

# CAS Data Visualization

Autorenprojekt  
Flight Fare Visualization



# Einleitung

## CAS Data | Visualization

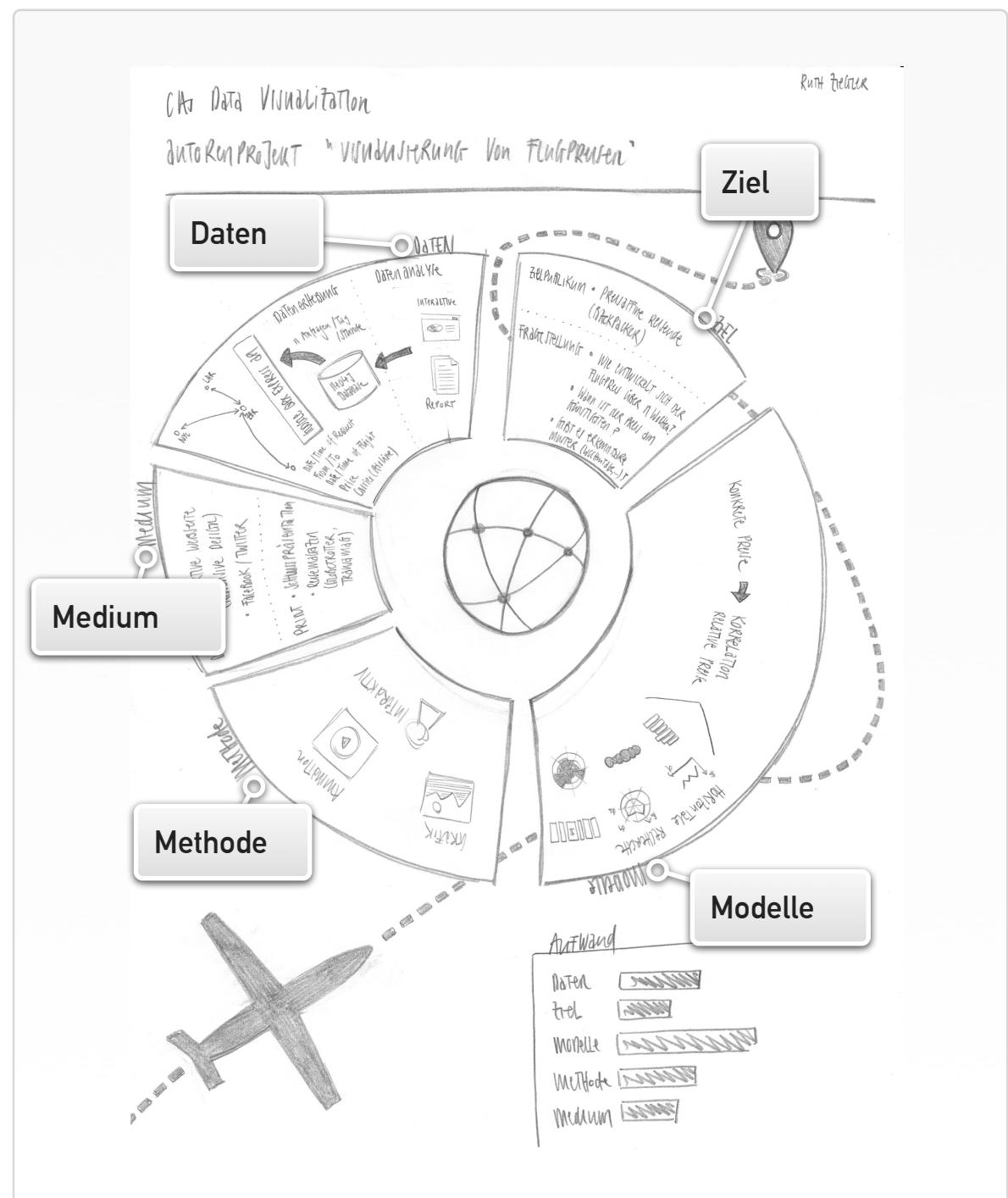
*Durchschnittlich gibt es täglich rund 93'000 Flüge, welche von ungefähr 9'000 Flughäfen weltweit starten. Zu jeder Zeit befinden sich rund um den Globus zwischen 8'000 und 13'000 Flugzeuge gleichzeitig in der Luft.* [www](#)

Jeder dieser Flüge besitzt einen Flugpreis, welcher sich über die Zeit verändert. Diese Veränderung der Flugpreise zu visualisieren ist das Ziel von Flight Fare Visualization (ffv). Dem interessierten Betrachter sollen diverse Fragestellungen mit Hilfe der Visualisierung beantwortet werden. Als primäre Zielgruppe sind preisaffine Reisende angesprochen.

Die vorliegende Prozess-Dokumentation beschreibt das Autorenprojekt "Flight Fare Visualization", welches im CAS Data Visualization an der Hochschule für Künste HKB erarbeitet wurde.

Das Autorenprojekt wurde begleitend zum CAS Data Visualization im Sommersemester 2016 durchgeführt.

# Übersicht



# Flight Fare Visualization

## Fragestellung

Die Fragestellung des Projektes ist neben dem Zielpublikum der zentrale Bestandteil. Diese legt einen Fokus auf die Visualisierung und ermöglicht dem Betrachter der Visualisierung die Fragen mit Hilfe dieser zu beantworten.

Im Laufe des Projektes musste ich meine Fragestellung immer weiter schärfen. Es hat sich gezeigt, dass die Fragestellung möglichst präzise formuliert sein muss, damit eine gezielte Visualisierungform gefunden werden kann.

Im Folgenden sind die unterschiedlichen Fragestellungen in der Chronologie aufgeführt, Version V2.1 sind die aktuellen Fragen, welche mit der Visualisierung beantwortet sind.

V1.0	<ul style="list-style-type: none"><li>- Wie entwickelt sich der Flugpreis über n Wochen?</li><li>- Wann ist der Preis am günstigsten?</li><li>- Gibt es erkennbare Muster (Wochentage, ...)?</li></ul>
V1.1	<ul style="list-style-type: none"><li>- Wie entwickelt sich der Flugpreis über n Wochen?</li><li>- Wann ist der Preis am günstigsten?</li><li>- Welche Unterschiede gibt es zwischen den Destinationen?</li></ul>
V2.0	<ul style="list-style-type: none"><li>- An welchem Wochentag soll ich buchen?</li><li>- Wie viele Wochen vor dem Abflug soll ich buchen?</li><li>- An welchem Wochentag soll ich abfliegen?</li><li>- Wie entwickelt sich der Flugpreis über n Tage vor Abflug?</li><li>- Wie viel spare ich beim richtigen Agent? Ist es günstiger direkt bei der Airline zu buchen als bei einem Agenten?</li><li>- Spielt die Tageszeit beim Buchen eine Rolle?</li><li>- Gibt es eine allgemeine Regel oder hängt es von der Destination ab? Sind Europaflüge und Oversea-Flüge gleich?</li><li>- Wie viel Geld kann ich durchschnittlich sparen, wenn ich am richtigen Zeitpunkt buche?</li></ul>
V2.1	<ul style="list-style-type: none"><li><b>- An welchem Wochentag soll ich buchen?</b></li><li><b>- Wie viele Wochen vor dem Abflug soll ich buchen?</b></li><li><b>- An welchem Wochentag soll ich abfliegen?</b></li><li><b>- Wie entwickelt sich der Flugpreis über n Tage vor Abflug?</b></li><li><b>- Wie viel Geld kann ich durchschnittlich sparen, wenn ich zum richtigen Zeitpunkt buche?</b></li></ul>

# Title of text

extra title



**Daten**

# Daten

## Datenerhebung

Im Rahmen der Datenerhebung habe ich mich damit beschäftigt, die Flugpreise abzufragen und für die weitere Verwendung im Projekt zu persistieren.

Ein Flugpreis wird für einen Flug angeboten. Ein Flugpreis wird nach unterschiedlichen Kriterien festgelegt. Jeder Flug besitzt einen Abflugsort (Origin) und einen Zielort (Destination). Der Flug kann entweder einfach (one-way journey) oder return (return journey) sein. Es gibt Direktflüge (single segment) oder Flüge mit Zwischenstopps (multi-segment). Die Klasse (Economy, Business oder First) ist ein weiteres Kriterium eines Fluges, welches sich auf den Preis auswirkt. Die Anzahl Personen ist ebenfalls ausschlaggebend für den Preis. Zudem ist die Fluggesellschaft (Carrier) ebenfalls eine Variable für den Flugpreis. Am Ende wird ein Flug mit den genannten Variablen von verschiedenen Anbietern (Agent) zu einem bestimmten Preis angeboten.

