Web-Engineering II

Prof. Dr. Sebastian von Klinski

Front-end Bibliotheken und Frameworks

- Für Umsetzung von Front-end stehen zahllose Frameworks und Bibliotheken zur Verfügung
- Sehr verbreitet…
  - **Programmiersprachen, die JavaScript erzeugen**
    - Code wird zu JavaScript kompiliert und im Browser ausgeführt
  - **Web-Frameworks**
    - Unterstützung beim dynamischen Aufbau der Oberfläche
    - Verwalten des Anwendungsstatus
    - Kommunikation mit dem Back-end
    - Spezifische Tags/ Definition von eigenen Tags/ Skriptsprachen/ etc.
  - **CSS-Bibliotheken**
    - Grafische Gestaltung der Webanwendung
    - Vordefinierten CSS-Styles
    - JavaScript-Files
- In der Regel Einsatz von Kombination dieser Gruppen

Programmiersprachen, die JavaScript erzeugen

Programmiersprachen, die JavaScript erzeugen

- Nicht viele Optionen auf dem Markt
  - TypeScript (nahezu objektorientierte Sprache)
  - Brython
    - Zusammengesetzt aus Browser + Python
    - Erlaubt das clientseitige Scripten in Python
  - JSX (JavaScript XML) über Babel (Trans-Kompiler)
    - Erlaubt die Einbettung von XML in JavaScript-Code

Quelle: https://de.wikipedia.org/wiki/Clientseitige_Skriptsprachen

TypeScript

- 2012 von Microsoft entwickelt Programmiersprache
- Konstrukte wie Klassen, Interfaces, Vererbung, Module und anonyme Funktionen
- Aus TypeScript heraus kann auch JavaScript verwendet werden (z.B. jQuery)
- TypeScript-Code wird zu JavaScript-Code kompiliert
- Installation

```
> npm install -g typescript
```

- Übersetzen von TypeScript-Datei (*.ts)

```
tsc greeter.ts
```

- Ergebnis ist JavaScript-Datei

TypeScript

Attribute im Konstruktor definiert

- Klassen

```
class Student {
    fullName: string;
    constructor(public firstName: string, public middleInitial: string, public lastName: string)
    {
        this.fullName = firstName + " " + middleInitial + " " + lastName;
    }
}

function greeter(person: Person) {
    return "Hello, " + person.firstName + " " + person.lastName;
}

let user = new Student("Jane", "M.", "User");
…
```

TypeScript

- Erzeugter Code
- Klassen in TypeScript werden auf Funktionen in JavaScript übertragen

```
var Student = /** @class */ (function () {
    function Student(firstName, middleInitial, lastName) {
        this.firstName = firstName;
        this.middleInitial = middleInitial;
        this.lastName = lastName;
        this.fullName = firstName + " " + middleInitial + " " + lastName;
    }
    return Student;
}());
function greeter(person) {
    return "Hello, " + person.firstName + " " + person.lastName;
}
var user = new Student("Jane", "M.", "User");
document.body.textContent = greeter(user);
```

TypeScript

- Einbettung in HTML-Seite

```
<!DOCTYPE html>
<html>
    <head><title>TypeScript Greeter</title></head>
    <body>
        <script src="greeter.js"></script>
    </body>
</html>
```

TypeScript

```
interface Person {
    firstName: string;
    lastName: string;
}
function greeter(person: Person) {
    return "Hello, " + person.firstName + " " + person.lastName;
}
let user = { firstName: "Jane", lastName: "User" };
document.body.textContent = greeter(user);
```

- Viele Abweichungen von richtiger objektorientierter Sprache wie Java
  - Definition von Attributen im Konstruktur
  - Interfaces haben State (Attribute)
  - Objekte können ohne Casting als Interface interpretiert werden, wenn es entsprechende Attribute gibt, etc.

