

# Entwicklung von Diagrammen im Web

Max Mathys

MNG Rämibühl

Betreuende Lehrperson: David Sichert hat er einen Titel?

4. Januar 2016

*Maturitätsarbeit*

2016

dies würde ich streichen

## Abstract

In dieser Arbeit wird behandelt, wie Datensätze dargestellt werden können, sodass der Informationsertrag erhöht werden kann.

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>1</b>
1.1	Bedeutung und Zweck von Diagrammen . . . . .	1
1.2	Ziel dieser Arbeit . . . . .	4
1.3	Wahl des Diagrammtyps für die Software . . . . .	4
1.4	Wahl der Programmiersprache für die Software . . . . .	5
<b>2</b>	<b>Hauptteil</b>	<b>6</b>
2.1	Softwaretechnologie . . . . .	6
2.1.1	JavaScript . . . . .	6
2.1.2	Hypertext Markup Language (HTML), Document Object Model (DOM) . . . . .	7
2.1.3	Scalable Vector Graphics (SVG) . . . . .	7
2.1.4	Cascading Style Sheets (CSS) . . . . .	7
2.2	Verarbeitung der Daten . . . . .	7
2.2.1	Rohdaten . . . . .	8
2.2.1.1	Comma-separated values (CSV) . . . . .	8
2.2.2	Filtering . . . . .	8
2.2.2.1	Konvertierung in JavaScript Objects . . . . .	9
2.2.2.2	Formatierung . . . . .	9
2.2.2.3	Datensatzspezifische Filteringkonfiguration und Hard Coding . . . . .	11
2.2.3	Aufbereitete Daten . . . . .	12
2.2.4	Mapping . . . . .	12
2.2.5	Geometriedaten . . . . .	12
2.2.6	Rendering . . . . .	12
2.2.7	Bilddaten . . . . .	12
<b>3</b>	<b>Schlusswort</b>	<b>13</b>

A Literaturverzeichnis	14
B Abbildungsverzeichnis	15
C Bestätigung der Eigenständigkeit	16

# Kapitel 1

## Einleitung

### 1.1 Bedeutung und Zweck von Diagrammen

Die graphische Darstellung von Daten, das Diagramm, ist von grosser Bedeutung für die Gesellschaft. Man findet Diagramme überall: In etlichen Wissenschaften, wo sie nicht wegzudenken sind, in jeglichen Industrien, in Zeitungen, in Werbungen.

Das Ziel eines Diagramms ist die visuelle Repräsentation einer gegebenen Datenmenge, um damit eine effektive Auswertung zu ermöglichen. Die Analyse, das Verständnis und die Kommunikation sollen erleichtert werden [7]. Der Leser sollte sich mit der Datenmenge auseinandersetzen können, ohne die Rohdaten selbst betrachten zu müssen. Je nachdem wie das Diagramm graphisch umgesetzt wird, können verschiedene Zusammenhänge und Informationen des Datensatzes hervorgehoben oder in den Hintergrund gestellt werden, was einen Einfluss auf die Vermittlung hat. Das Ziel jedoch eines Diagramms ist es, die darzustellende Datenmenge möglichst unverfälscht darzustellen [6].

Im Informationszeitalter sind Daten von grosser Bedeutung, der Datenfluss vergrössert sich ständig. Um Daten darstellen zu können, müssen sie zuerst gesammelt, sortiert und formatiert werden, bevor mit der Auswertung begonnen werden kann. Das Sammeln von Daten stellt oftmals keine besondere Schwierigkeit dar, das Auswerten, Darstellen und Interpretieren ist eine Herausforderung.

Konventionell werden Diagramme in Zeitungen, Artikeln abgedruckt. Der Autor trifft Entscheidungen, wie die Daten in graphischer Form dargestellt werden und erstellt auf Basis dieser ein Diagramm. Nach dem Druck kann es vom Leser betrachtet werden, jedoch kann dieser die Darstellung nicht mehr verändern, das Diagramm ist statisch. Er hat darum keinen Einfluss, wie die Daten in diesem **statischen Diagramm** dargestellt und an ihn vermittelt werden.

