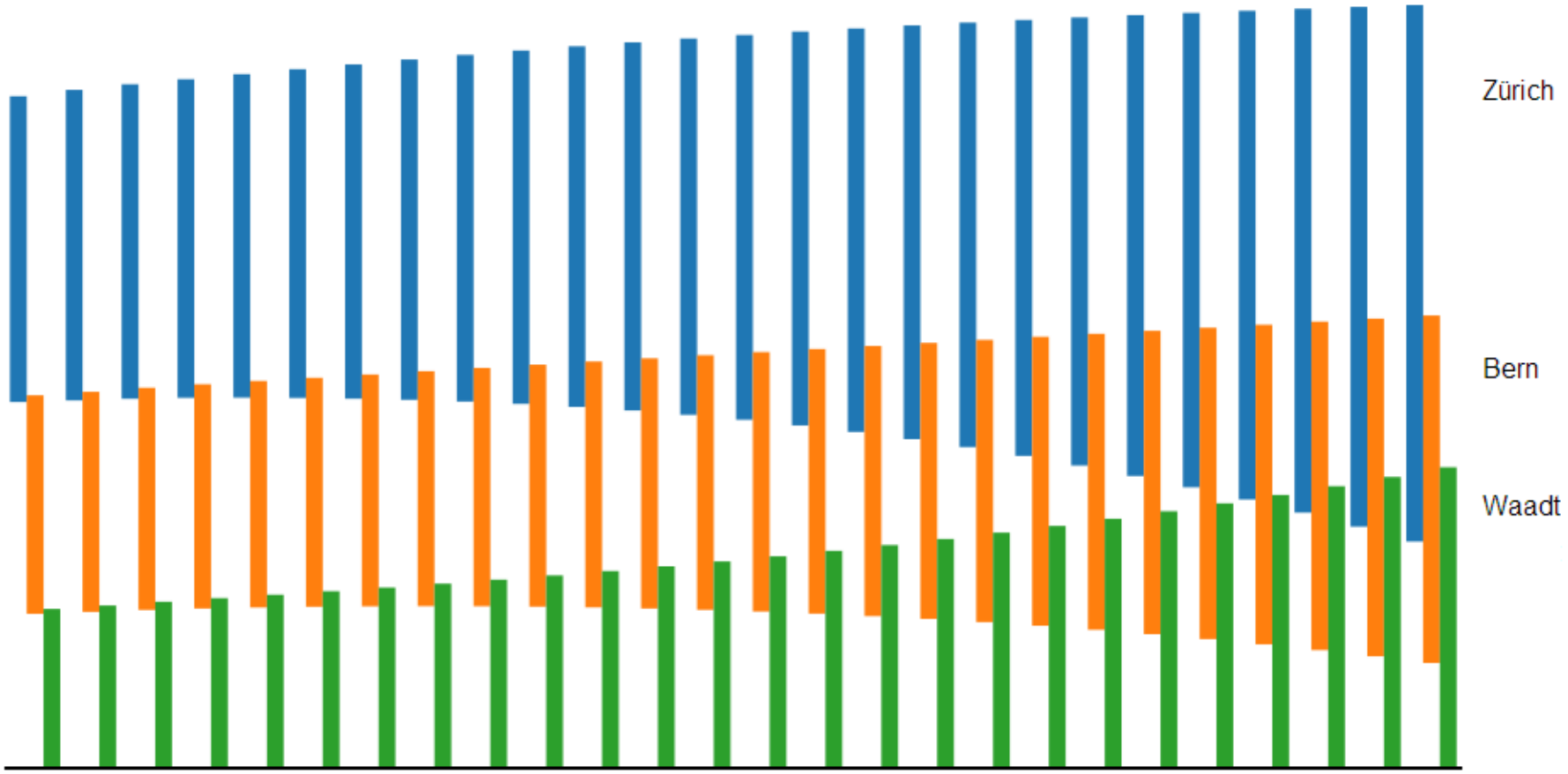




Falsches Datumsformat

```
56      .y1(function(d) { return y(d.price); });
57
58  d3.csv("bevoelkerungswachstum_absolut.csv", function(data) {
59    var parse = d3.time.format("%b %Y").parse;
60
61    // Nest stock values by symbol.
62    symbols = d3.nest()
63        .key(function(d) { return d.symbol; })
64        .entries(stocks = data);
65
66    // Parse dates and numbers. We assume values are sorted by date.
67    // Also compute the maximum price per symbol, needed for the y-domain.
68    symbols.forEach(function(s) {
```

Korrektes Datumsformat

```
56      .y1(function(d) { return y(d.price); });
57
58  d3.csv("bevoelkerungswachstum_shown_cal.csv", function(data) {
59    var parse = d3.time.format("%Y").parse;
60
61    // Nest stock values by symbol.
62    symbols = d3.nest()
63        .key(function(d) { return d.symbol; })
64        .entries(stocks = data);
65
```

Jetzt klappts!