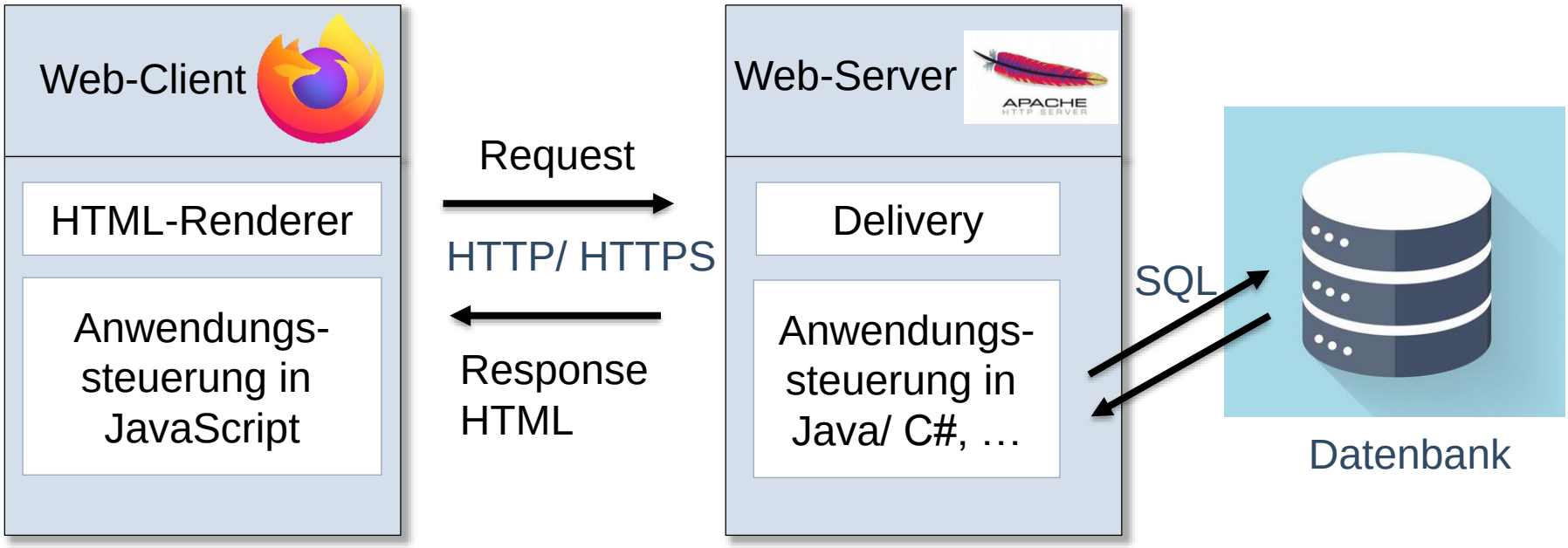
TypeScript

- Syntax eine Mischung von Java und JavaScript
- Umsetzung von Webseiten direkt mit TypeScript eher selten
- Häufig in Kombination mit Web-Framework (Angular, Vue.js)