Durch das Aufkommen von modernen Computern, Smartphones haben sich die Möglichkeiten zur Darstellung erweitert: Dem Nutzer ist es möglich, mit dem Diagramm zu interagieren. Man findet unter anderem in Online-Zeitschriften solche **dy-namischen Diagramme**, welche Daten darstellen, die mit dem Artikel zu tun haben und interaktiv vom Nutzer bedient werden können. Die Abbildungen 1.1 und 1.2 demonstrieren, wie das Potential der Interaktion auf eine interessante Weise ausgenutzt werden kann.

In der Abbildung 1.1 ist ein Blasendiagramm dargestellt. Jede Blase stellt eine Firma dar, je nach Börsenwert ist die Blase unterschiedlich gross. Die Firmen sind in horizontaler Richtung nach ihrem Steuertarif geordnet, die verschiedenen Farben stellen die jeweilige Industrie, in der die Firma tätig ist, dar. Wenn die Maus über die Blase fährt, wird ein *Popup* sichtbar, das weitere Informationen zu der Firma anzeigt wie der exakte Steuertarif, die absolute Menge an Steuern, die bezahlt wurden oder der Umsatz der Firma.

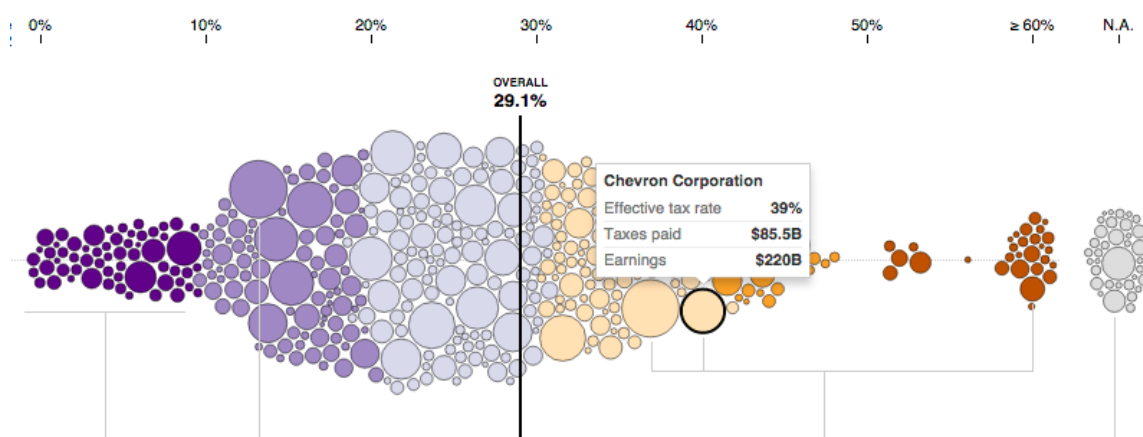


Abbildung 1.1: Ein Blasendiagramm, das Steuerabgaben und Steuersätze von US-Firmen darstellt. [4]

Die Webseite in Abbildung 1.2 berechnet anhand von vom Benutzer festgelegten Faktoren, ob der Kauf eines Hauses profitabler wäre als die Miete. Es sind Faktoren vorhanden wie der Zinssatz, die Entwicklung der Miete und der Wert des Hauses, der Steuersatz für Grundstücke, die Inflationsrate. Die verschiedenen Faktoren kann der Benutzer durch einen *Slider* anpassen. Dabei kann er durch die Höhe der Balken über dem Slider sehen, wie sich der Faktor auf das Resultat auswirkt (ob der Kauf oder die Miete profitabler wäre). Beim verschieben des Sliders wird das Ergebnis automatisch aktualisiert, so sieht der Nutzer den Zusammenhang der Faktoren und der Berechnung.

