// Geosoft I

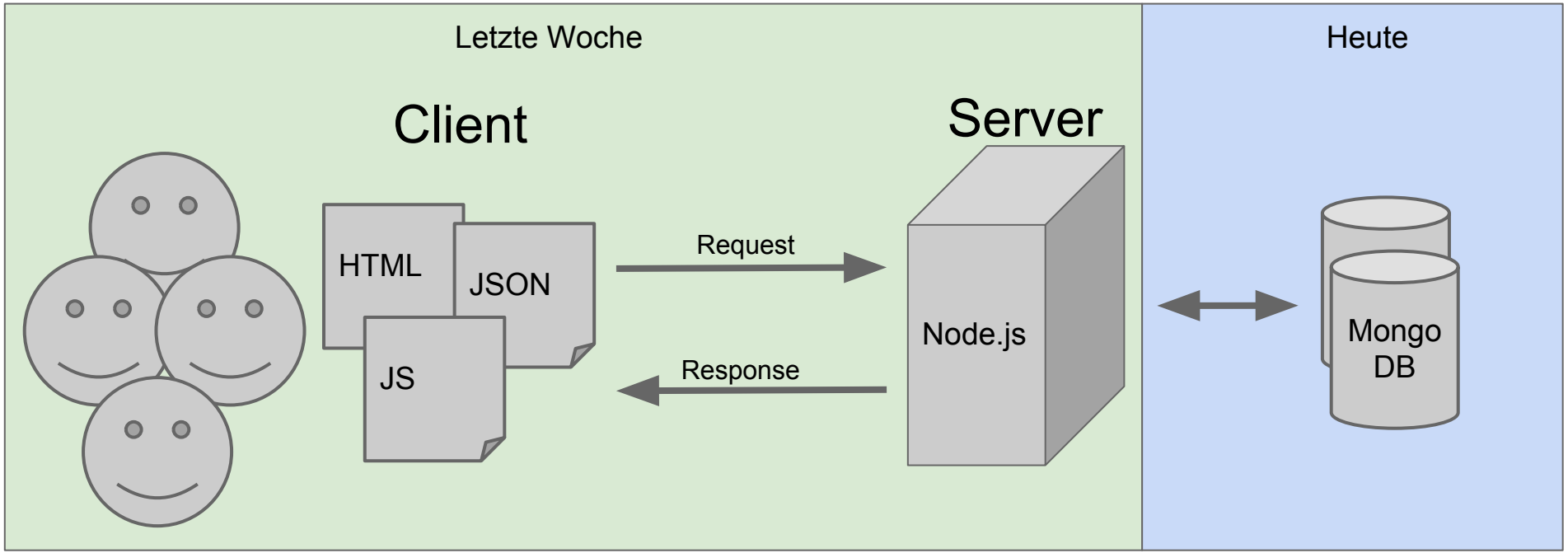
// Termin 0000000000001000

// Agenda

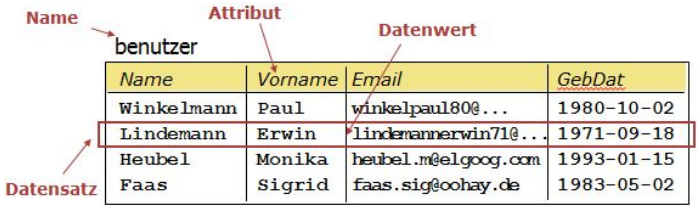
1. Code Review
2. Node.js Mongo DB
3. Node.js Streams
4. Node.js Websockets

Node.js MongoDB

// Node.js II


Letzte Woche
Heute
Client
Server
HTML
JSON
JS
Request
Response
Node.js
Mongo
DB

// Node.js II



Name	Vorname	Email	GebDat
Winkelmann	Paul	winkelpaul80@...	1980-10-02
Lindemann	Erwin	lindemannerwin71@...	1971-09-18
Heubel	Monika	heubel.m@elgoog.com	1993-01-15
Faas	Sigrid	faas.sig@oohay.de	1983-05-02