Web-Frameworks

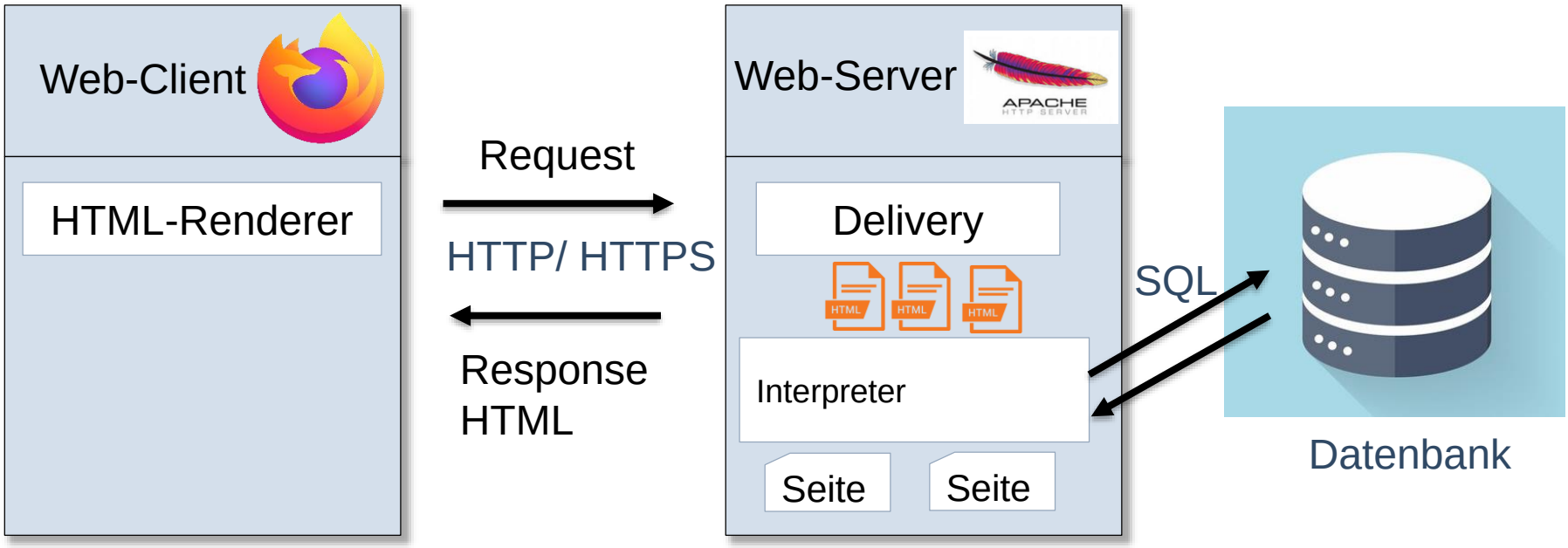
Web-Frameworks

- Unterscheidung in serverseitig/ clientseitig (meist leichte Überschneidungen)
- Bezeichnung beschreibt Ort der Anwendungssteuerung
- Serverseitig: Anwendungssteuerung in der Regel in Java/ C# (.Net) auf Server
- Clientseitig: Anwendungssteuerung in JavaScript im Browser



Serverseitige Web-Frameworks

- Server interpretiert Skript-Tags → Kompiliert zu HTML-Seiten → Browser
- Interaktionen werden von Client an Server gesandt
- Server bearbeitet Anfragen und schickt Antwort als neue Seite
- Beispiele: PHP, ASP.net, JSP
- Heutzutage eher selten verwendet

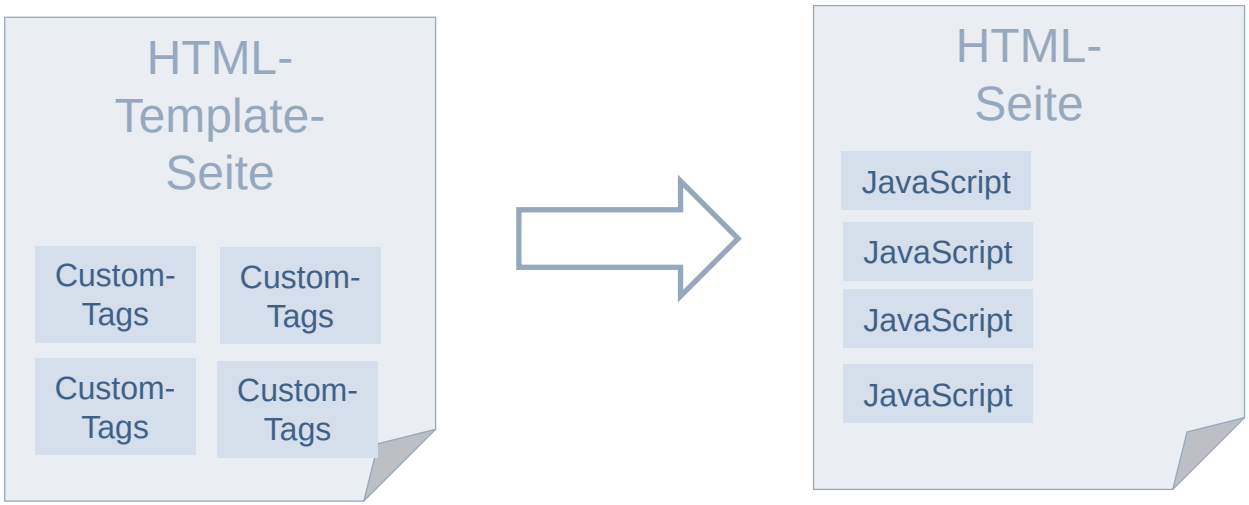


Der Client „Browser“

- Im Browser können die folgenden Sprachen ausgeführt werden
  - JavaScript
    - 1995 von Netscape entwickelt
    - Zum Auswerten von Benutzerinteraktionen und um Inhalte zu verändern, nachzuladen oder zu generieren
    - Möglichkeiten von HTML und CSS erweitern
  - VB-Script
    - Von Microsoft entwickelte Skriptsprache
    - Kann auf COM-Komponenten unter Windows zugreifen
    - In Microsoft-Browsern unterstützt
  - ActionScript
    - Programmiersprache von Adobe
    - Für Adobe Flash, Flex oder Air entwickelt
- Praktisch kommt heute nur noch JavaScript zur Anwendung