Um die riesige Menge von Daten überschaubarer zu machen und so die Komplexität der Visualisierung von Beginn weg etwas einzuschränken, habe ich mich entschieden die Daten nach folgenden Kriterien zu erheben:

1. **Origin Zürich**  
Es werden nur Preise zu Flügen mit Abflug-Flughafen Zürich erhoben.
2. **One-way journey**  
Es werden nur Preise zu einfachen Flügen (Hinflug) gemessen.

3. **Single person**  
Der Flugpreis wird immer für eine Person angefragt.
4. **Economy class**  
Es werden nur Preise für die Economy-Klasse erhoben.
5. **Single-segment flights**  
Es werden nur Direktflüge berücksichtigt.

Unter diesen Einschränkungen ergeben sich folgende mögliche Variablen für die Visualisierung:

1. **Destination**  
Es werden zwanzig unterschiedliche Zielflughäfen abgefragt.
2. **Zeitpunkt des Abfluges**  
Der Zeitpunkt des Abfluges lässt sich nach Bedarf in folgende Variablen weiter unterteilen
  - a. Datum
  - b. Tageszeit
  - c. Wochentag
  - d. Kalenderwoche
3. **Zeitpunkt der Preisanfrage**  
Auch der Zeitpunkt der Preisanfrage (request date) lässt sich nach Bedarf in dieselben Variablen wie beim Zeitpunkt des Abflugs unterteilen.
4. **Carrier**  
Alle verfügbaren Fluggesellschaften pro Destination werden erhoben.
5. **Agent**  
Alle verfügbaren Anbieter werden erhoben.

### **Ermittlung der Destinationen**

Es sollen zwanzig Destinationen ermittelt werden - zehn Destinationen innerhalb Europa und weitere zehn Destinationen ausserhalb Europas (Overseas).

Die Destinationen werden nach folgenden Regeln ausgewählt:

1. Es wird die Liste der geschäftigsten Flughäfen von Europa bzw. Weltweit als Grundlage genommen 

2. Pro Land darf nur ein Flughafen vorkommen
3. Das Land darf keine direkte Grenze zur Schweiz besitzen
4. Es muss mindestens ein Direktflug angeboten werden
5. Es werden die Top 5 anhand der Liste von 1 genommen
6. Es werden die Flop 5 anhand der Liste von 1 genommen
7. Die Regeln werden solange wiederholt, bis je 10 Destinationen ausgewählt sind

### **Europa**

Rang	Flughafen, Stadt (Land)	ITAC Code	Bemerkung
1	London Heathrow, London (GB)	LHR	x
2	Charles de Gaule, Paris (FR)	CDG	Direkte Grenze
3	Istanbul Ataturk, Istanbul (TR)	IST	x
4	Frankfurt, Frankfurt (DE)		Direkte Grenze
5	Amsterdam Shipol, Amsterdam (NL)	AMS	x
6	Adolfo Suarez Madrid, Madrid (ES)	MAD	x
7	Munich, München (DE)		Direkte Grenze
8	Leonardo da Vinci, Rome (IT)		Direkte Grenze
9	London Gatwick, London (GB)		Land vorhanden
10	Barcelona, Barcelona (ES)		Land vorhanden
11	Sherementyev, Moskau (RU)	SVO	x
...			

Rang	Flughafen, Stadt (Land)	ITAC Code	Bemerkung
7	Munich, München (DE)		Direkte Grenze
8	Leonardo da Vinci, Rome (IT)		Direkte Grenze
9	London Gatwick, London (GB)		Land vorhanden
10	Barcelona, Barcelona (ES)		Land vorhanden
11	Sherementyev, Moskau (RU)	SVO	x
...			
80	Riga International, Riga (LTU)	RIX	x
85	Keflavik, Reykjavik (IS)	KEF	x
87	Belgrade Nikola Tesla, Belgrade (SRB)	BEG	x
88	Malta, Malta (MT)	MLA	x
89	Rhodes, Griechenland (GR)	RHO	x
96	Trondheim, Norwegen (N)	TRD	Kein Direktflug

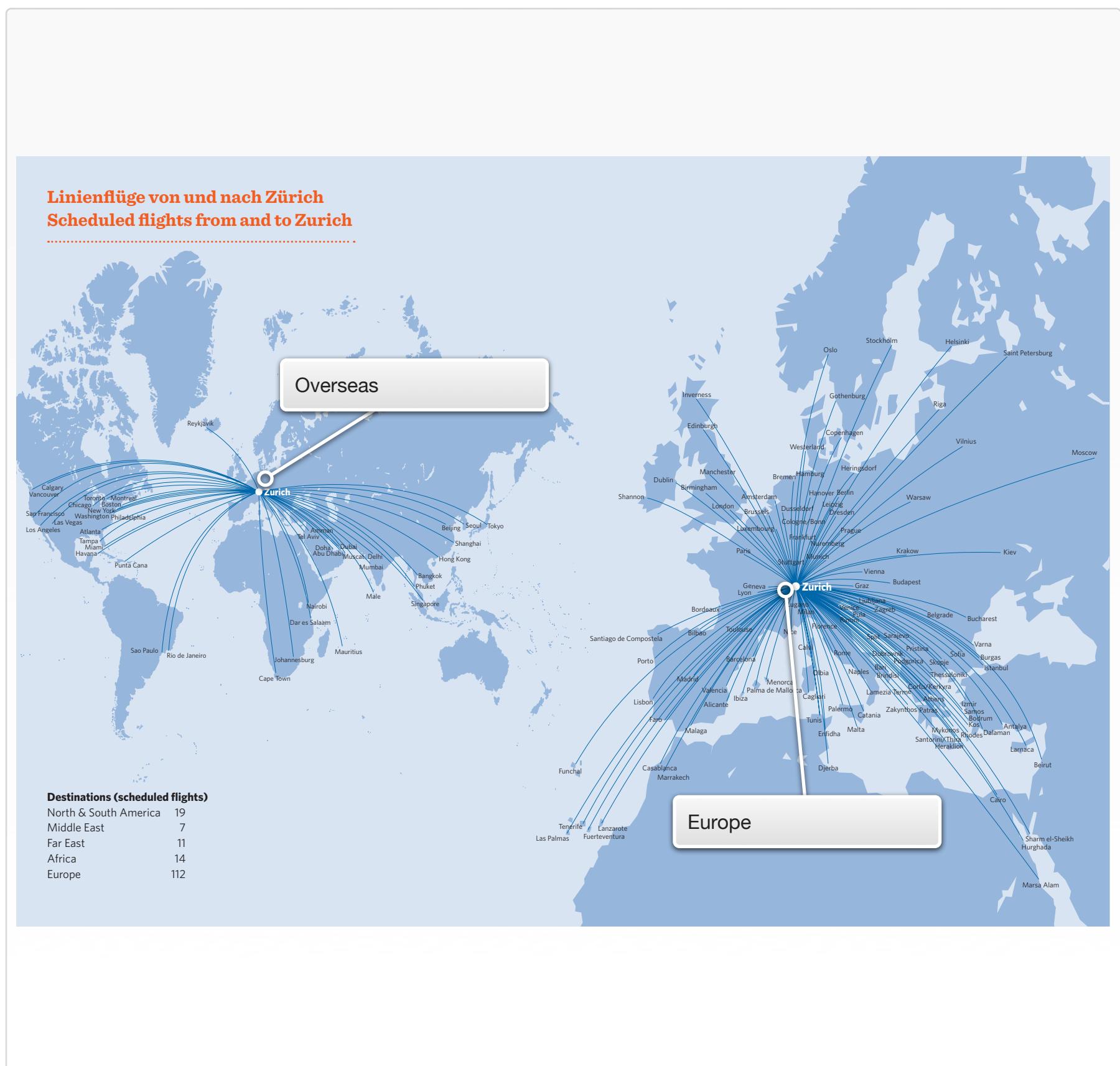
### Weltweit

Rang	Flughafen, Stadt (Land)	ITAC Code	Bemerkung
1	Hartsfield-Jackson Atlanta, Atlanta (US) John F. Kennedy, New York (US)	ATL JFK	x
2	Beijing Capital, Peking (CN)	PEK	x

Rang	Flughafen, Stadt (Land)	ITAC Code	Bemerkung
3	Dubai International, Dubai (AE)	DXB	x
4	Tokyo Haneda, Tokyo (JP)	NRT	x
5	Hong Kong, Hong Kong (CN)		Land vorhanden
...			
16	Singapore Changi, Singapur (SG)	SIN	x
...			
20	Suvarabhum, Bangkok (TH)	BKK	x
22	Seoul Incheon, Seoul (KR)	ICN	x
33	Toronto Pearson, Toronto (CAN)	YYZ	x
35	Chhatrapati Shivaji, Mumbai (IN)	BOM	x
40	Sydney Kingsford-Smith, Sydney (AU)	SYD	Kein Direktflug
41	Sao Paolo-Guarulhos, Sao Paolo (BR)	GRU	x
44	Taiwan Taoyuan, Tapei (TW)	TPE	Kein Direktflug
45	Benito Juarez, Mexico City (MX)	MEX	Kein Direktflug
49	Ninoy Aquino, Manila (PH)	MNL	Kein Direktflug

Bemerkung: Anstelle von ATL habe ich mich für New York (JFK) entschieden, da nach Atlanta nur sehr wenige Direktflüge vorhanden sind.

