PROGRAMMIERPROJEKT 2016

WEB-ANWENDUNGEN

Mathias Weber und Annette Bieniusa

ÜBERBLICK

- Was ist eine Web-Anwendung
- Model-View-Controller Pattern
- Dokumentation mit UML

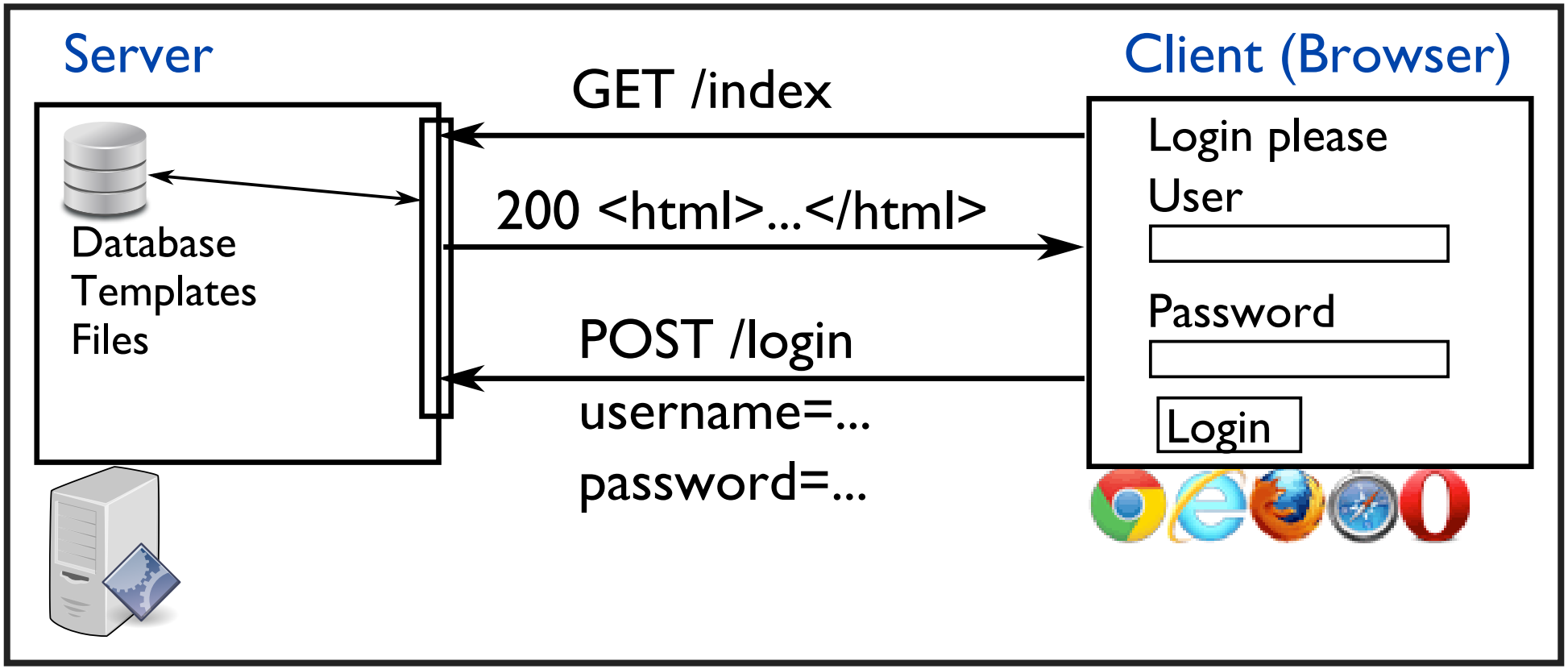
WEB-ANWENDUNG

- HTML ist ein Format um Inhalte einer Seite zu beschreiben
- Inhalte grundsätzlich statisch
- CSS erlaubt Gestaltung der Inhalte, hauptsächlich statisch
- JavaScript erlaubt dynamische Änderungen am HTML
  - Aber: Kein direkter Austausch von Informationen zwischen Benutzern möglich

