Web Engineering

Dr. Michael Lesniak
mlesniak@micromata.de

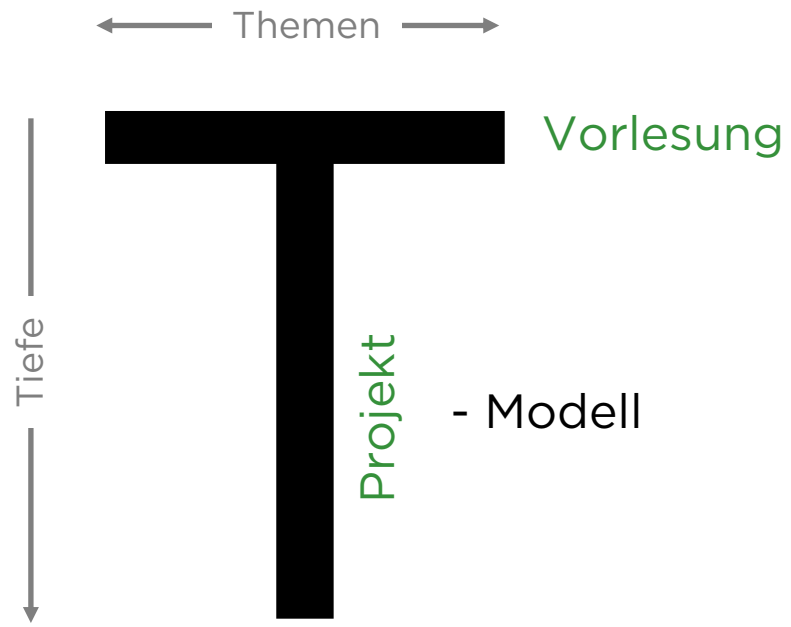
06. Mai 2019

MICROMATA

Organisatorisches

- Tempo und Schwierigkeit – bei keinem Feedback gehe ich davon aus, es ist ok.
  - relevant für a) Projektaufgabe und b) Verständnis in der Vorlesung
  - Feedback in #lecture-2019-05-06 oder @Michael Lesniak
- Themenwünsche in #lecture-request
- WIP: Pool für Übungen
- Gruppeneinteilung

Wissen (Vorlesung vs. Projekt)



Heute...

- Organisatorisches
- Review Übungsaufgaben
- HTTP – euer Wissen?
- Client- vs Serverseitiges Rendering (Architekturen)
- Code
  - Persistenz im Backend
  - Erste Schritte mit React im Frontend