- NoSQL Datenbanken sind hip
  - “not only SQL”
  - nicht-relational
  - Vorteile im Web
    - schnell einsatzbereit
    - optimiert für viele Schreib-/Lesevorgänge
    - flexiblere Datenmodelle

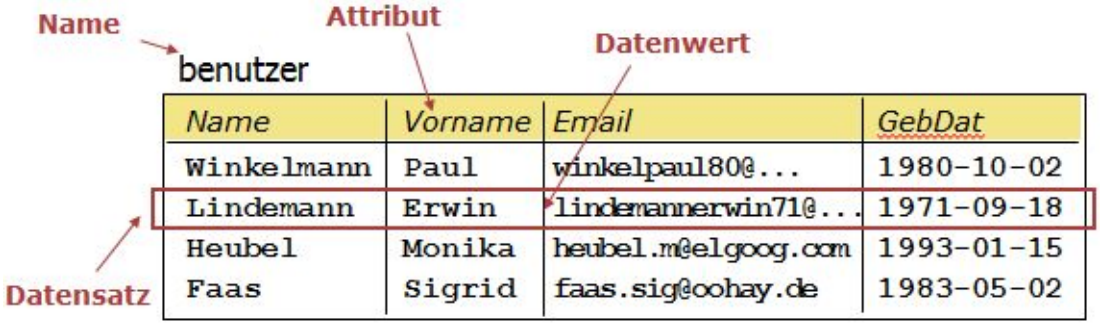
// Node.js II

Bildquelle:
http://www.linux-magazin.de/var/linux_magazin/storage/images/media/linux-magazin/ausgabe/2010/09/schneller-als-die-datenbank/abb1.png/540859-1-ger-DE/abb1.png.png

// Warum MongoDB?

- Frei und Open Source
- Skaliert sehr gut
- Wird von vielen Betreibern eingesetzt

Quelle: https://www.mongodb.com/mongodb-scale



SQL DB

```
{
  "Name": "Lindemann",
  "Vorname": "Erwin",
  "Email": "lindemannerwin71@...",
  "GebDat": "1971-09-18"
}
```

MONGO DB

// Node.js II - MongoDB

/* MongoDB Live Action*/

// Resourcen

- https://www.mongodb.com/what-is-mongodb
- https://www.mongodb.com/nosql-explained
- Tutorial (Part 3 bis 5): https://goo.gl/TdOi2f