## INTERACTIVE 1.1 Swiss-Linienflüge von und nach Zürich



## **Applikation zur Datenerhebung**

Für die Datenerhebung gemäss den vorgestellten Kriterien habe ich ein Java-Programm, ein sogenannter Crawler, entwickelt. Dieser fragt pro Tag viermal via API die Flugpreise für alle zwanzig Destinationen ab. Die Abfrage wird jeweils für den nächsten Tag und bis 90 Tage in die Zukunft durchgeführt.

Beispiel: Abfrage am Montag, 01. Juni 2016

Die Abfrage liefert die Flugpreise für alle Destinationen für den 2. Juni bis zum xx. August 2016.

Ursprünglich wollte ich die Abfrage via Google Flight API durchführen. Google erlaubt jedoch nur 50 kostenfreie Abfragen pro Tag. Bei 20 Destinationen à 90 Tage und vierfacher Abfrage pro Tag sind diese Gratis-Abfragen viel zu wenig. Aus diesem Grund musste ich mich nach einer Alternative umsehen. Die Lösung ist die Verwendung des Travel APIs von Skyscanner Business , welches eine unbegrenzte Anzahl Abfragen zulässt.

[http://business.skyscanner.net/portal/en-GB/Documentation/ApiOverview?\\_ga=1.20813560.1378450330.1462389746](http://business.skyscanner.net/portal/en-GB/Documentation/ApiOverview?_ga=1.20813560.1378450330.1462389746)

Die Antworten von Skyscanner werden transformiert und in einer neo4j Datenbank gespeichert.

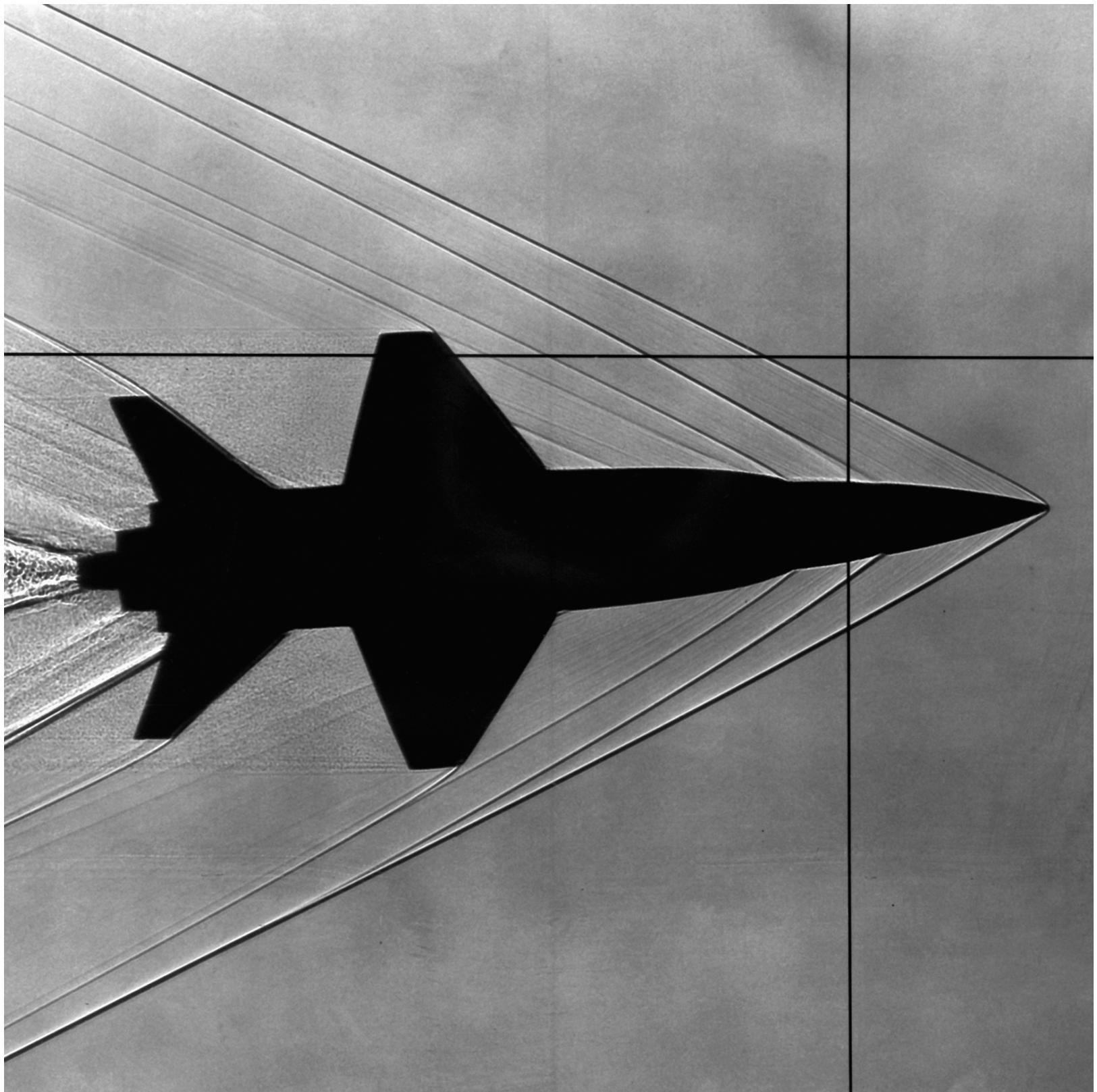
# Daten

## Datenanalyse

*Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam tristique nibh ac sem consectetur tristique. Nulla mauris odio, dignissim varius neque at, ultrices efficitur lacus. Maecenas cursus dignissim mauris vitae molestie. Ut luctus tortor in ultricies condimentum. Aenean auctor ante libero, vel facilisis felis gravida eget. Donec felis quam, consequat tempor orci vitae, viverra iaculis purus. Ut cursus eros sed erat dictum, et viverra elit dictum. Mauris egestas felis urna, nec ultricies purus tempus ut. Aenean tortor turpis, lobortis sit amet suscipit at, volutpat sit amet enim.*

Morbi dictum ligula ante, sit amet ultricies est bibendum non. Nulla massa elit, efficitur in viverra ut, convallis et leo. Nunc malesuada vitae libero ut posuere. Donec quis libero non orci malesuada posuere. In cursus risus in ante elementum fringilla. Maecenas pharetra leo at sollicitudin interdum. Vivamus aliquam, metus non ultrices aliquet, lectus turpis pulvinar diam, vel hendrerit nisi nunc elementum arcu. Suspendisse quis massa hendrerit, elementum dui quis, tristique magna. Sed condimentum in mauris ac tempor. Aenean lacus diam, commodo non fringilla eu, feugiat ac odio. Curabitur eu ultrices orci.