- Web-Anwendungen laufen auf **Servern** und kommunizieren mit dem Browser des Benutzers
- Austausch von Informationen zwischen Benutzern möglich
- Server **speichert Informationen** zur weiteren Verwendung
- **Client-Server** prinzip

KLASSISCHE SICHT

- Server erzeugt HTML und sendet es an den Browser
- Browser zeigt HTML dem Benutzer an
- Benutzer gibt Informationen in den Browser ein
- Browser sendet Informationen an den Server
- Empfangene Informationen können in Generierung des HTML eingehen

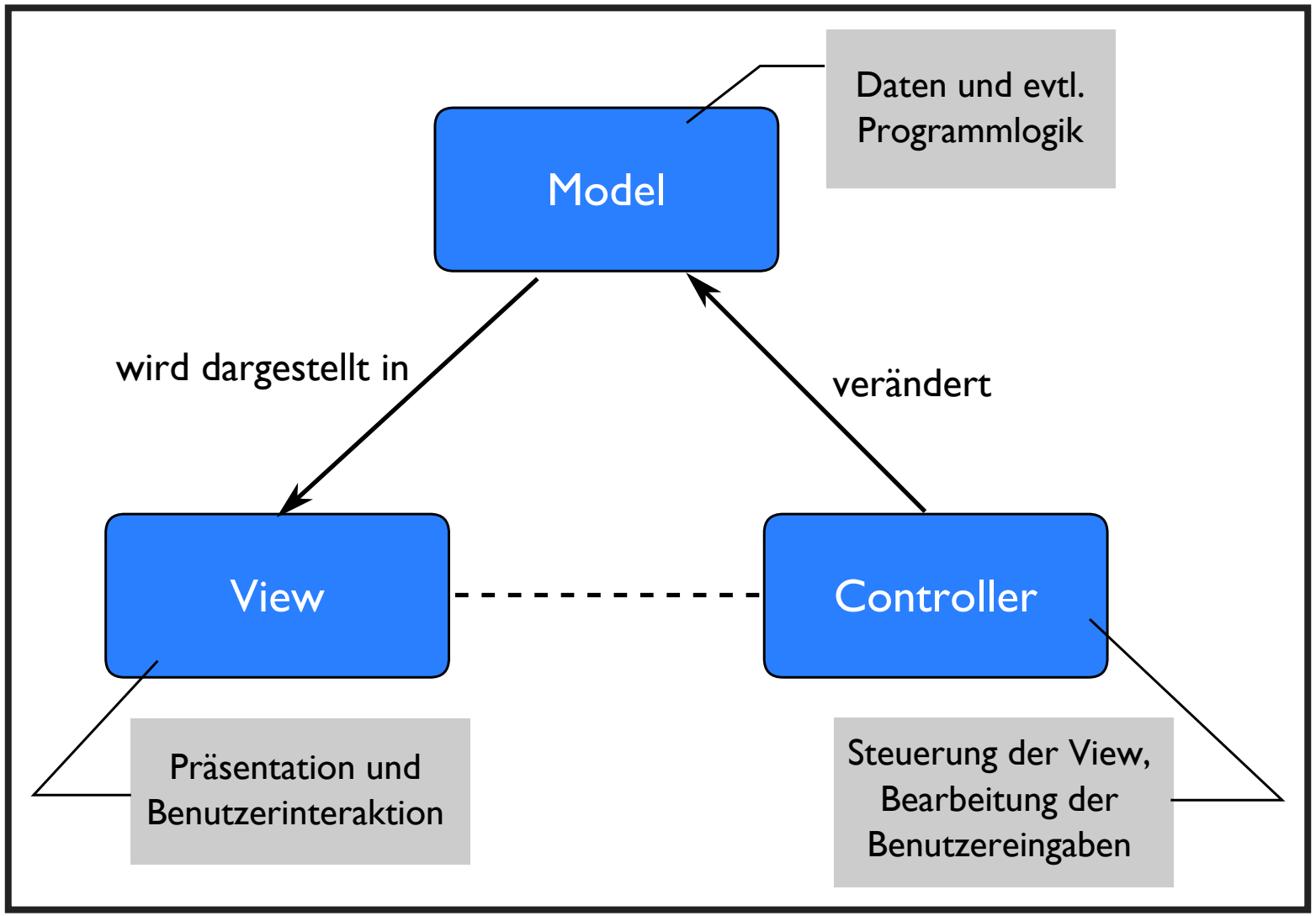

Server
Client (Browser)
GET /index
Login please
User
200 <html>...</html>
Database
Templates
Files
Password
POST /login
Login
username=...
password=...

MODERNE SICHT

- Server stellt Daten als Service zur Verfügung (z.B. REST-Services)
- Browser verarbeitet Daten mit Hilfe von JavaScript Anwendungen (z.B. AngularJS)
- *Wir verwenden hier die klassische Sicht!*