Node.js Streams

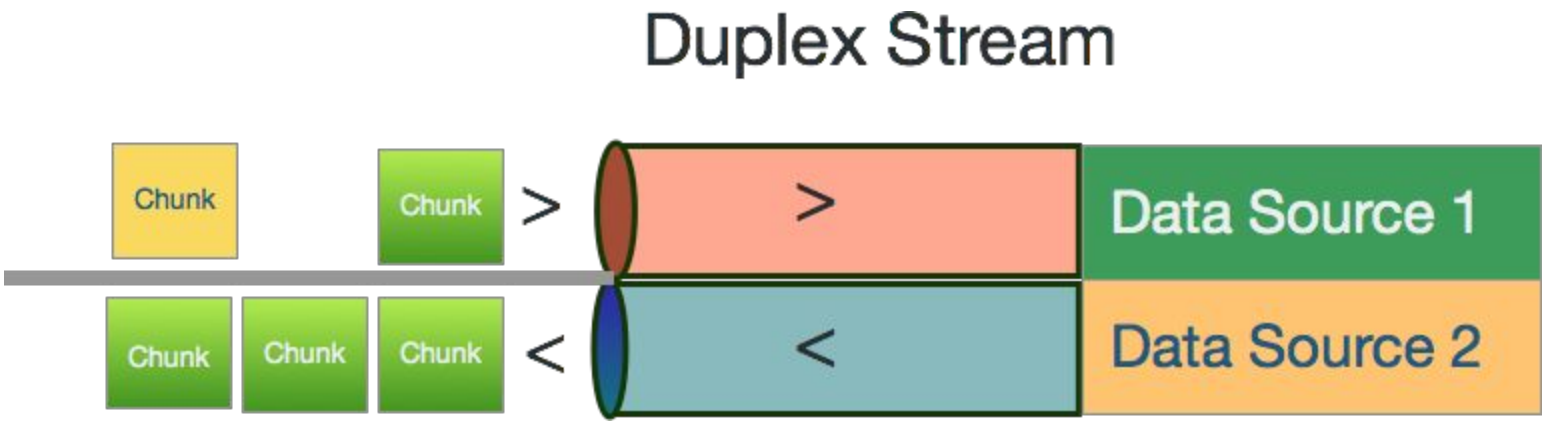
// Node.js Streams

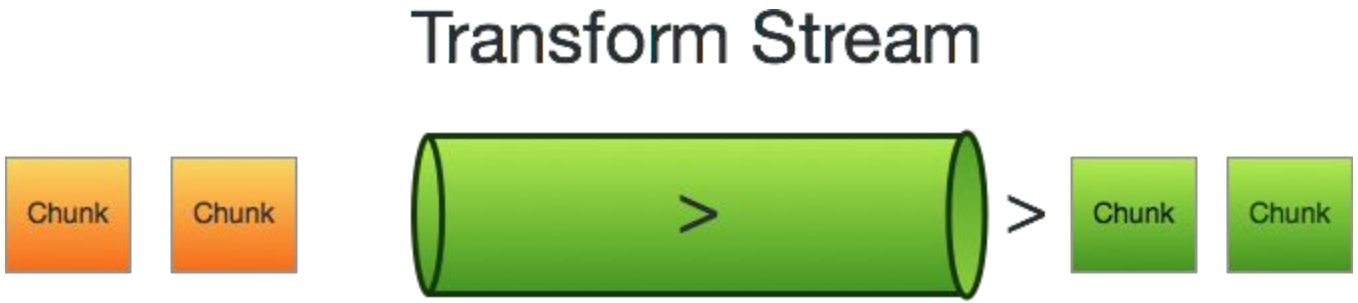
- Wozu?
  - Effiziente(re) und skalierbare Anwendung entwickeln

- Was ist es?
  - Vergleichbar mit einem Array
  - Unterschied
    - Arrays bewahren alle Daten zeitgleich im Speicher
    - Streams werden peu à peu mit Daten gefüllt

// Node.js - Arten von Streams

- Lesbare Streams (Daten aus einer Datenquelle auszulesen)
- Schreibbare Streams (Transport von Daten zu einer Datenquelle)
- Duplex Streams (sowohl les- als auch schreibbar)
  - Transform Streams