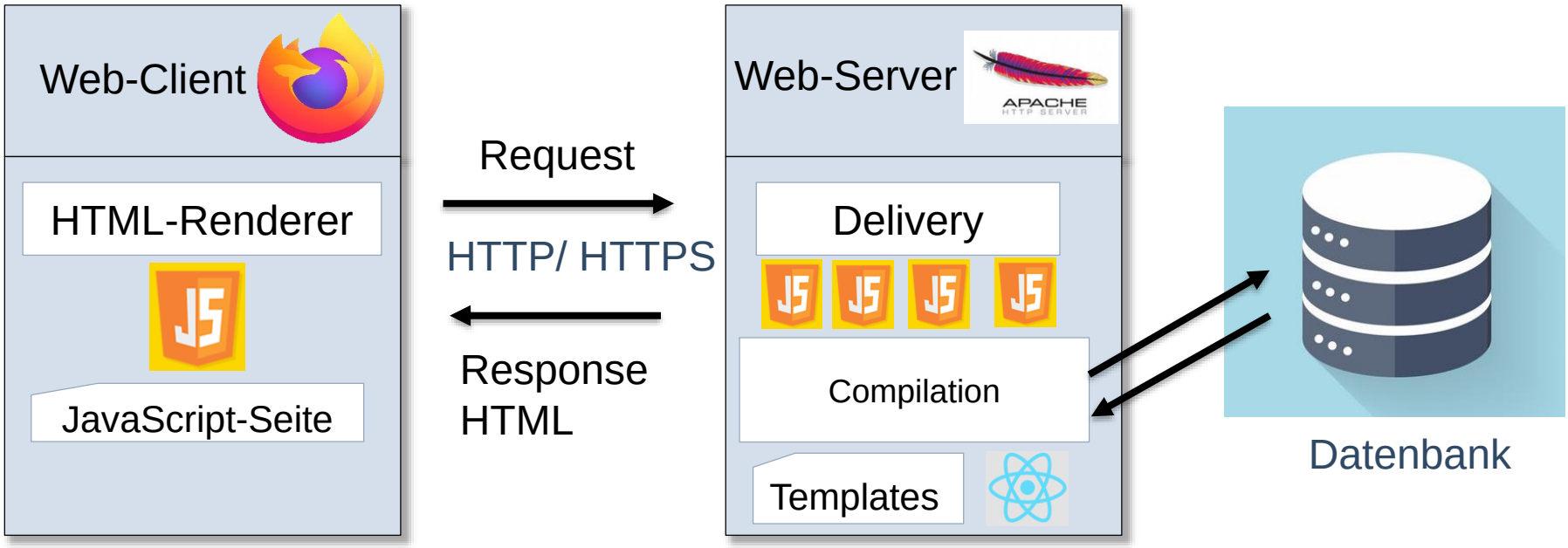
Clientseitige Web-Frameworks

- Clientseitige Web-Frameworks erzeugen durch Kompilation JavaScript-Code, der dann im Browser ausgeführt wird
- Verbreitete Web-Frameworks
  - React, Angular, Vue.js
- Der JavaScript-Code wird in der Regel durch Kompiler bei der Entwicklung oder zur Laufzeit durch Kompiler im Browser erzeugt (z.B. bei Angular)



Clientseitige Web-Frameworks

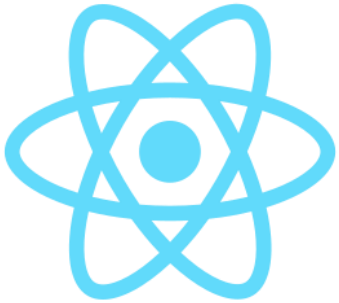
- Web-Server liefert HTML-Seiten mit JavaScript
- JavaScript-Code wird im Browser ausgeführt
- Interaktionen werden in JavaScript bearbeitet
- Ggfls. Kommunikation mit dem Server per AJAX
- Rendering der Änderungen in Seite im Browser durch DOM-Manipulation