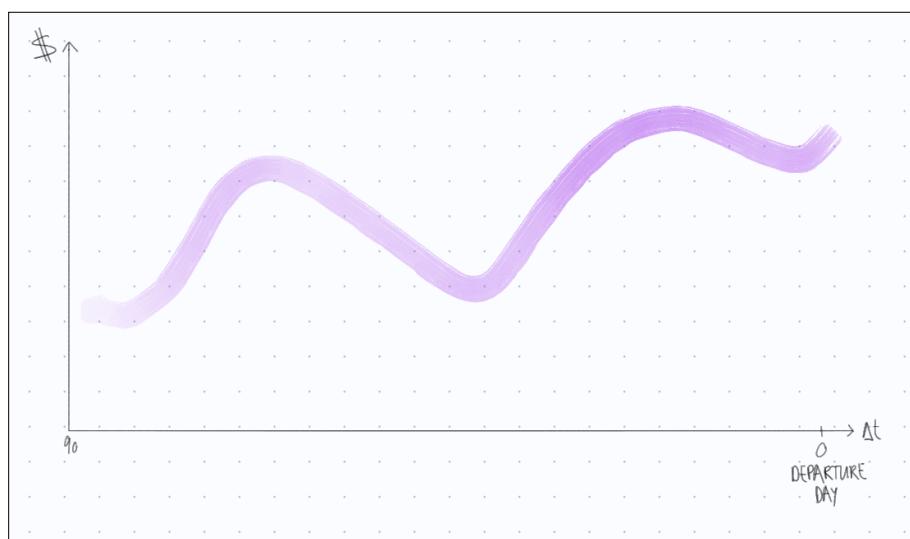
Donec efficitur tellus lectus, varius commodo quam blandit nec. Aenean non euismod justo. Morbi vitae libero quis tellus accumsan convallis nec sit amet arcu. Nulla mollis turpis ornare fringilla lacinia. Integer pharetra odio lacus, ut tincidunt libero pellentesque eleifend. Etiam sit amet felis ac arcu facilisis semper et nec nisi. Aliquam purus magna, porta non justo et, rutrum placerat velit. Mauris a enim imperdiet, ultrices dui ut, mollis sem. Donec eu viverra urna, a vehicula ex. Sed ullamcorper, nibh sed tristique scelerisque, massa sem luctus mauris, vel sollicitudin arcu quam quis lorem. Etiam vulputate, nibh porta varius tempor,



## Visuelle Konzeption

# Konzeption/Paper Prototyping

In der Konzeption habe ich mit verschiedenen Visualisierungsformen verschiedene horizontale Prototypen erstellt. In dieser Phase des Projektes ging es mir darum, möglichst breit zu denken. Als Ziel dieser Phase hatte ich die Selektion von einem bis zwei Prototypen, welche ich im Coding mit meinen Daten verbinden wollte. Aufgrund der grossen Datenmenge war es schwierig und sehr zeitaufwändig bei Papierprototypen die echten Daten zu verwenden.

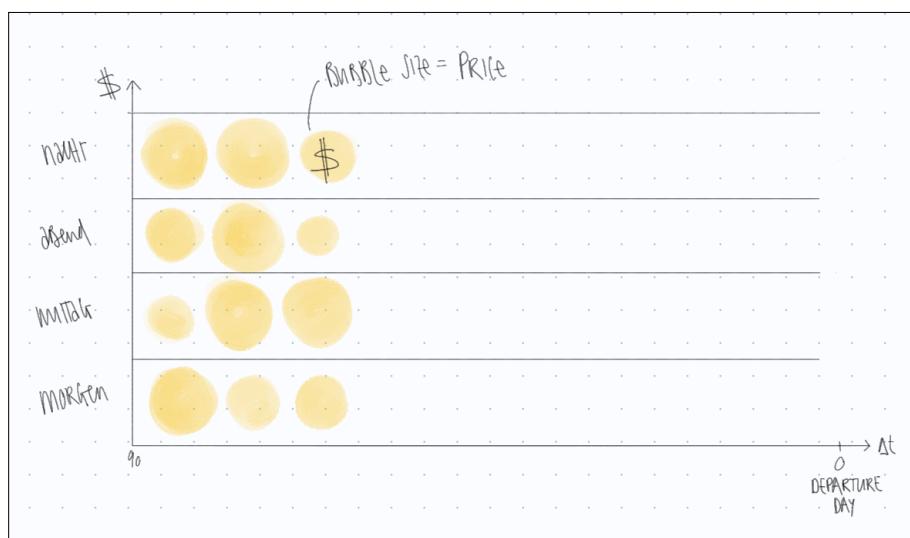


## Liniendiagramm

Eine einfache Darstellung ist das Liniendiagramm. Die Linie zeigt die Preisentwicklung über die Zeit vor dem Abflug.

Vorteil Preisentwicklung als Liniendiagramm ist geläufig.

Nachteil Die einzelnen Flüge können nicht sinnvoll überlagert werden. Multiples wären eine Möglichkeit dazu. Allerdings muss mit mehr als 100 Flügen pro Destination gerechnet werden. Dies ist mit Multiples auch nicht mehr sinnvoll darzustellen.

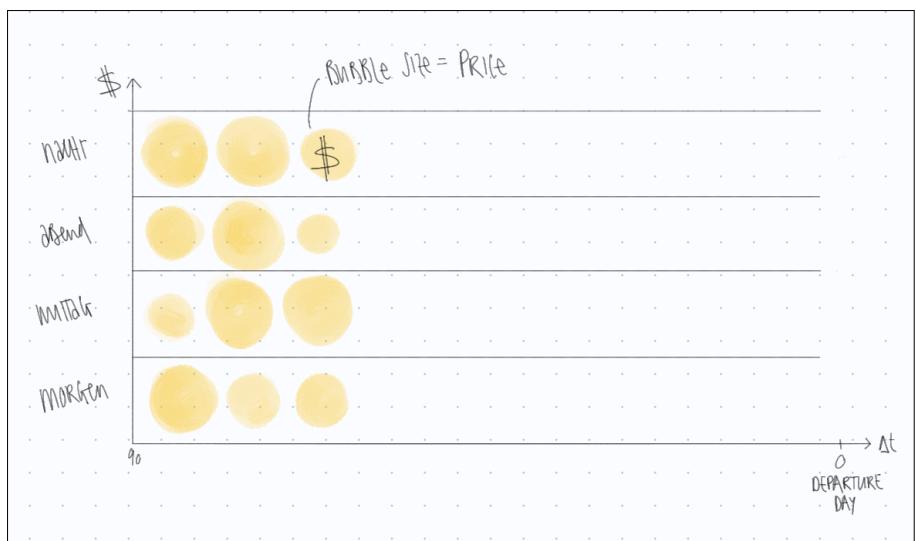


## Bubble Chart

Im Bubble Chart wird die Preishöhe mit der Grösse des Bubbles visualisiert.

Vorteil:

Nachteil: Analog dem Liniendiagramm. Die Y-Achse könnte alternativ anstelle der Tageszeit der Anfrage den Anfragetag an sich darstellen. Damit wäre eine



## Matrix

Ähnlich die das Bubble Chart. Die Preise werden jedoch über eine Farbkodierung dargestellt. Die Fläche ist unabhängig vom Preis einheitlich gross.

Vorteil Erlaubt eine gute Vergleichbarkeit innerhalb des Flugs und zwischen den Flügen und ist einfach verständlich.