MODEL-VIEW-CONTROLLER

- Muster zur **Strukturierung** von Software
  - Architekturmuster
  - Entwurfsmuster
- **Wiederverwendbarkeit** von Komponenten
  - Bsp.: Austausch der GUI (hier: des HTML-Codes)


Model
Daten und evtl. Programmlogik
wird dargestellt in
verändert
View
Controller
Präsentation und Benutzerinteraktion
Steuerung der View, Bearbeitung der Benutzereingaben

ZUORDNUNG VON FUNKTIONALITÄT

- Geschäfts-/Programm**logik** sollte Teil des Modells sein
  - **Testbarkeit!**
- **Validierung** von Benutzereingaben
  - Einfache Überprüfungen des Formats: View/Controller
  - Komplexere Überprüfungen: Modell
- Daten-**Formatierung** (z.B. Datumsformat)
  - Für unser Projekt: meist Controller

UNIFIED MODELING LANGUAGE (UML)

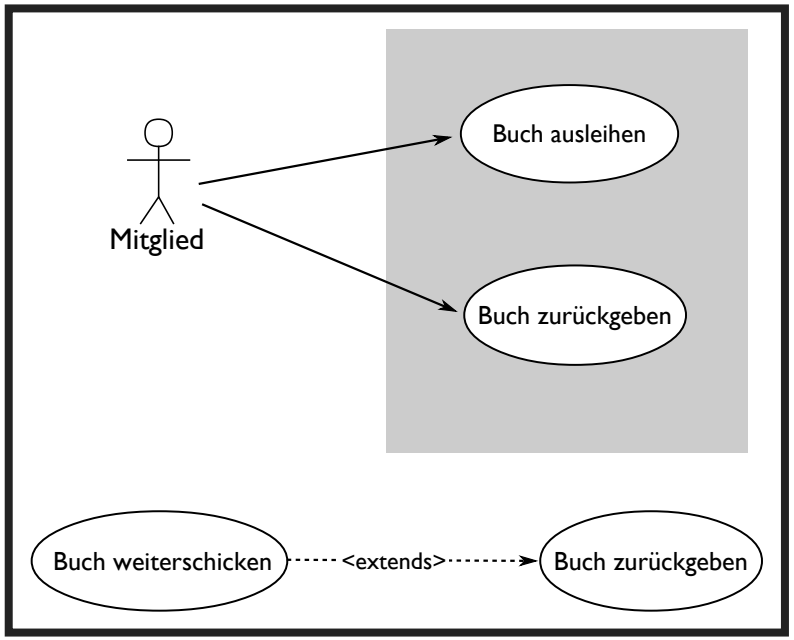
- **Modellierungssprache** zur Spezifikation, Konstruktion und Dokumentation von Software