Clientseitige Web-Frameworks

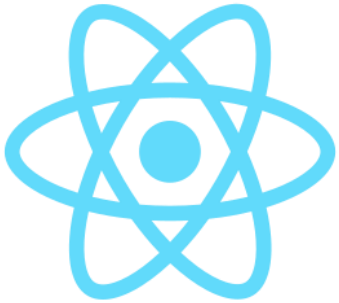
- Fokus der clientseitigen Frameworks
  - Dynamische Anpassung der Webseite
  - Einfache AJAX-Einbindung
  - Automatische, selektive Aktualisierung der Seite
  - Umgang mit dem State auf dem Client
- Beispiele: React, Angular, Vue

React

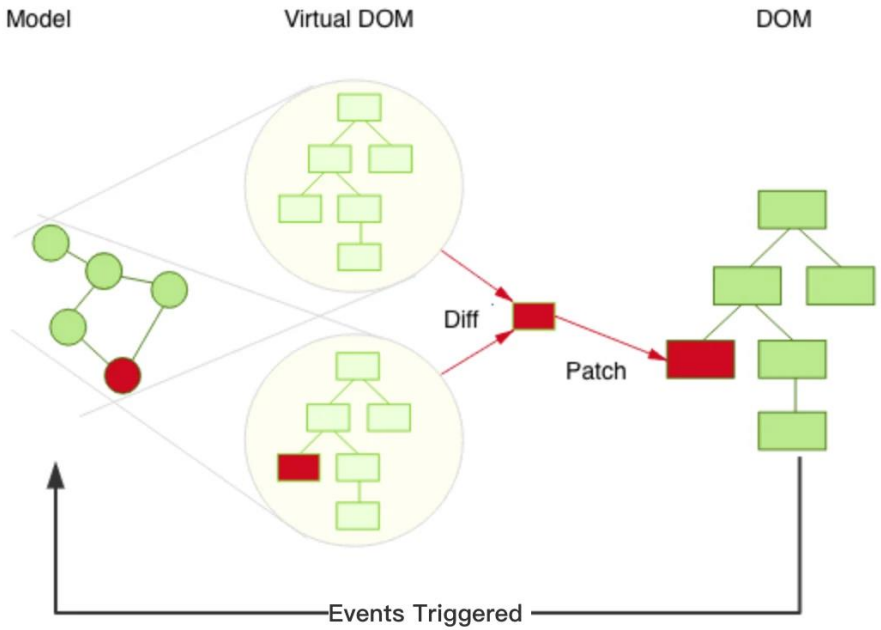


- JavaScript-Softwarebibliothek
- Entwickelt von Facebook
- Für Multi- und Single-Page-Anwendungen
- Komponenten werden als selbst definierte HTML-Tags repräsentiert
- Häufig für die Umsetzung von Single-Page-Applications
- Open Source Projekt
- Kritikpunkt:
  - Vermischung von Rendering und Logik
  - Verstoß gegen MVC-Paradima
  - Ist das in der Praxis ein Problem?
- Einsatz bei:
  - Facebook, Twitter, Netflix, Airbnb, PayPal, The New York Times, Yahoo, Walmart, Uber und Microsoft

React: Virtual DOM



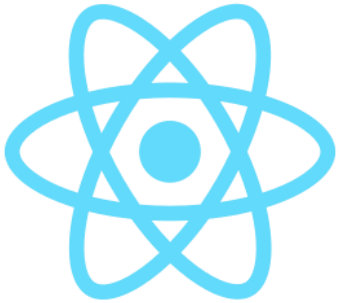
- Aktualisieren des DOM aufwändig
- Schaffung von Virtual DOM
- DOM-Diffing: selektive Aktualisieren des DOM auf Basis eines Vergleichs zwischen ursprünglichem und geändertem Virtual DOM
- → im DOM wird nur der wirklich geänderte Teil aktualisiert



Virtual DOM ist Grundlage für deutlich bessere Performance als bei anderen Web-Frameworks