Nachteil Keiner

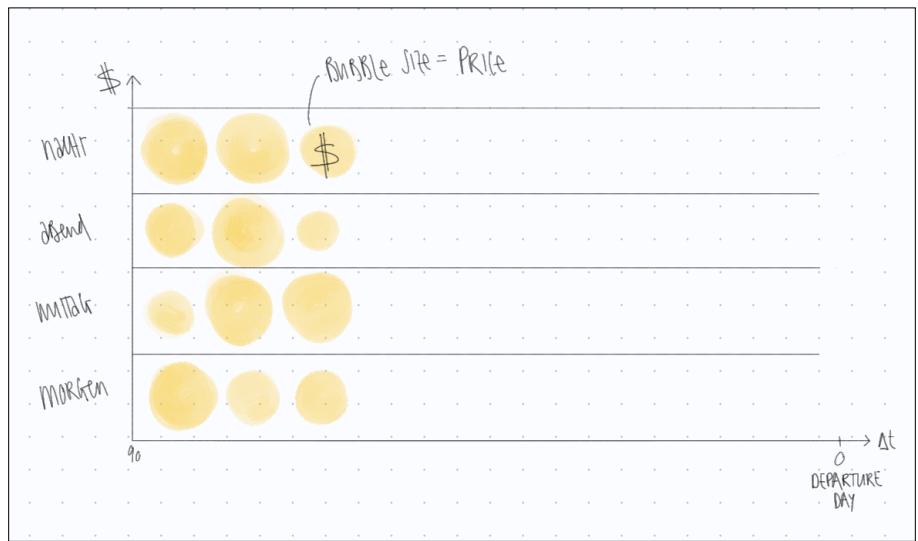


## Stacked Bar Chart

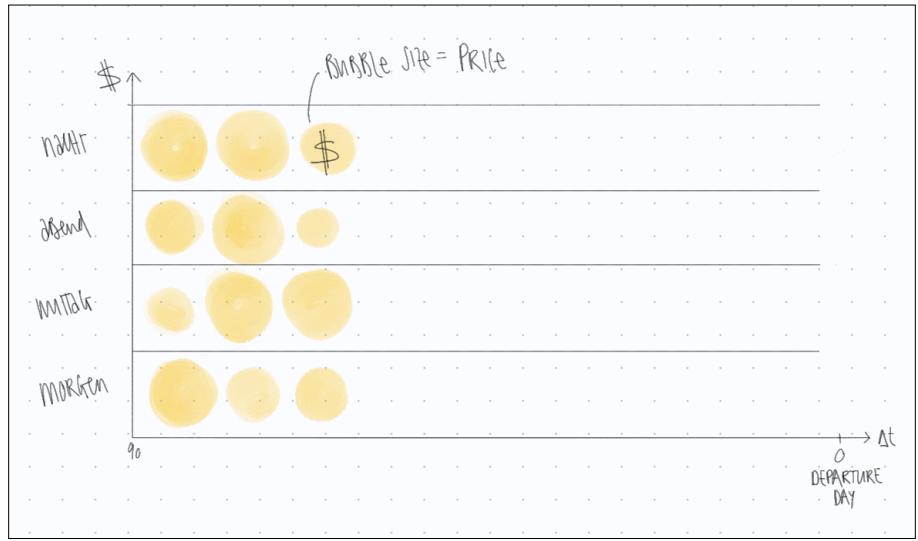
Die Preise werden in einem horizontalen Stacked Bar Chart dargestellt. Jeder Anfragetag vor Abflug ist entsprechend der Preishöhe breiter oder weniger breit.

Vorteil Kompakte Darstellung. Verschiedene Flüge können mit Multiples gut untereinander gelegt werden.

Nachteil Vergleichbarkeit ist eingeschränkt aufgrund der unterschiedlichen Balkenbreite. Es gibt kein Raster an dem sich der Benutzer orientieren kann. Außerdem werden die Balken unterschiedlich lang, was die Vergleichbarkeit zusätzlich erhöht.



**Spinnendiagramm**



**Flächendiagramm mit vertikaler Zeitachse**

# Visuelle Konzeption

## Erkenntnisse

*Aus der visuellen Konzeption wurden folgende Feststellungen festgehalten. Diese fliessen in die Umsetzungsphase ein.*

Folgende Feststellungen erachte ich als zentral für den weiteren Visualisierungssprozess:

- Die Verwendung von Multiples bietet sich an, um die einzelnen Flugverbindungen zu vergleichen. Dabei ist eine Gruppierung der Verbindungen in die zwei Kategorien Europa und Overseas denkbar.

*Nachtrag aus der Umsetzungsphase: Multiples um Destinationen zu vergleichen sind schwierig, da die Aussagekraft eines Vergleichs nicht sehr hoch ist (z.B. unterschiedliche Flugdauer, unterschiedliche Carrier und Anbieter, ...). Außerdem hat jede Destination unterschiedliche Anzahl Flüge, was ein Direktvergleich zusätzlich erschwert.*

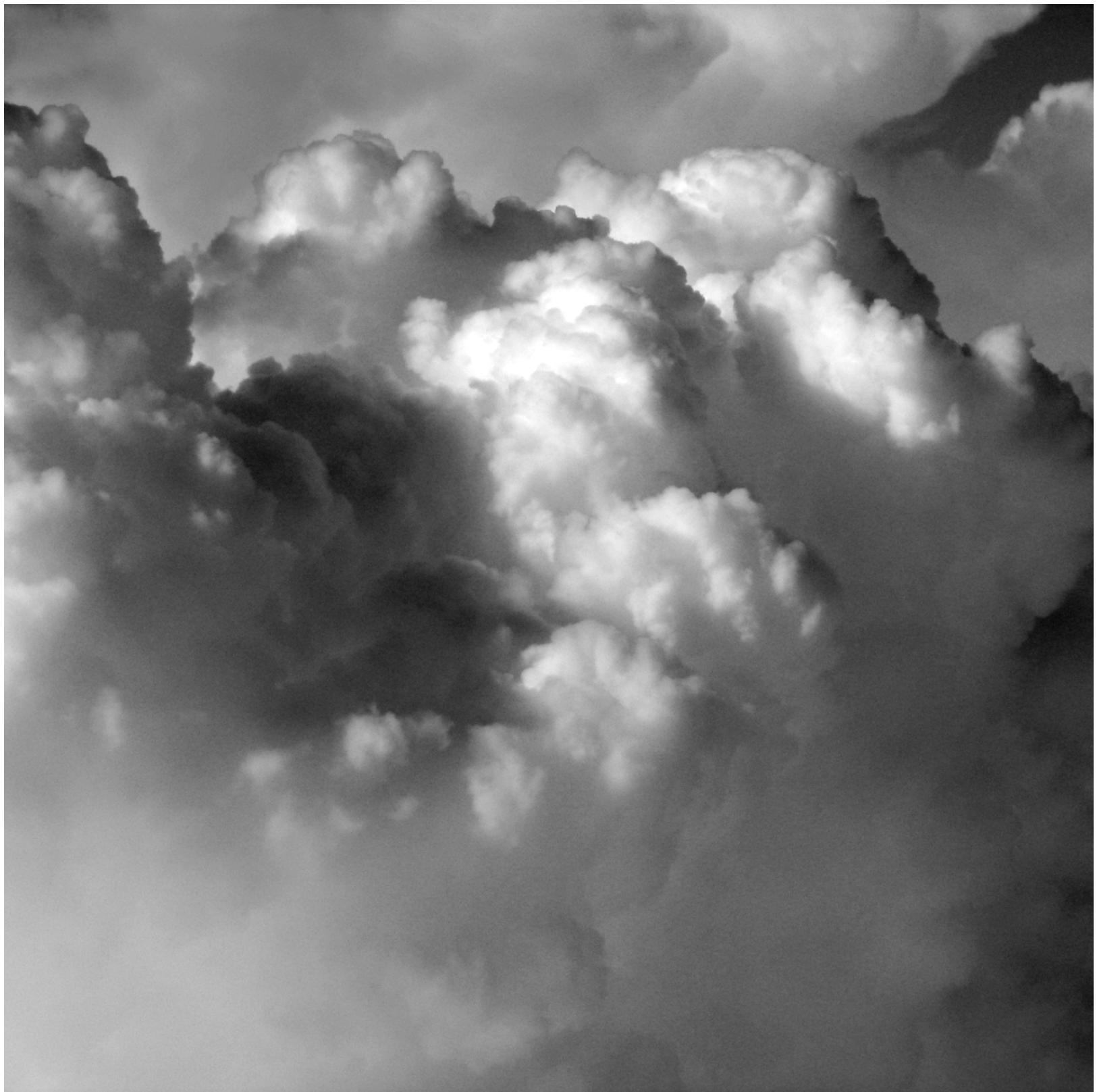
- Für eine Visualisierung einer Zeitvariablen bieten sich lineare Darstellungen an, ggf. können diese jedoch auch zweidimensional erfolgen (vgl. Abbildung 5, Skizze C).
- Wenn die Daten absolut sind, kann ein Wert auch auf einem Kalender gezeigt werden. Bei relativen Daten (z.B. 56 Tage vor Abflug) ist eine Anordnung auf einem Kalender nicht sinnvoll.
- Um die Vergleichbarkeit der Tage zu erhalten, sollte auf einem statischen Raster für die Tage gearbeitet werden. Dies ist umso wichtiger, wenn mit Multiples gearbeitet wird.
- Eine Visualisierung der Tage mittels Kreis scheint nicht sinnvoll, da es sich beim Beobachtungszeitraum von drei Monaten um keine abgeschlossene "Zeitreihe" handelt (wie z.B. 24 Stunden für ein Tag oder 12 Monate für ein Jahr).

- Eine Visualisierung durch eine Zeitachse in Kombination mit Flächen (vgl. Abbildung 3) scheint nicht zielführend, wenn kein Ganzes (100%) existiert. Analog verhält es sich beim Einsatz von TreeMaps.
- Eine Relativierung der Preisentwicklung durch Abbildung in unterschiedlichen Stufen (rot – blau mit unterschiedlicher Transparenz) schafft eine gute Vergleichbarkeit. Eine flächenproportionale Darstellung hingegen scheint mir eine erschwerete Vergleichbarkeit zu bieten (vgl. Abbildung 1, Skizze 2).