Quelle:http://hugozap.com/development/2015/11/01/nodejs-streams-intro-diagrams

// Node.js II - Stream

/* Stream Live Action*/

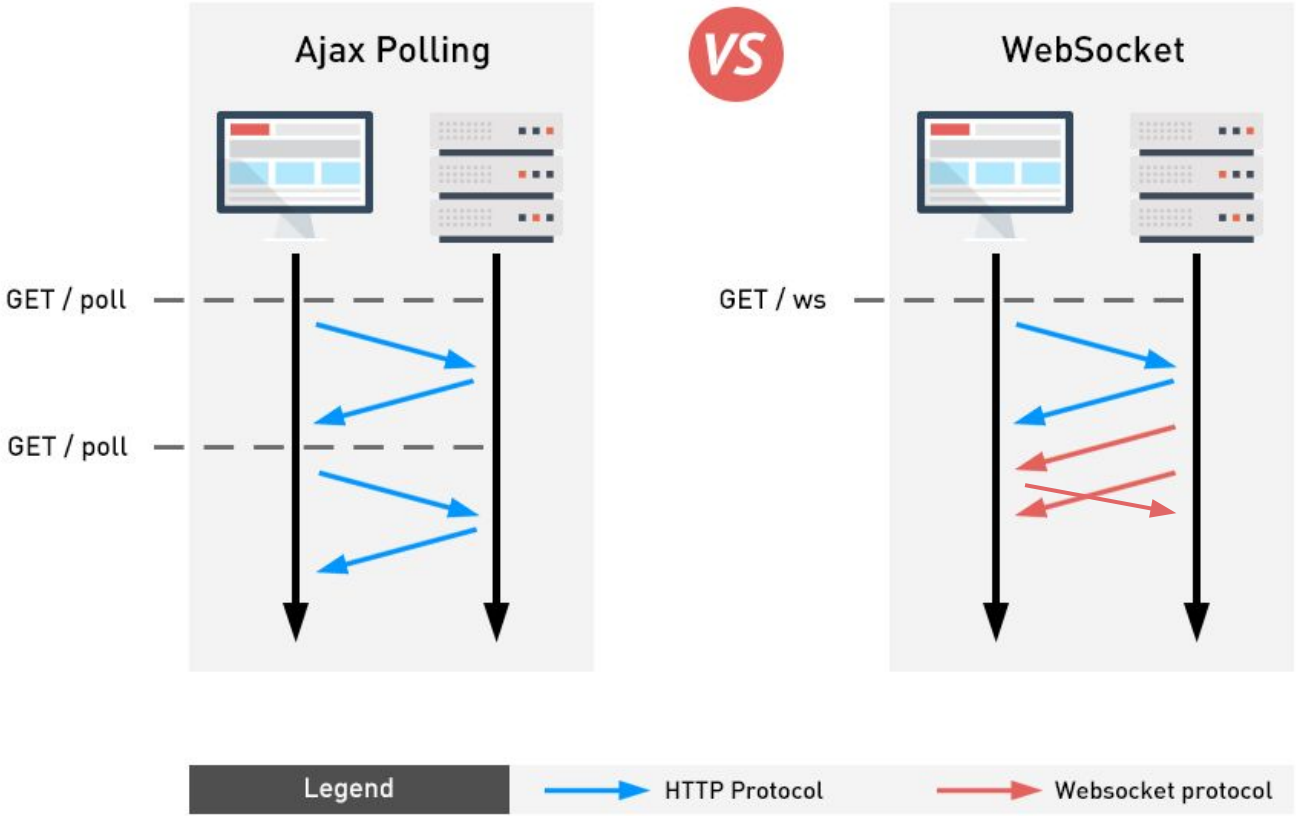
// Resourcen

- https://www.heise.de/developer/artikel/Einfuehrung-in-Node-js-Folge-17-Streams-verwenden-3705280.html

- https://nodejs.org/api/stream.html#stream_duplex_and_transform_streams

Node.js Websockets

// Node.js II - Websockets



Bildquelle: https://blog.resellerspanel.com/wp-content/uploads/2016/03/websocket-scheme.png

Node.js II - Websockets (Beispiel)