Quelle: https://x-team.com/blog/react-vs-angular/

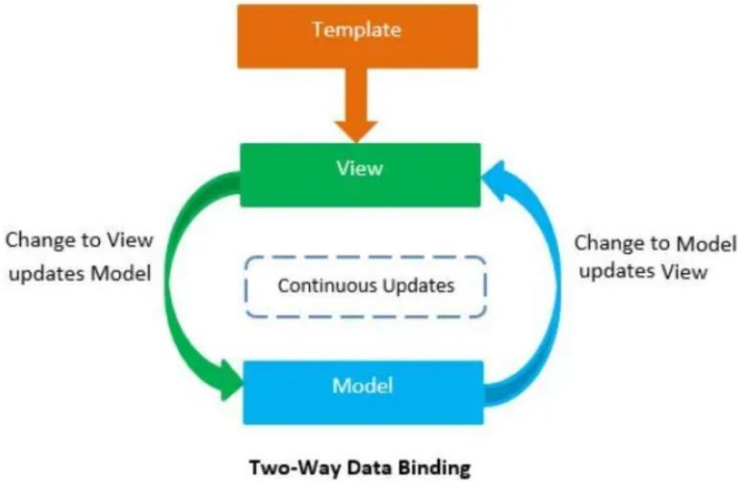
React Native



- Native mobile Applikationen
- Nur leicht andere Syntax als bei React
- Exportieren auf verschiedene Plattformen (Android, IOS, etc.)
- Auch Einbinden von nativen Komponenten in Objective-C, Java, or Swift.

Angular

- Auf TypeScript-basiertes Front-End-Webapplikationsframework
- Für Multi- und Single-Page-Anwendungen
- Entwickelt von Google
- Open-Source-Software
- Vollständiges MVC-Framework
- Bidirektionales Binding zwischen View und Model
- Konzipiert für Umsetzung von
  - Progressive Web-Apps
  - Native Mobile-Apps
  - Desktop-Appliation



Quelle: https://de.wikipedia.org/wiki/Angular

Angular

- Kompilation
  - JIT-Mode: Source-Code und Compiler werden im Browser geladen
  - AOT-Mode (Ahead-Of-Time Compilation): Übersetzung während der Entwicklung, Browser erhält JavaScript
- Native mobile Applikation
  - Vewendet NativeScript
  - Nicht wirklich „nativ“, HTML in einem Mobile-App-Container
    - Verwendet Ionic
    - Ionic ist ein Open-Source-Webframework zur Erstellung von Hybrid-Apps und Progressive Web Apps auf Basis von HTML5, CSS, Sass und JavaScript/TypeScript.
    - Ionic verwendet Apache Cordova Container, in dem Web-Anwendungen laufen
- Einsatz bei
  - McDonald’s, AT&T, HBO, Apple, Forbes, Adobe, Nike und Microsoft

Quelle: https://de.wikipedia.org/wiki/Angular

Vue.js

- Clientseitiges JavaScript-Webframework zum Erstellen von Single-Page-Webanwendungen
- „Nur Kenntnisse in JavaScript und HTML notwendig“

```
Vue.component('my-component', {
  template: '<div>Dies ist eine neue Komponente</div>'
})
new Vue({
  el: '#example'
})
```

```
<div id="example">
  <my-component></my-component>
</div>
```

```
<div id="example">
  <div>Dies ist eine neue Komponente</div>
</div>
```

Quelle: https://de.wikipedia.org/wiki/Angular

Web-Frameworks

- Bei Vergleichen zwischen Angular und React schneiden beide häufig weitgehend gleich ab
  - React hat deutlich größere Community
  - Performance bisher bei React besser (durch virtuellen DOM)
- Vue.js
    - Noch recht jung
    - Von keinem großen Unternehmen gestützt
    - Bisher nur bei wenigen großen Unternehmen im Einsatz: Alibaba und Baidu
    - Entwicklung werden die kommenden Jahre zeigen