*Nachtrag aus der Umsetzungsphase: Es wurde eine einfarbige Skala (sequential) bevorzugt. Eine zweifarbige Skala (diverging) hat die Leserlichkeit der Matrix erschwert.*

- Die Vergleichbarkeit von Flugstrecken kann über die zurückgelegte Distanz eines Fluges, aber auch über die Dauer eines Fluges erreicht werden. Die Flugdauer ist unter Umständen sogar aussagekräftiger, da die Phasen "Start" und "Landung" bei jedem Flug gleich berücksichtigt werden (diese Aussage ist anhand der Daten zu prüfen).
- Mögliche Piktogramme für die Verwendung in den Visualisierungen könnten das Dollar-Zeichen (Preis) oder ein Flugzeug-Symbol sein.
- Der Einsatz einer typografischen Visualisierung kann für einen Vergleich qualitativer Merkmale der Destinationen, wie z.B. Flugdauer, etc., geprüft werden.

**Aus der visuellen Konzeptionsphase wurden die Fragestellungen nochmals geschärft, damit eine fokussiertere Visualisierung möglich ist (vgl. Kapitel Einleitung - Fragestellung, V2.1).**



**Coding**

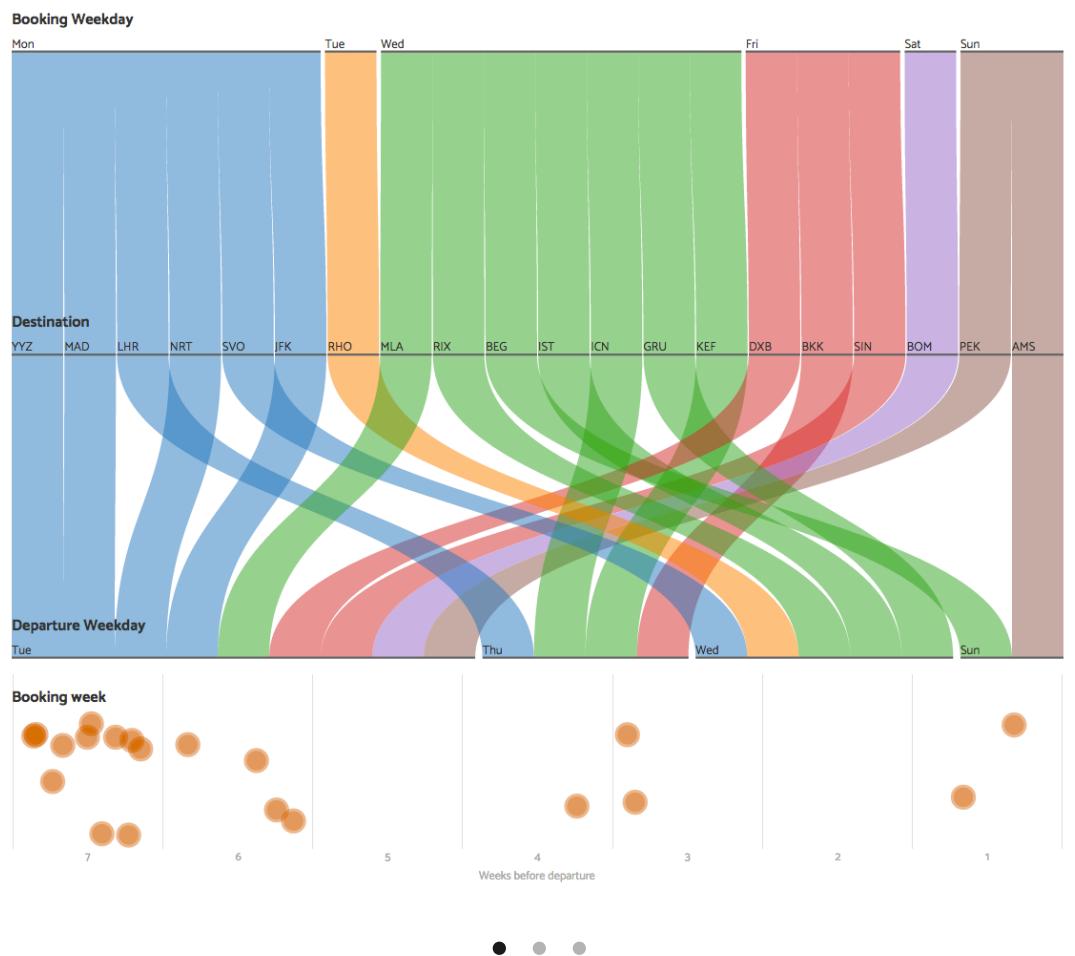
# Coding

## Frontend

*Im Frontend wird die Datenvisualisierung aufbereitet. Dazu werden die Daten vom Backend via REST-Service abgefragt. Für die Visualisierung werden die Daten mittels D3.js berechnet und dargestellt.*

Die Visualisierung besteht aus folgenden Teil-Visualisierungen:

1. Parallel Set kombiniert mit Zeitstrahl
2. Matrix mit interaktiver Preisentwicklung und -ersparnis
3. Durchschnittliche Preisersparnis pro Destination



Die erste Visualisierung adressiert die folgenden Fragen aus den initialen Fragestellungen:

- An welchem Wochentag soll ich abfliegen?
- An welchem Wochentag soll ich einen Flug buchen?
- Wie viele Wochen vor dem Abflug soll ich einen Flug buchen?

Für die Visualisierung habe ich mich für die Visualisierungsform des Parallel Sets entschieden. Die Fragen a. und b. fragen beide nach einem Wochentag. Dies ist eine kategorische, abgeschlossene Menge von Werten. Auch die Destination liegt in einem kategorischen Wertebereich.

Ergänzend zum Parallel Set habe ich die Wochen als Zeitstrahl visualisiert. Es wäre möglich gewesen, diese auch als weitere Kategorie in die Parallel Set Visualisierung aufzunehmen. Dagegen haben aus meiner Sicht jedoch zwei Gründe gesprochen.

Erstens wird die Anzahl Kategorien bei den Wochen mit zunehmender Datumsmenge und einer Verlängerung der Zeitreihen ansteigen. Die Anzahl Kategorien sind also nicht zwingend abschliessend.

Zweitens hat mir die Symmetrie in der Visualisierung mit den 3 Variablen "Booking Weekday", "Destination" und "Departure Weekday" sehr gut gefallen. Durch eine weitere Kategorie wäre aus meiner Sicht auch die Verständlichkeit der Parallel Set Visualisierung reduziert worden.

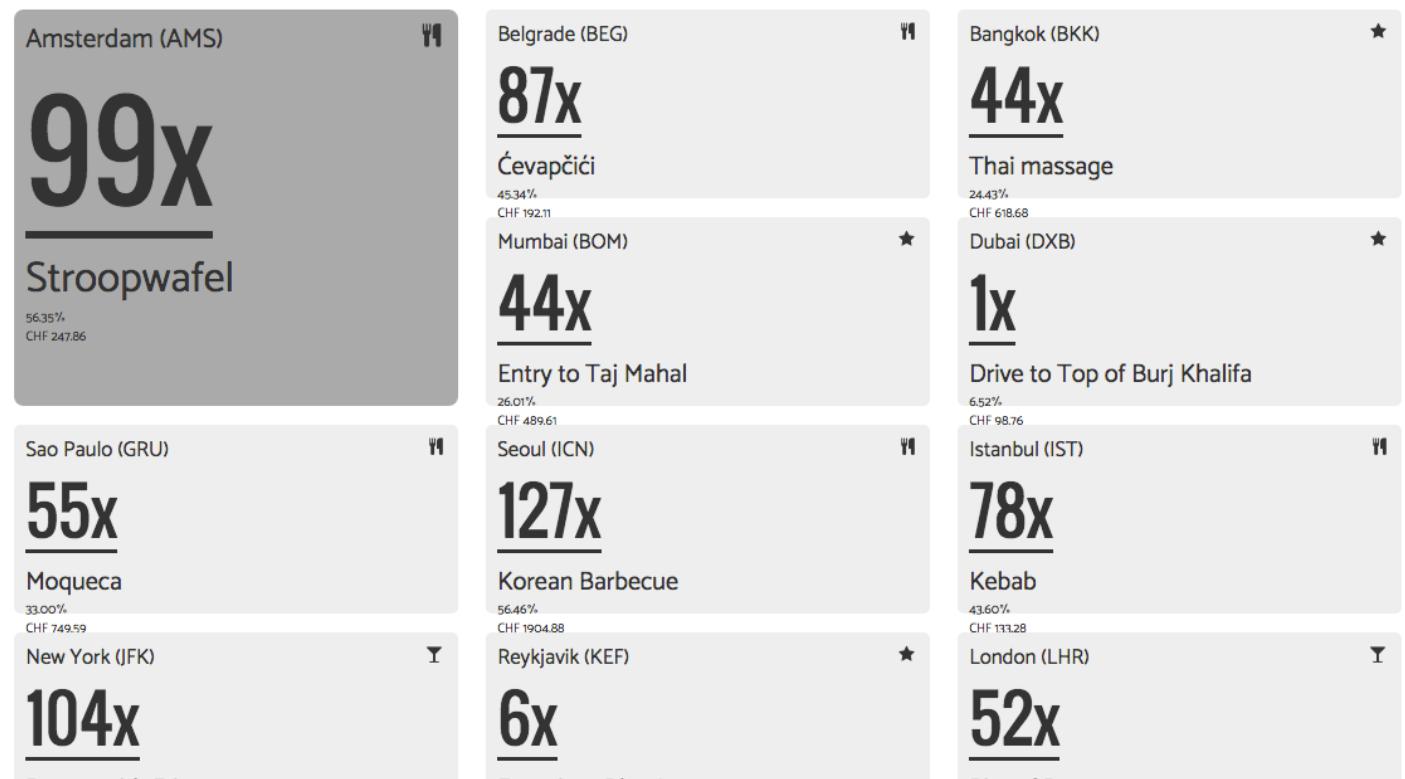
Da die beiden Visualisierungen jedoch miteinander kommunizieren ist für mich der Zusammenhang zwischen den beiden Teilen geglückt und die Visualisierung stellt sich als Ganzes dar.