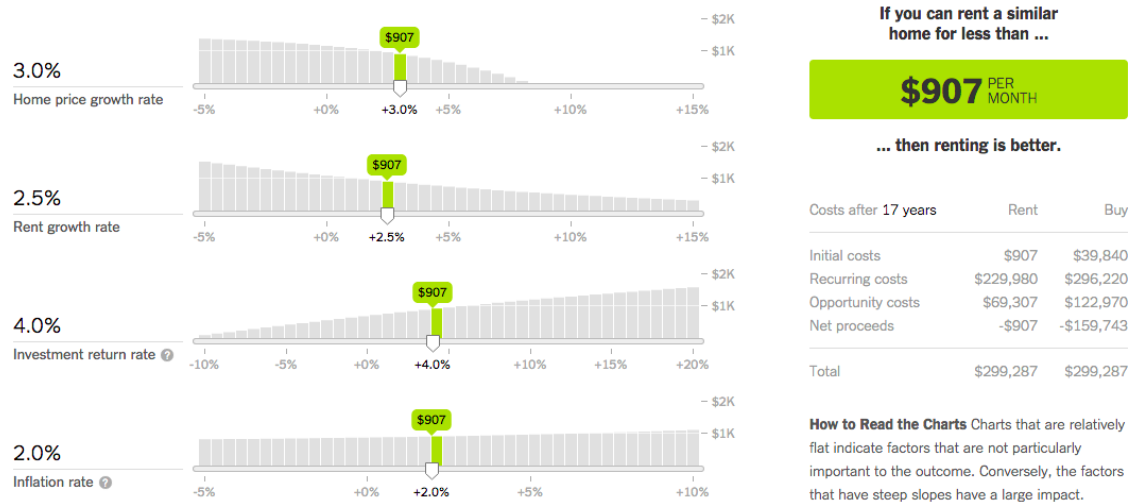


Abbildung 1.2: Interaktives Diagramm mit verstellbaren Reglern, das die Profitabilität des Kaufs eines Hauses darstellt. [2]

Diese Maturaarbeit wurde von dem Artikel *The Eyes Have It: A Task by Data Type for Information Visualizations* von B. Shneidermann [8] inspiriert. Der Artikel untersucht, wie ein Benutzer eine Programmoberfläche wahrnimmt und auf welche Weise er mit ihr interagiert. Auch beschreibt der Artikel, wie die Oberfläche aufgebaut sein sollte, damit der Benutzer sich im Programm orientieren kann, die Programmlogik versteht und es ohne grossen Aufwand bedienen kann. In dem Artikel stellte der zudem das *Mantra der Informationsvisualisierung (Information-Seeking Mantra)* auf, das beschreibt, wie der Benutzer typischerweise mit einer Oberfläche interagiert: Überblick zuerst, Zoomen und Filtern, als letztes Details auf Abruf [8].

An den Diagrammen in 1.1 und 1.2 kann man den Vorteil von dynamischen Diagrammen gut demonstrieren: In 1.1 wird der dritte Teil der Mantra der Informationsvisualisierung angewendet, Details auf Abruf: Der Tooltip zusätzliche Informationen einer angewählten Firma an. Wegen Platzmangel wäre es nicht möglich gewesen, die gesamte Information aller Firmen in einem statischen Diagramm darzustellen.

In 1.2 verändern sich die einzelnen Diagramme je nach Einstellung der Faktoren. Diese Verhaltensweise ist nicht umzusetzen in einem statischen Diagramm.

Die von Shneidermann beschriebenen Prinzipien für das Design von Benutzeroberflächen werden in dieser Arbeit berücksichtigt. Es werden interaktive, dynamische Diagramme entwickelt, die den Informationsertrag des Nutzers verbessern sollten. Er sollte sich mit den Daten effizienter auseinandersetzen können, besser verstehen und Spass an der Erkundung des Datensatzes haben [8].

## 1.2 Ziel dieser Arbeit

Das Ziel dieser Arbeit ist die eigene Entwicklung einer Software, die dynamische Diagramme in einem Web-Browser darstellt. Sie sollen den Prinzipien von Shneidermann folgen, und dem Nutzer erlauben, eine bessere Analyse, ein besseres Verständnis über des Datensatzes zu erlangen. Auch soll es dem Nutzer Spass machen und ihn ermutigen, sich mit dem Diagramm auseinanderzusetzen.

Ich will auch die „Taktik“ zeigen, mit der diese Diagramme erstellt werden, welche Probleme bei der Entwicklung aufgetreten sind und verwendete Entwicklungswerkzeuge vorstellen.