---------- 5 Minuten Pause ----------

- Übung

Review Übungsaufgaben

- Review generell gewünscht? Abstimmung **jetzt** in #lecture-2019-05-06

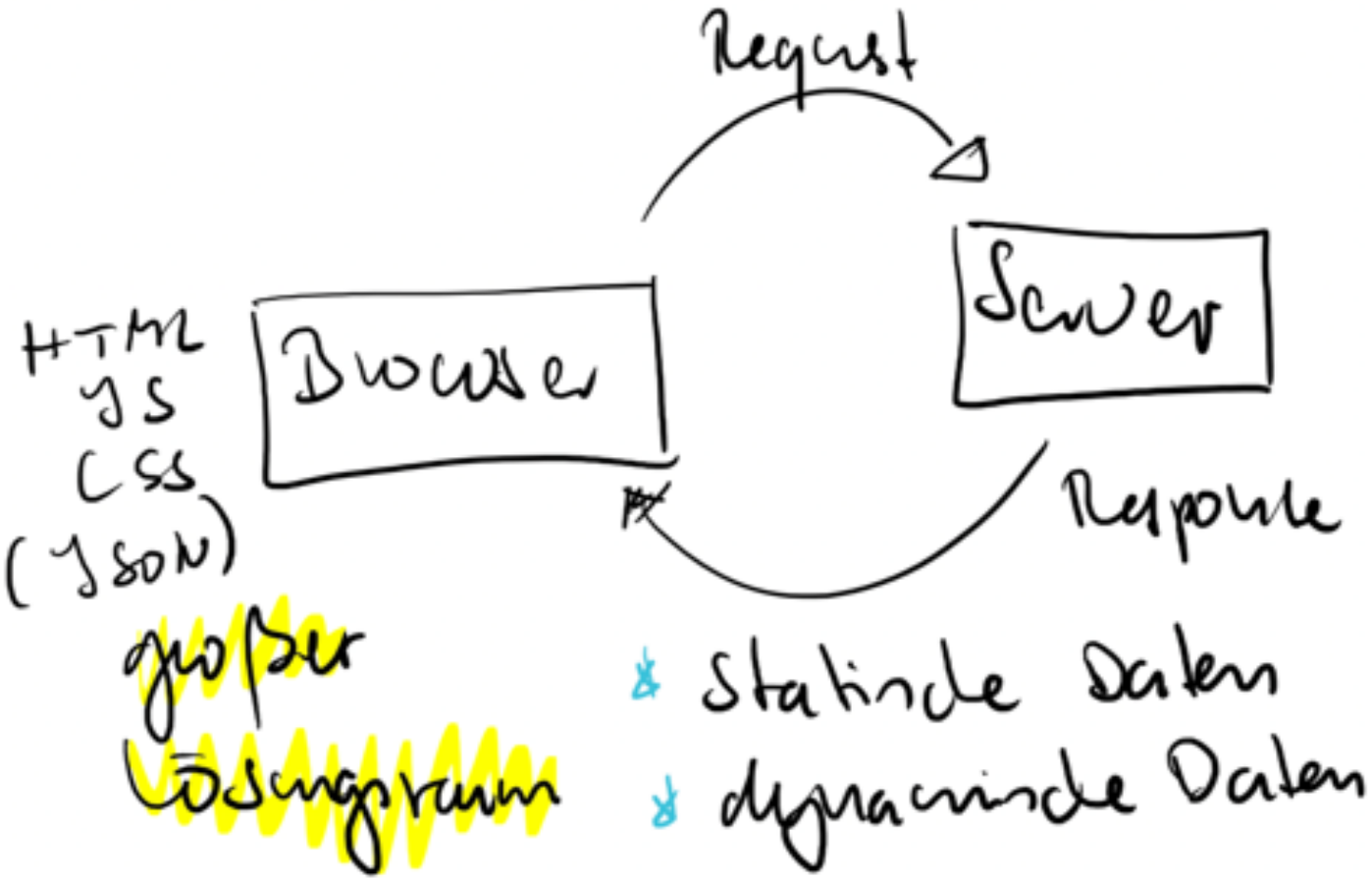
- Aufgaben
  - Nachbauen des Codes
  - Input-Möglichkeit und Verarbeitung im Backend
  - Frontend-Eingabemöglichkeit

HTTP-Basics (ggf. nächste Vorlesung)