Die zweite Visualisierung erlaubt dem Betrachter eine Vertiefung in die Materie.  
Sämtliche vollständigen Flüge werden in einer Matrix dargestellt.

Durch die Matrix kann der Benutzer schnell und einfach zwischen einzelnen Flügen vergleichen und so Muster erkennen.

Die Schwierigkeit der Visualisierung lag darin, das auf beiden Achsen Zeitreihen visualisiert werden. Die Höhe des Preises ist durch eine Farbkodierung in der einzelnen Zelle abgebildet.



Die letzte Visualisierung ist eine rein textuelle Visualisierung. Diese präsentiert dem Betrachter noch eine andere, unerwartete Seite.

Anstelle der Preisentwicklung steht hier nochmals die folgende Frage aus den initialen Fragestellungen im Zentrum:

a. Wie viel kann ich durchschnittlich pro Destination sparen, wenn ich zum richtigen Zeitpunkt buche?

# Coding

## Backend

*Das Backend bereitet die Daten aus der Datenbank auf und stellt diese via REST-Serice zur Verfügung. Folgende Schnittstellen stehen zur Verfügung.*

### **GET api/carrier**

Liefert alle Carrier von allen Destinationen zurück.

---

### **GET api/carrier/{destination}**

Liefert alle Carrier für eine Destination.

---

### **GET api/count/{carrier}**

Zählt die Anzahl Flüge für einen Carrier (alle Destinationen).

---

### **GET api/dayHistogramBook/{destination}**

Liefert das Historgramm der günstigsten Flüge nach Wochentagen für eine Destination.

Beispiel <http://ffv.mochila.ch/api/dayHistogramBook/lhr>

#### Resultat

```
{"Thu":546,"Tue":513,"Sat":567,"Wed":626,"Fri":413,"Mon":652,"Sun":561}
```

---

### **GET api/dayHistogramBook/{destination}/{carrier}**

Analog obiger Beschreibung, jedoch zusätzlich nach Carrier eingeschränkt.

---

## **GET api/destination**

Liefert alle Destinationen zurück.

---

## **GET api/destination/{destination}**

Liefert alle Flugpreise zu einer bestimmten Destination. Dies beinhaltet alle Preise für jeden Flug von 1 bis n Tage vor Abflug (Delta).

Beispiel <http://ffv.mochila.ch/api/destination/kef>

Resultat (gekürzt)

```
[{"carrier": "FI", "number": "569", "destination": "KEF", "origin": "ZHR", "departure": 1473249600000, "prices": [{"request": 1473112800000, "price": 586.18, "delta": 1}, {"request": 1468965600000, "price": 228.16, "delta": 49}], "minPrice": 228.16, "maxPrice": 586.18, "complete": true}, {"carrier": "FI", "number": "569", "destination": "KEF", "origin": "ZHR", "departure": 1471176000000, "prices": [{"request": 1471039200000, "price": 589.46, "delta": 1}, {"request": 1466892000000, "price": 447.61, "delta": 49}], "minPrice": 445.0, "maxPrice": 607.16, "complete": true}]
```

---

## **GET api/destination/{destination}/{carrier}**

Analog vorherigem Aufruf, jedoch nach Carrier eingeschränkt.

---

## **GET api/flights/{destination}**

Analog api/destination/{destination}, jedoch werden die Daten in der Ausgabe nicht hierarchisch ausgegeben, sondern flach.

Beispiel <http://ffv.mochila.ch/api/destination/kef>

Resultat (gekürzt)

```
[{"destination": "KEF", "origin": "ZHR", "carrier": "FI", "flightNumber": "FI569", "departureDate": "2016-09-07", "departureTime": "14:00:00", "dts": 1473249600000, "departureWeekday": "Mi", "requestDate": "2016-09-06", "deltaTime": 1, "price": 586.18, "bin": 6}, {"destination": "KEF", "origin": "ZHR", "carrier": "FI", "flightNumber": "FI569", "departureDate": "2016-09-07", "departureTime": "14:00:00", "dts": 1473249600000, "departureWeekday": "Mi", "requestDate": "2016-09-05", "deltaTime": 2, "price": 583.97, "bin": 5}...]
```

---

### **GET api/flights/{destination}/{carrier}**

Analog vorigem Aufruf mit zusätzlicher Einschränkung nach Carrier.

---

### **GET api/min/{destination}**

Liefert den günstigsten Preis für eine Destination über alle Flüge. Als Resultat wird der Tag vor Abflug sowie die Wahrscheinlichkeit zurückgegeben.

Beispiel <http://ffv.mochila.ch/api/min/kef>

Resultat {"value": "14", "propability": 0.181818181818182}

---

### **GET api/min/{destination}/{carrier}**

Analog vorigem Aufruf, jedoch weitere Einschränkung nach Carrier.

---

### **GET api/minWeekdayBook/{destination}**

Liefert den günstigsten Wochentag zum Buchen für die angegebene Destination. Als Ergebnis wird der Wochentag mit der zugehörigen Wahrscheinlichkeit geliefert.

Beispiel <http://ffv.mochila.ch/api/minWeekdayBook/kef>

Resultat {"value": "Wed", "propability": 0.1661807580174927}

---

### **GET api/minWeekdayBook/{destination}/{carrier}**

Analog vorigem Aufruf mit zusätzlicher Einschränkung nach Carrier.

---

### **GET api/minWeekdayFlight/{destination}**

Liefert den günstigsten Wochentag zum Abfliegen für die angegebenen Destination. Als Ergebnis wird der Wochentag mit der zugehörigen Wahrscheinlichkeit geliefert.

Beispiel <http://ffv.mochila.ch/api/minWeekdayFlight/kef>

Resultat {"value": "Thu", "propability": 0.2727272727272727}

---

### **GET api/minWeekdayFlight/{destination}/{carrier}**

Analog vorigem Aufruf mit zusätzlicher Einschränkung nach Carrier.

---

### **GET api/minhist/{destination}**

Liefert ein Histogramm, welches über alle Flüge einer Destination die günstigsten Flüge pro Delta-Tag (1 bis n Tage vor Abflug) aufzeigt.

Beispiel <http://ffv.mochila.ch/api/minhist/kef>

Resultat

[1,0,0,1,0,0,5,0,0,0,0,1,1,12,2,1,3,0,2,0,2,1,4,2,2,2,0,2,2,1,2,1,0,0,2,1,1,0,1,1,3,7,0,0,0,0,0,0]

---

### **GET api/minhist/{destination}/{carrier}**

Analog vorigem Aufruf mit zusätzlicher Einschränkung nach Carrier.

---

### **GET api/savings/{destination}**

Liefert die durchschnittliche Preisersparnis für die gewählte Destination. Dazu wird für jeden Flug die Differenz zwischen minimalem und maximalem Flugpreis berechnet. Am Ende wird von allen Differenzen der Mittelwert berechnet.