## 1.3 Wahl des Diagrammtyps für die Software

Es gibt sehr viele Arten von Diagrammen, welche sich unterschiedlich für verschiedene Datentypen eignen. Diese Applikation beschränkt sich auf zwei grundlegende Diagrammtypen: Das *Punktediagramm* und das *Liniendiagramm*.

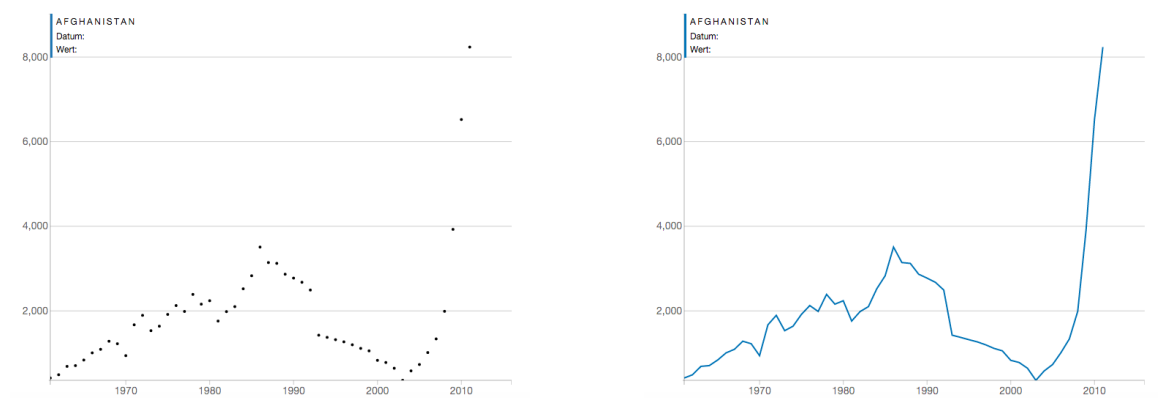


Abbildung 1.3: Beispiel von Diagrammen am Datensatz des CO<sub>2</sub>-Verbrauchs von Afghanistan. Links: Punktediagramm. Rechts: Liniendiagramm mit linearer Interpolation.

Das Punktediagramm und das Liniendiagramm werden in der Wissenschaft als auch in anderen Medien oft verwendet. In diesen Diagrammen werden Skalen in der Ebene oder Raum festgelegt, die ein Koordinatensystem beschreiben. Mindestens eine Achse dient als Skala der unabhängigen Variablen (der Dimension, die den Beobachtungsraum der Daten aufspannen). Die restlichen Achsen dienen als Skala der abhängigen Variablen, Ausprägungen, die abgetragen werden und mit graphischen Symbolen, wie etwa Kreise, Kreuze oder Vierecke markiert werden [7]. Beim Liniendiagramm werden zusätzlich die benachbarten Punkte durch eine Linien oder Kur-



venabschnitt verbunden, was Trends und Strukturen der Daten deutlicher darstellt, und Datenpunkte, die durch Linien verbunden sind, werden zusätzlich gruppiert [9]. In der Abbildung 1.3 sind ein Punktediagramm und ein Liniendiagramm dargestellt, die den Verlauf den CO<sub>2</sub>-Verbrauchs darstellen.

Ich habe das Diagramm in der Abbildung 1.3 dargestellt. Wort "ich" hier und an anderen Stellen (suchen) vermeiden, ersetzen durch eine Passivkonstruktion. me, Benutzer werden. Zudem können solche Diagramme oft nur einen spezifischen Datentyp darstellen. Benutzer sind sich oft an Punkte- und Liniendiagramme gewöhnt.

## 1.4 Wahl der Programmiersprache für die Software

Interaktive Diagramme werden in der Praxis fast ausschliesslich für den Web-Browser entwickelt, wie auch in dieser Arbeit. Technologien wie HTML, CSS, SVG, JavaScript werden verwendet, diese ermöglichen die dynamische Manipulation durch den Nutzer. Diese Technologien werden durch praktisch alle neueren Smartphones, Tablets, Laptops, Desktops unterstützt.