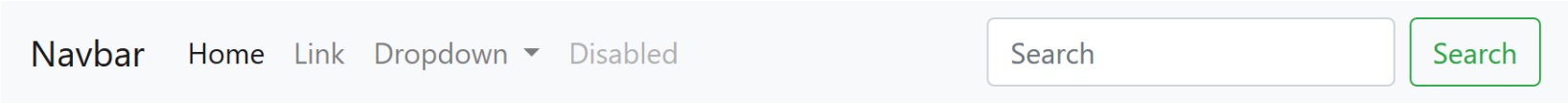
CSS Bibliotheken

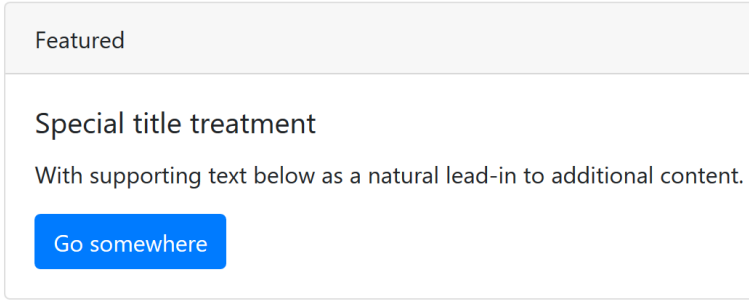
CSS-JavaScript Bibliotheken

- Aufgabe von Web-Frameworks wie React, Angular, Vue.js ist Aufbau der Anwendung und Verknüpfung mit State sowie Back-end
- Layout und grafische Gestaltung über CSS umgesetzt
- Für Gestaltung per CSS gibt es zahlreiche Bibliotheken
- Erleichtern die Umsetzung von klassischen HTML-Komponenten über fertige CSS und JavaScript-Bibliotheken
- Vereinfachen insbesondere vollständig responsive Umsetzung von Seiten
- Beispiele:
  - Bootstrap, Semantic-UI, Foundation, Materialize, Material UI, etc.
- Am stärksten verbreitet: Bootstrap

CSS-JavaScript Bibliotheken

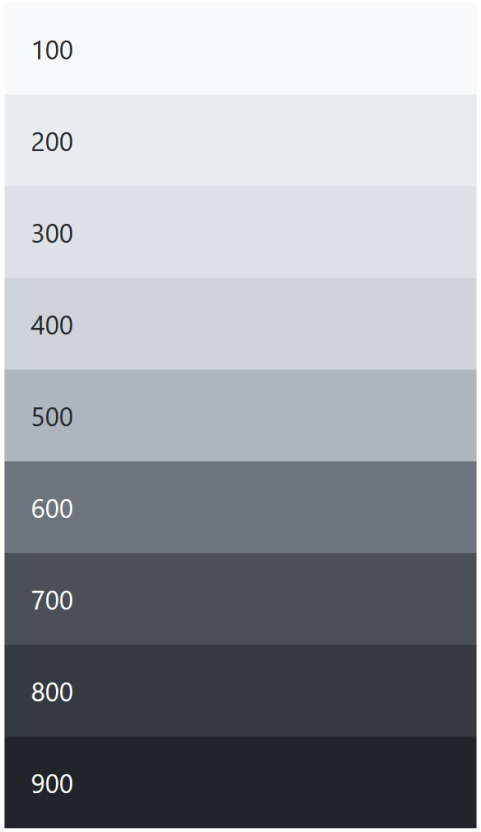
- Vordefinierte Aspekte
  - Grid-System: erweiterte Version von Flexbox
  - Buttons, Badges, Cards, Menus, Tab-Reiter, etc.





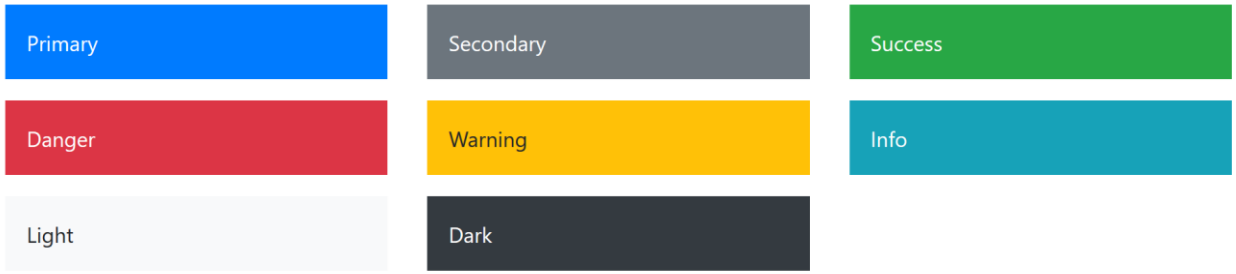
CSS-JavaScript Bibliotheken

- Farb- und Grauwertschema




```
.custom-element {
  color: gray("100");
  background-color: theme-color("dark");
}

a {
  color: var(--blue);
}
```



CSS-JavaScript Bibliotheken

- Einsatz von CSS-Bibliotheken erspart umfangreiche grafische Optimierungen
- Setzen auf Best-Practices auf
- Basieren auf umfangreichen Studien zur Software-Ergonomie
- Erleichtern insbesondere Umsetzung von vollständig responsiven Webanwendungen