http://demos.fmeserver.com/fmepedia-real-time-drawing/

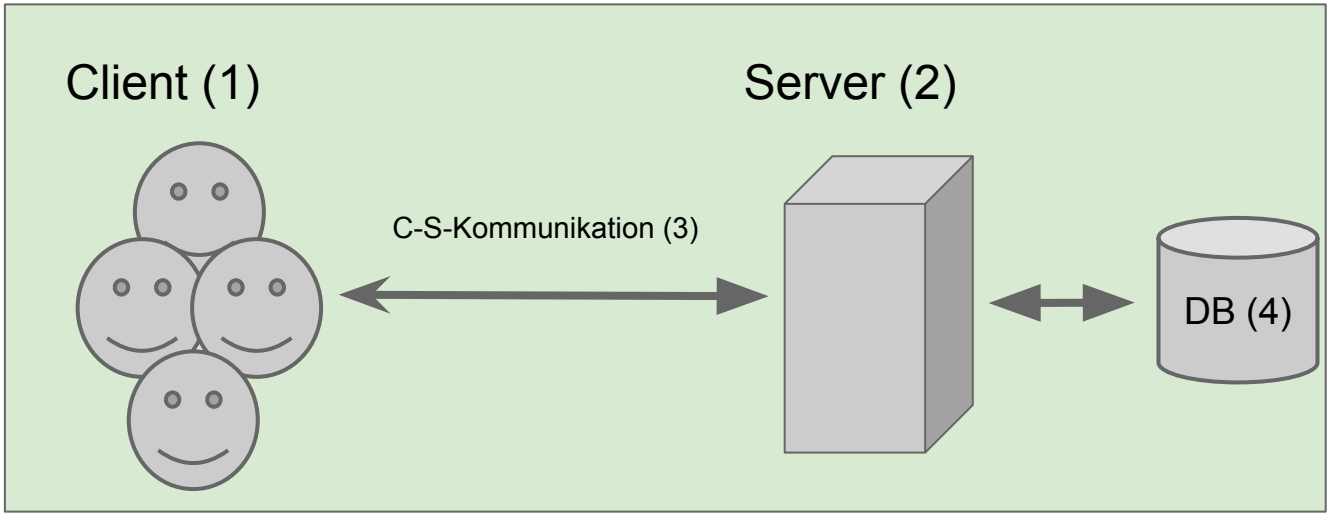
// Node.js II

- Socket.IO für bidirektionale Kommunikation via Websockets

- /* Websockets-Live-Action */

// Resourcen

- https://darrenderidder.github.io/talks/ModulePatterns
- http://nodecode.de/chat-nodejs-websocket (404 error?)
- https://github.com/nodecode



- Ajax (3)
- Bootstrap (1)
- CSS (1)
- HTML (1)
- JavaScript (1 & 2, 3, 4)
- jQuery (1, 3)
- JSNLOG (1, 3)
- JSON (1, 2, 3, 4)
- MongoDB (4)
- Node.js (2)
- Websockets (3)

// Aufgabe 6 (bis zum 06.07.2017)

Aufgabenteil 1 - Node.js Tutorial:

Arbeitet das Closebrace Node.js Tutorial (Dead-Simple Step-By-Step Guide - goo.gl/whUIz5) bis zum Ende durch. Ladet die erstellten und kommentierten Sourcecode - Dateien mit eurer Abgabe mit hoch.

// Aufgabe 6 (bis zum 06.07.2017)

Aufgabenteil 2:

Erweitert eure letzte Abgabe 5 um folgende Funktionalitäten:

- Speicherfunktion für die zuletzt in Leaflet gezeichnete Geometrie im GeoJSON Format mit einem frei wählbaren Namen in der Datenbank
- Laden der Geometrien aus der Datenbank. Danach sollen diese auf einer Karte angezeigt werden

Richtet dazu MongoDB und Node.JS ein, installiert einen MongoDB Treiber in Node.JS und nutzt in eurer Webseite Ajax für die Kommunikation zwischen lokaler Webseite und dem Node.JS Script. Node.JS muss dabei mindestens für die Datenbankkommunikation verwendet werden.

Achtet bitte dabei weiterhin darauf, dass der Code gut strukturiert, ausreichend kommentiert und das Logs ausgegeben werden. Gebt für die verwendeten Node-Module nicht den Ordner ‘node_modules’ ab, sondern stellt sicher dass die verwendeten Module in der ‘package.json’ Datei aufgelistet sind. Stellt sicher, dass die Anwendung direkt installierbar und auszuführen ist.

Abgabeformat:

- ZIP-Datei mit Namen im Format “Aufgabe_Aufgabennummer_Nachname1_Nachname2” (Beispiel “Aufgabe_1_Degbelo_Pfeiffer.zip”)

// Gruppeneinteilung

Friedrich, Jahnich, Formaniuk
Trzaska, Glahe
Thieme, Speckamp
Tebrügge, Bollow
Holtkamp, Heines

Pagel, Pfeiffer
Bagert, Tolksdorf
Buss, Karic
Tiemann, Seifert