Rechtschreibung

# Kapitel 2

## Hauptteil

Nachstehend wird beschrieben, wie die Applikation entwickelt wurde.

### 2.1 Softwaretechnologie

Für die Entwicklung einer Applikation für den Web-Browser werden verschiedene Technologien benötigt, die jeweils für einen einzelnen Aspekt der Applikation verantwortlich sind.

#### 2.1.1 JavaScript

JavaScript ist die Programmiersprache, die der Web-Browser unterstützt. Die Programmlogik wird in dieser Sprache geschrieben.

Viele Programmierer tendieren dazu, den JavaScript Code nach persönlicher Weise und auf nicht konsequente Weise zu formatieren, was die Lesbarkeit des Programmcodes, für den Autor als auch besonders für andere Betrachter, beeinträchtigt. Dies kann sich vor allem in umfangreichen Projekten, wo mehrere hundert Zeilen Code vorhanden sind, markant auf die Übersichtlichkeit auswirken. Da Open-Source von mehr und mehr praktiziert wird, gewinnen Syntaxkonventionen an Bedeutung. „Das Anwenden der [Syntax-] Konvention bedeutet, die Syntaxkonventionen der Community und den Belang der Lesbarkeit des Programmcodes über die persönliche Programmierweise zu stellen“ [1]. Für bessere Programmqualität und Lesbarkeit wird deshalb in dieser Arbeit der *JavaScript Standard Style* gebraucht, eine verbreitete Syntaxkonvention.

Diese Software verwendet die Programmierbibliothek *Data-Driven-Documents (D3)*. Sie wurde von Michael Bostock, Vadim Ogievetsky und Jeffrey Heer erstellt und dient zur Entwicklung von Visualisationen im Web. Sie erleichtert die Benutzung des

Document Object Models (DOM) (vgl. **Abbildung <todo>**), ermöglicht effizienteres Debugging und verbessert die Leistung („*Performance*“) der Applikation [3].

### 2.1.2 Hypertext Markup Language (HTML), Document Object Model (DOM)

Hypertext Markup Language (HTML) ist in der Entwicklung einer Web-Applikation für den Inhalt (Text, Links, Bilder, Buttons...) der Seite zuständig. Die Sprache beschreibt durch *Elemente*, die einen Wert haben können, verschachtelt sein können und denen *Attribute* zugewiesen werden können die darzustellenden Informationen.

Das Document Object Model (DOM) ermöglicht die dynamische Manipulation dieser Elemente durch Schnittstellen in JavaScript-Programmen.

### 2.1.3 Scalable Vector Graphics (SVG)

Scalable Vector Graphics (SVG) ist ein Format für Vektorgrafiken. **hier in diesem Abschnitt ist nichts kursiv.** Vektorgrafiken stehen nicht wie andere Bildformate (JPG, PNG) aus Pixeln, sondern aus Elementen wie Kreisen, Ellipsen, Rechtecke, Linien. SVG-Grafiken können im Browser dargestellt werden und durch JavaScript ebenfalls dynamisch manipuliert werden.

### 2.1.4 Cascading Style Sheets (CSS)

Cascading Style Sheets (CSS) beschreiben die Darstellung der anzuzeigenden Elemente, die in HTML-Dokumenten oder SVG-Grafiken vorkommen. CSS-Attribute können in HTML- oder SVG-Elementen im Attribut „class“ durch *Klassen* zugewiesen werden oder direkt im Attribut „style“ definiert werden.

## 2.2 Verarbeitung der Daten

Bevor das Diagramm im Browser dargestellt werden kann, müssen zuerst die Daten **geladen verarbeitet werden**. Die Beispiele dieser Arbeit nutzen **grösstenteils** **ausschliesslich** öffentliche Daten, die frei verfügbar sind. Die Diagramme in Abbildung 1.3 benutzen einen Datensatz der Weltbank, der den  $CO_2$ -Verbrauch von Afghanistan beinhaltet.

Beim Prozess zur Veranschaulichung von Daten wird die **Visualisierungspipeline** durchlaufen [5]. Diese Pipeline stellt drei wesentliche Schritte des Prozesses dar: Die Datenaufbereitung (*Filtering*), die Erzeugung des Geometriemodells (*Mapping*) und die Bildgenerierung (*Rendering*).