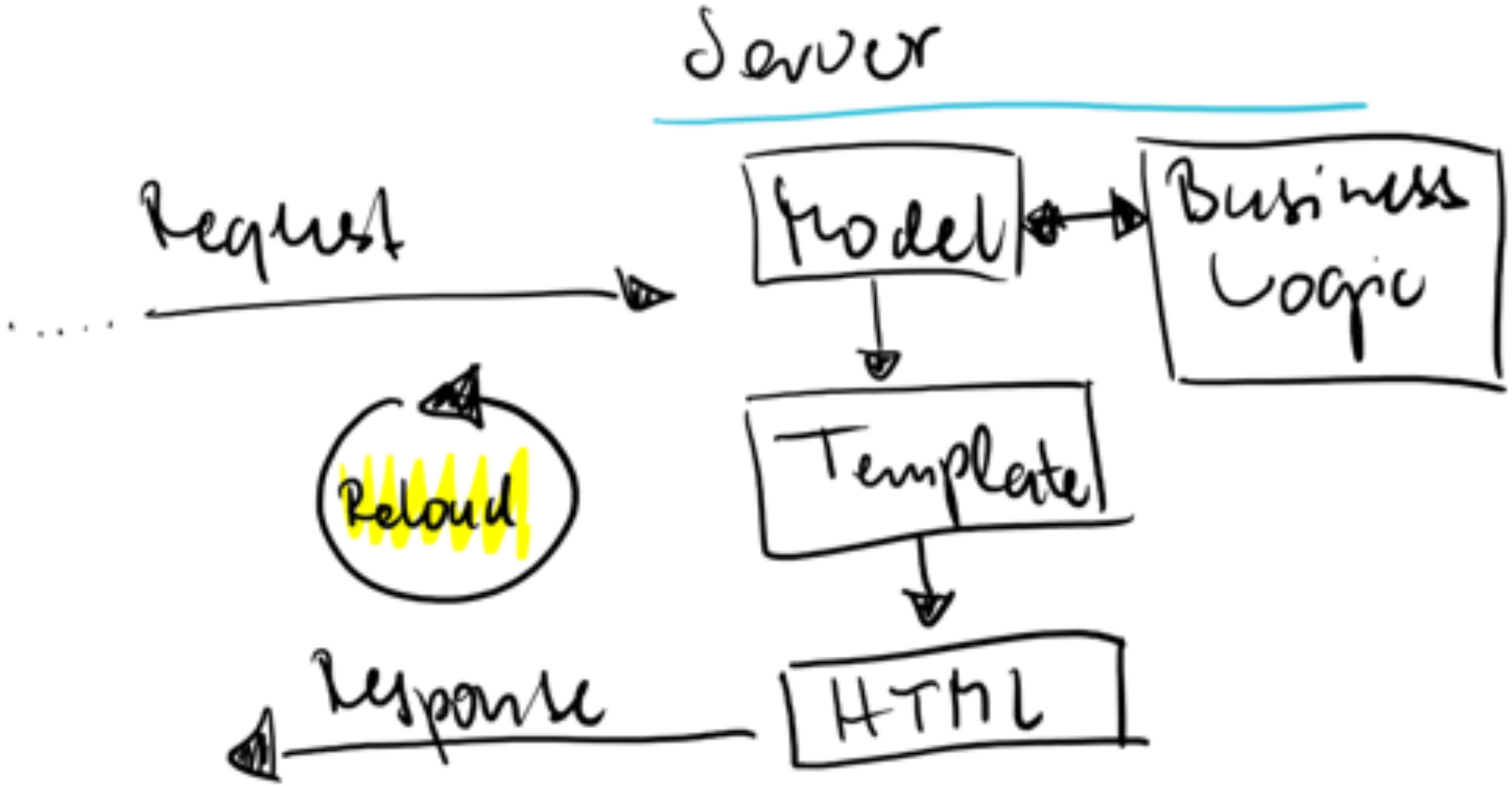
- Was ist bekannt? -> #lecture-2019-05-06
- HTTP-Protokoll – generelles Verständnis (Client/Server, Ports, ...)?
- Unterschied Ports 80 und 443?
- Bekannte HTTP-Header?
- Bekannte HTTP-Status-Codes?
- Debugging-Möglichkeiten im Browser?

- Das nächste Mal frage ich sowas im Slack vor einer Vorlesung ab, daher bitte ab und zu reinschauen ;-) ...