- **Semantik** diverser Begriffe und Beziehungen zwischen Software-Objekten

- **Grafische Notation** für statische Strukturen und dynamische Abläufe / Verhalten

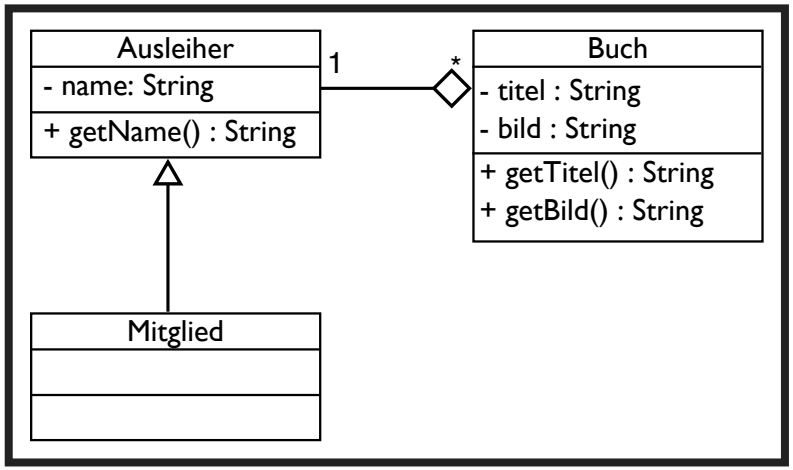
- In dieser Veranstaltung:
  - Nutzerfalldiagramme
  - Klassendiagramme
  - Sequenzdiagramme

NUTZERFALL/ANWENDUNGSFALL-DIAGRAMME



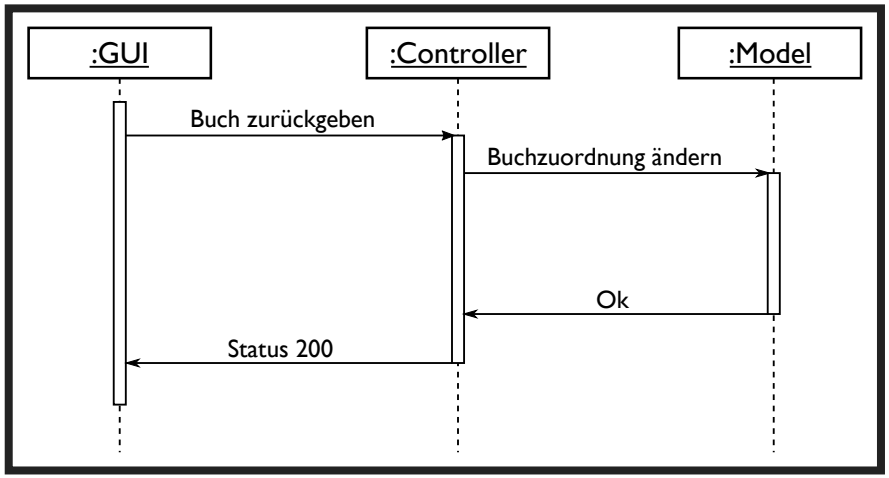
- **Akteure**: Personen, Kunden, Administratoren, andere Systeme, ...
- **Anwendungsfälle** (mit kurzem Beschreibungstext)
- Generalisierung, Extends-/Includes-**Beziehung** (auch mit Bedingung)

KLASSENDIAGRAMME



- Klassen: Attribute und Operationen
  - Sichtbarkeit (+ public, - private)
  - ordered, read-only, ...
- Generalisierung: gerichtete Beziehung zwischen generellen und spezielleren Klassen
- Assoziationen: Relation zwischen Klassen
  - Multiplizitäten
  - Komposition (ausgefüllt)/Aggregation
- Schnittstellen

SEQUENZDIAGRAMME



- (A-)Synchrone Aufrufe
- Lebenslinien von Objekten
- Aktivierungsbalken: Objekte verfügen über den Kontrollfluss