**Zitat nach unten, also hierhin, verschieben**

**source**

## 2.2.1 Rohdaten

### 2.2.1.1 Comma-seperated values (CSV)

Das Comma-seperated values (CSV) Format *Comma-seperated values (CSV)*. CSV stellt eine Tabelle dar. Zeilen werden im Format durch einen Umbruch und Spaltenwerte durch ein Komma getrennt. Optional steht in der ersten Zeile (Abbildung 2.1 links, Zeile 1) der Datei die Beschriftung der Spalten.

In Abbildung 2.1 wird die Funktion des CSV-Formats demonstriert.

Date,Value  
2010-12-31,4220.717  
2009-12-31,4352.729  
2008-12-31,5555.505  
2007-12-31,5067.794

Date	Value
2010-21-31	4220.717
2009-21-31	4352.729
2008-21-31	5555.505
2007-21-31	5067.794

Abbildung 2.1: Demonstration des CSV-Formats. Links: Die CSV-Datei in Rohtext. Rechts: Darstellung der Informationen der CSV-Datei in einer Tabelle.

Man verwendet das CSV-Format oft, weil es sehr einfach aufgebaut ist. Das Lesen von solchen Tabellenformaten ist ohne grossen Aufwand in Programmen umsetzbar.

Zudem beschränkt sich das CSV-Format nur auf die Vermittlung der Daten, und beinhaltet keine Informationen zu der Darstellung der Tabelle. Excel-Dateien (XLS/XLSX) hingegen speichern auch Daten zu der Formatierung, auch Zeilengrösse, Textgrösse, Textformat und viele mehr. Für die Zwecke dieser Software sind diese Informationen irrelevant.

Wenn die Software die CSV-Datei laden und verarbeiten will, dann braucht sie zunächst zusätzliche Informationen zu der Datei: Die *URL*, die den Ort der Datei angibt.

entweder aus dem web oder lokal

Die Software lädt die Daten durch eine *Ajax-Anfrage (Asynchronous JavaScript and XML)* herunter, bevor sie sie weiter verarbeitet.

## 2.2.2 Filtering

Das Filtering ist der Prozess, der Rohdaten in aufbereitete Daten umwandelt. Die Aufgaben des Filtering sind zum Beispiel die Vervollständigung, Reduzierung, Korrektur der Daten, sodass sie in den folgenden Schritten der Visualisationspipeline verwendet werden können [7, Kap. 2].

### 2.2.2.1 Konvertierung in JavaScript Objects



Als erster Schritt wird in der Applikation die CSV-Datei in *JavaScript-Objekte* umgewandelt. Das Laden und *Parsing* der Datei von CSV zu Objekten ist von D3 implementiert.

JavaScript Objekte werden im *JavaScript Object Notation*-Format (*JSON*) dargestellt.

Im Abbildung 2.2 ist der Prozess der Umwandlung ersichtlich: Es wird ein *Array* von allen Zeilen in der Tabelle (ausgenommen der ersten Zeile, wo die Spalten beschriftet werden) erstellt. Jede Zeile wird als Objekt mit den dazugehörigen Spalten dargestellt.

Date	Value
2010-12-31	4220.717
2009-12-31	4352.729
2008-12-31	5555.505
2007-12-31	5067.794

```
[
  {
    "Date": "2010-12-31",
    "Value": "4220.717"
  },
  {
    "Date": "2009-12-31",
    "Value": "4352.729"
  },
  {
    "Date": "2008-12-31",
    "Value": "5555.505"
  },
  {
    "Date": "2007-12-31",
    "Value": "5067.794"
  }
]
```

Abbildung 2.2: Konvertierung von CSV zu JavaScript Objekten. Links: CSV. Rechts: JSON.