Technische Architektur einer Web-Anwendung



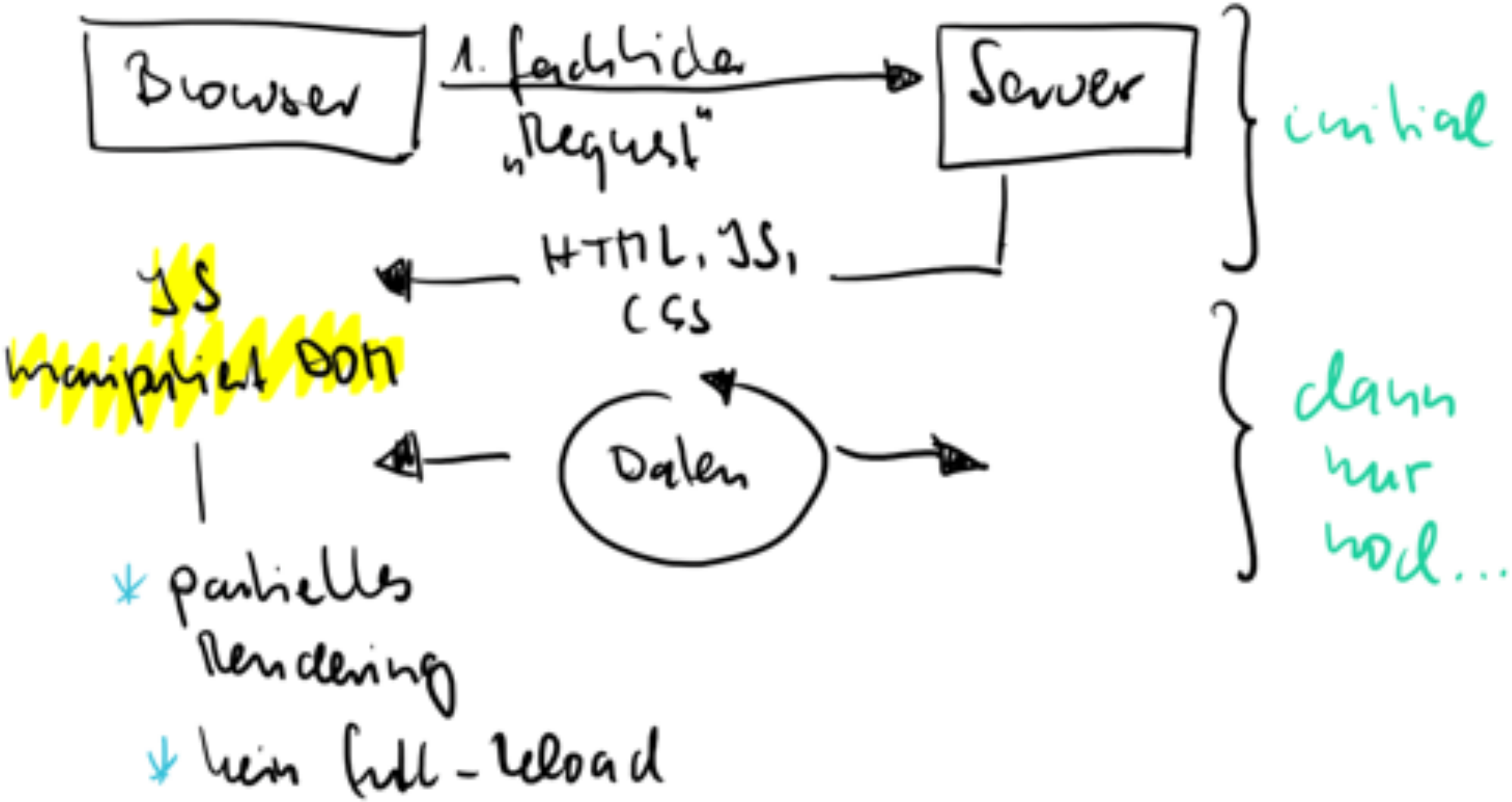
Ansatz: Models + Templates



Ansatz: Models + Templates // Code

Talk is cheap. Show me the code.
- Linus Torvalds (2000-08-25)

Ansatz: SPA


Browser
1. fachliche „Request"
Server
initial
HTML, JS, CSS
JS manipuliert DOM
Daten
dann nur noch...
partielles Rendering
kein full-reload

Tradeoffs ...

- Beide Ansätze (Server-seitiges und Client-seitiges) Rendering haben auch heute noch ihre Daseins-Berechtigung
- Tradeoffs bewusst machen
  - Entwicklungsgeschwindigkeit
  - Browser-Kompatiblität
  - Interaktivität der Seite
  - Seitengröße
  - Leistungsfähigkeit der Clients
  - ...

Code-Walkthrough

- Daten im Backend persistieren
  - Spring-Data JPA (Repositories, Entity, Service, Autowired)
  - Docker für Postgres
- Erste Schritte mit React
  - Grundlegender Aufbau
  - Eine erste Komponente

Übung

- Nachbauen
  - Docker installieren
  - Postgresql starten; siehe Eintrag in README.md
  - Commit für Commit auschecken und verstehen (ggf. fragen!)
- Backend
  - Zusatzmethode: nur die letzten 5 aktuellsten Einträge
    - POJO braucht einen Zeitstempel
    - Zur Vereinfachung: `@Query` Annotation und `nativeQuery = true`
- Frontend
  - Nummerierung und Daten an jedem Post anzeigen
  - Eingabefeld aus letzter Übung einbauen und Post-Eingabe ermöglichen (ohne React!)
  - Click-Handler für Click auf einzelne Zeile mit „Geklickt Ausgabe“ in der Browser-Konsole