Das Resultat wird als absolute Preisdifferenz sowie in Prozent zurückgeliefert.

Beispiel <http://ffv.mochila.ch/api/savings/kef>

Resultat {"absolut":324.95257575757574,"relative":0.40469160161980877}

---

### **GET api/savings/{destination}/{carrier}**

Analog vorigem Aufruf mit zusätzlicher Einschränkung nach Carrier.

---



# Form und Methode

**Die Fülle von gesammelten Daten soll dem Betrachter in Form eines Explorable zugänglich gemacht werden. Aus diesem Grund soll die Visualisierung als interaktive Webseite realisiert werden. Um dem Betrachter trotzdem einen Teil einer Geschichte zu erzählen werden die Grafiken um erläuternde Texte ergänzt. Die restliche Geschichte soll dann im Kopf des Betrachters entstehen.**

Im Laufe des Projekts hat sich dann die ursprünglich gewählte Form weiter bestätigt. Um dem Zugriffstrend von Webseiten gerecht zu werden, habe ich die Webseite responsive gemacht. Aufgrund der Datenmenge ist es teils zwar etwas gar klein, jedoch kann die Webseite noch immer betrachtet werden.

Aus zwei Gründen habe ich mich im Verlauf des Projekts für mehr als eine Visualisierung entschieden:

- Auf einem Bildschirm (bzw. Mobile Device) steht nur beschränkter Platz zur Verfügung. Um die Verständlichkeit zu gewähren muss die Grafik unterteilt werden.
- Eine einzelne Grafik soll - zumindest auf dem Desktop - auf einem Bildschirm mit heutiger Auflösung ohne Scrollen platzfinden.
- Die unterschiedlichen Fragestellungen besitzen unterschiedliche Betrachtungslevel. Diese in einer gemeinsamen Grafik zu visualisieren wäre nicht sinnvoll.
- Aus Sicht des Storytellings macht es durchaus Sinn eine Visualisierung in mehrere Grafiken zu unterteilen. Dies erlaubt es eine entsprechende Geschichte glaubwürdig anhand des Erzählbogens aufzubauen.

Neben der Webseite wurden für die Ausstellung Plakate sowie Visitenkarten erstellt. Die Visitenkarten dienen als "Brücke" zwischen der physischen und der digitalen Welt.



Anhang

# Anhang

## Lessons Learned

Folgende Erfahrungen habe ich im Verlauf des Autorenprojekts für mich gemacht.

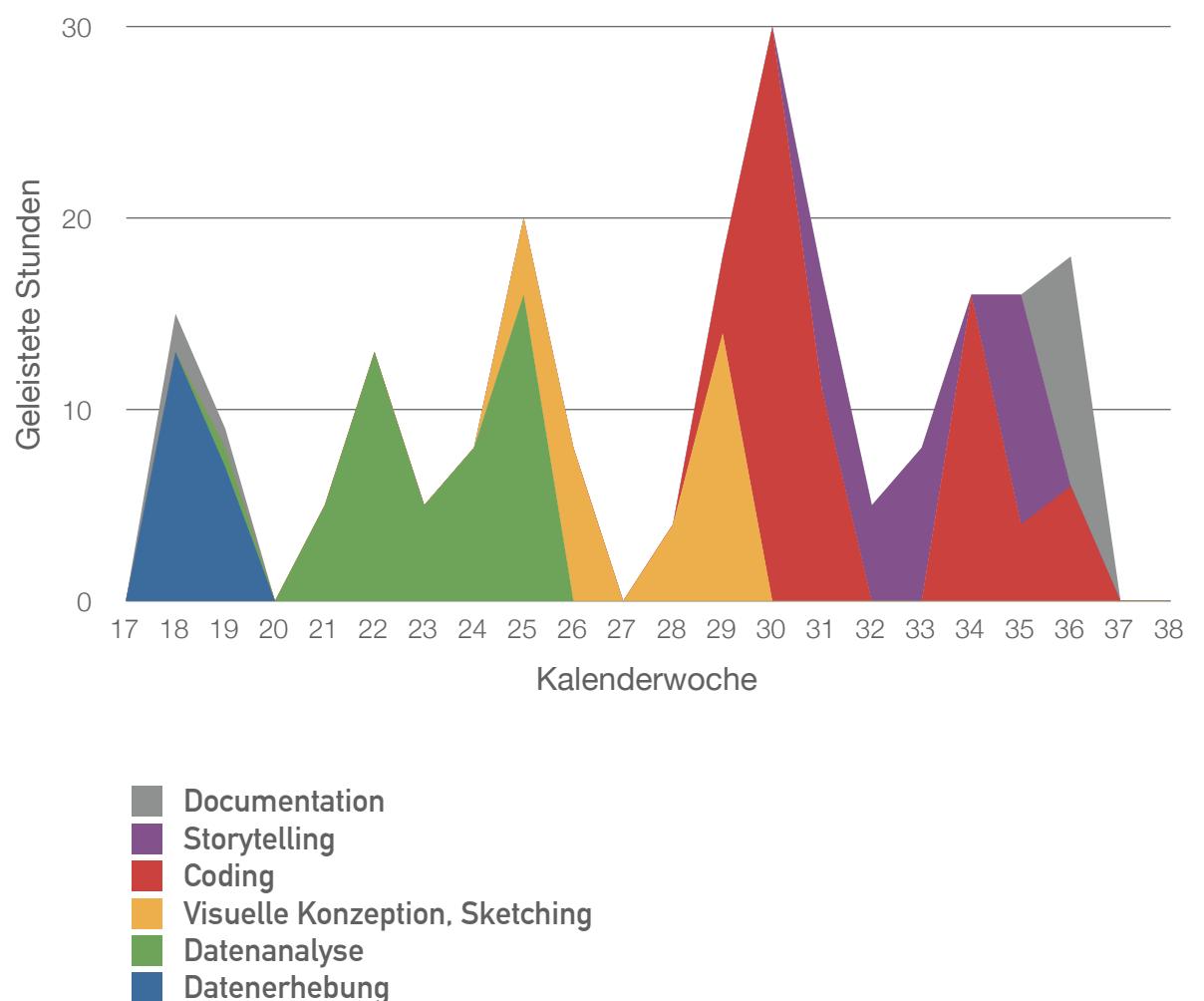
1. Erst mit einer klaren Vorstellung an die Visualisierung gehen.
2. In kleinen Iterationen (Rapid Prototyping) einzelne Visualisierungen mit den echten Daten prüfen.
3. Interaktion benötigt in der Umsetzung viel Aufwand. Deshalb soll diese nur gezielt eingesetzt werden (vgl. 5.). Außerdem muss diese im Optimalfall noch auf das Endgerät (Desktop, Mobile) angepasst werden, da gewisse Interaktionen nur für bestimmte Endgeräte sinnvoll sind (z.B. MouseOver).
4. Eine klare Priorisierung der Features vornehmen. Wichtiges von “Nice-To-Have” trennen.
5. Kosten-Nutzen immer vor Augen halten.
6. dd

# Anhang

## Administratives

### Arbeitsnachweis

Die aufgeführten Stunden sind die Aufwände, welche ausserhalb der Kursmodule an der HKB für das Autorenprojekt investiert wurden.



# Anhang

## Links

In der Linkssammlung habe ich einige Links gesammelt, welche mir während dem CAS weitergeholfen haben bzw. die ich noch im Detail betrachten möchte.

### Datenerhebung/-analyse

#### Coding

Learn JS Data - Data manipulation, munging and processing in JavaScript  
<http://learnjsdata.com/index.html>

Data Scrolling for Explainers - A tutorial  
<http://vallandingham.me/scroller.html>

Textures.js - SVG patterns for Data Visualization  
<http://riccardoscalco.github.io/textures/>

### Visuelle Konzeption, Inspiration

#### Dokumentation

iBooks Author Templates  
<https://itunes.apple.com/us/app/templates-for-ibooks-author/id527161787?mt=12>  
<https://www.ibooksauthortemplates.com>



# Copyright

© Ruth Ziegler, CAS Data Visualization 2016

*Hiermit bestätige ich, die vorliegende Arbeit mit eigenen Händen gemacht zu haben und blablabla.*

*Ruth Ziegler  
Steinhofstrasse 44  
CH-6005 Luzern  
ziegler.ruth@gmail.com  
<http://www.mochila.ch>*

# **Lorem**

*Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.*

---

## **Related Glossary Terms**

Drag related terms here

---

**Index**

[Find Term](#)

[Unknown glossary link](#)