### 2.2.2.2 Formatierung

Die Software benötigt nun Anweisungen, um den Datensatz (im JSON-Format) zu formatieren, damit er im Diagramm verwendet werden kann. Der Prozess muss folgende Aufgaben erledigen:

- (Zeichenstring-) Elemente gegebenenfalls in JavaScript-Objekte umwandeln



- Falls mehrere Datensätze vorhanden sind, diese *mergen*, also in einen einzigen Array zusammenfassen
- Den gesamten gemergten Datensatz nach der unabhängigen Variable aufsteigend sortieren

**Umwandlung zu JavaScript-Objekten.** Das CSV-Format unterscheidet nicht zwischen Datentypen. Alle Werte in CSV-Dateien sind Zeichenstrings.

In dem Beispiel in der Abbildung 2.2 rechts, Zeile 3 wird für das erste Objekt im Array das Attribut mit dem Namen „Date“ definiert. Der Datentyp ist hier ein Zeichenstring. Die Umwandlung in das JavaScript-Date-Objekt, das ein Datum darstellt, ist sinnvoll: Das JavaScript-Objekt beherrscht viele Funktionen, wie zum Beispiel die Ausgabe der Anzahl Millisekunden, die seit dem 1. Januar 1970 vergangen sind. Dies ist beim Vergleichen von verschiedenen Date-Objekten nützlich. Das Date-Objekt ist zum Beispiel auch fähig, das Datum in einem Format auszugeben, das den lokalen Konventionen entspricht: 28.10.2015 (Schweiz), 10/28/2015 (USA).

Nummern, wie in Abbildung 2.2 rechts, Zeile 4, im Attribut mit dem Namen „Value“ definiert, wandeln wir in Nummern um. Nur an Nummer-Objekten können Rechenoperationen durchgeführt werden.

**Merging von Datensätzen.** Oft werden mehrere Datensätze in der Software geladen. Ein Beispiel ist der CO<sub>2</sub>-Verbrauch von Ländern: Für jedes Land wird eine separate CSV-Datei geladen und in ein JavaScript-Array umgewandelt.

Ich entschieden, alle geladenen Datensätze (Arrays) in ein Datensatz (Array) zusammenzufassen (*merge*), weil dies für die Programmlogik mehr Sinn macht.

Da Spalten von verschiedenen Datensätzen meist mit gleichem Namen beschriftet sind, könnte man nach dem Merge die Spalten nicht unterscheiden (Abbildung 2.3 oben). Darum wird die Beschriftung aller Spalten der abhängigen Variablen durch eine eindeutige ID ersetzt (Abbildung 2.3 unten). Die eindeutige ID wird durch den Spaltennamen und die URL der Datensatzes generiert: Der Spaltennamen wird ein Rautenzeichen und die URL angehängt. Dies ermöglicht, dass man trotz Merge die Objekte dem herkömmlichen Datensatz zuordnen kann.

In der Abbildung 2.3 wurden zwei Datensätze mit dem Dateinamen ch-co2.csv und af-co2.csv gemergt. Im unteren Beispiel werden die Spaltennamen der abhängigen Variablen durch die eindeutige ID ersetzt.


**Sortieren vom gemergten Datensatz.** Der Datensatz wird nach der unabhängigen Variable ansteigend sortiert, damit die Berechnung von Interpolationen ermöglicht werden.


<pre>[   {     "Date": "2010-12-31",     "Value": "4220.717"   },   {     "Date": "2009-12-31",     "Value": "4352.729"   },   {     "Date": "2010-12-31",     "Value": "1320.717"   },   {     "Date": "2009-12-31",     "Value": "7353.129"   } ]</pre>	<pre>[   {     "Date": "2010-12-31",     "Value#ch-co2.csv": "4220.717"   },   {     "Date": "2009-12-31",     "Value#ch-co2.csv": "4352.729"   },   {     "Date": "2010-12-31",     "Value#af-co2.csv": "1320.717"   },   {     "Date": "2009-12-31",     "Value#af-co2.csv": "7353.129"   } ]</pre>
---	---


Abbildung 2.3: Demonstration der Merge-Strategie und Anwendung der ID-Generierung. Oben: Gemergter Datensatz, ohne eindeutige IDs. Unten: Gemergter Datensatz, mit eindeutigen IDs.

### 2.2.2.3 Datensatzspezifische Filteringkonfiguration und Hard Coding

Die Implementation dieser Anweisungen zum Filtering (Spezifikation der URLs, abhängige und unabhängige Variablen, Datentypen der Spalten des Datensatzes) kann auf zwei Arten durchgeführt werden. <in bezug auf datensatz formulieren>

- Die Anweisungen zum Filtering sind hardgecoded 
- Die Anweisungen zum Filtering sind in einer *Konfiguration* abgelegt (softcoding)

**Hard Coding.** Die Anweisungen (Konfiguration) zum Filtering sind direkt im Programmcode abgelegt. Falls Datensätze mit anderen URLs, Spaltennamen oder Datentypen in der Applikation verwendet werden wollen, so muss der Programmcode entsprechend angepasst werden. 

**Soft Coding.** Beim Soft Coding sind die Anweisungen (Konfiguration) für das Filtering in einer externen Ressource abgelegt. Diese Applikation liest die Konfiguration..... 

**2.2.3    Aufbereitete Daten**

**2.2.4    Mapping**

**2.2.5    Geometriedaten**

**2.2.6    Rendering**

**2.2.7    Bilddaten**



# Kapitel 3

## Schlusswort

<future work weitere ideen!> <mehr dimensionen durch farben, grösse von punk-  
ten.....>

# Anhang A

## Literaturverzeichnis

- [1] Feross Aboukhadijeh u. a. *JavaScript Standard Style. One Style to Rule Them All*. 3. Okt. 2015. URL: <https://github.com/feross/standard> (besucht am 03.10.2015).
- [2] Michael Bostock, Shan Carter und Archie Tse. *Is It Better to Rent or Buy?* New York Times. 17. Juli 2015. URL: <http://www.nytimes.com/interactive/2014/upshot/buy-rent-calculator.html> (besucht am 04.10.2015).
- [3] Michael Bostock, Vadim Ogievetsky und Jeffrey Heer. “D3: Data-Driven Documents”. In: *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)* (2011). URL: <http://vis.stanford.edu/papers/d3>.
- [4] Michael Bostock u. a. *Across U.S. Companies, Tax Rates Vary Greatly*. New York Times. 25. März 2015. URL: <http://www.nytimes.com/interactive/2013/05/25/sunday-review/corporate-taxes.html> (besucht am 04.10.2015).
- [5] Robert Haber und David McNabb. *Visualization idioms: A conceptual model for scientific Visualization in Scientific Computing*. Los Alamitos: IEEE-Computer Society Press, 1990.
- [6] Volker Jung. *Integrierte Benutzerunterstützung für die Visualisierung in Geo-Informationssystemen*. Stuttgart: Fraunhofer IRB Verlag, 1998. ISBN: 3-8167-5218-7, 978-3-8167-5218-9.
- [7] Heidrun Schumann und Wolfgang Müller. *Visualisierung. Grundlagen und allgemeine Methoden*. 1. Aufl. Springer-Verlag Berlin Heidelberg, 2000. ISBN: 978-3-540-64944-1. DOI: 10.1007/978-3-642-57193-0.
- [8] Ben Shneiderman. “The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations”. In: (1996). ISSN: 1049-2615.
- [9] Jens Wandmacher. *Software-Ergonomie*. Berlin: Gruyter, 1. März 1993. ISBN: 311012971X, 9783110129717.

# Anhang B

## Abbildungsverzeichnis

1.1	Blasendiagramm in The New York Times . . . . .	2
1.2	Interaktives Diagramm in The New York Times . . . . .	3
1.3	Vergleich zwischen Punktediagramm und Liniendiagramm . . . . .	4
2.1	Demonstration des CSV-Formats . . . . .	8
2.2	CSV und JSON . . . . .	9
2.3	Merge-Strategie . . . . .	11

## Anhang C

### Bestätigung der Eigenständigkeit

Der/die Unterzeichnete bestätigt mit Unterschrift, dass die Arbeit selbständig verfasst und in schriftliche Form gebracht worden ist, dass die Mitwirkung anderer Personen auf Beratung und Korrekturlesen beschränkt hat und dass alle verwendeten Unterlagen und Gewährspersonen aufgeführt sind.