Web Engineering 2 - HS2021 marco.agostini@ost.ch

1 NodeJS

Was muss ein Webserver können: HTTP Anfragen annehmen, Actions ausführen basierend auf URL, HTTP Antworten senden. **Eigenschaften:** JS runtime, event-driven non-blocking I/O model, NPM.

2 ExpressJS

2.1 Middleware

2.2 Cookie

Der Server sendet die Cookies, welcher der Client dann speichert und immer mitsendet. Das Secret wird für die Signierung benötigt.

2.3 Session

Eine Session benötigt Cookies als Wiedererkennungsmerkmal des Client. Kann HTTP-Stateless umgehen (Wiederspricht REST).

2.4 JSON Web Token (JWT)

Ziel: Stateless Sever. Daten: Ausgestellt für, Ablauf-Datum, Signatur. Nach Authentisierung wird dem Client JWT gesendet und nachher immer wiederverwendet (Autorisation Header).

2.5 Same-Origin Policy (SOP)

Verhindert die Datenverarbeitung von anderen Quellen. **Origin:** host, protocol, port.

2.6 Cross-Origin Ressource Sharing (CORS)

Ermöglich Datenverarbeitung von anderen Quellen (SOP). Integriert in fetch.

3 ECMAScript

Wird vom ECMA Komitee entwickelt und spezifiziert. Ab ES2015/ES6 jährlich eine neue Version. Jedes Feature muss vier Stages durchlaufen. **Polyfill:** Neues Feature für altem Standard geschrieben. **Transpilling:** Ändert Code auf alten Standard für Support.

4 TypeScript

Ist ein Pre-Processor, der JavaScript generiert. **Features:** Static Typing. **Basistypten:** boolean, number, string, null, undefined, (any). Variablen mit unkonw kann alles zugewiesen werden.

```
let myVar: any = 1;
let myString: string = 'abcd';
```

4.1 Komplexe Typen

Array, Tupels, Enums, String Literal Union Types

4.2 Funktionsdeklaration

Funktionsparameter können optional sein.

```
function add(s1: string, s2: string): string;
function add(s1: string, s2?: string): string;
```

4.3 Klassen

```
class Counter {
  #doors: number; //private class field
  public static readonly WOOD_FACTORS = {'oak': 80};
  // Konstruktor Code wird automatisch generiert
  constructor(public make:string, public colors: string);

}
```

4.4 Interfaces

Interfaces (1 pro Klasse) können in Deklaration von Klassen genutzt werden.

4.5 Type Assertion

```
const myCanvas = document.getElementById("main_canvas
    ") as HTMLCanvasElement;
```

4.6 Type Operator

Mit dem Keyword lässt sich ein Type aus einem anderen Typ ableiten.

```
type Point = { x: number, y: number };
type P = keyof Point;
```

4.7 Template Literal Types

```
type Shipper = 'UPS' | 'FEDEX' | 'DHL';
type TrackingType = 'Overnight' | ' Priority ' | 'Economy';
type ShipmentMethod = `${Shipper}-${TrackingType}`;
```

4.8 Generics

5 Responsive Layout

Flexibles Layout: Dynamisches (grössenadaptives) Layout welches sich ohne Media-Queries umsetzen lassen. **Reponsives Layout:** Dynamisches Layout welches für unterschiedliche Geräte, Bereiche von Display-Grössen und unterschiedliche Medien separates Layouts definiert.

5.1 Media Queries

Kann fast alle Einheiten verwenden (em=best Accessibility). **Typische Triggerpunkte:** 480px, 768px, 992px.

```
@media screen { [width|min-width|max-width]:700px) {}
@meida (min-width: 20em) and|not (max-width: 30em) {}
```

5.2 ViewPort

Mobile Geräte benötigen Deklaration des Viewports für Media Queries.

5.3 CSS

5.3.1 calc()

Kann verschiedene Werte direkt im CSS verrechnen (Ausser Media Queries).

5.3.2 min(), max()

Vergleich mehrere Werte und gibt den kleinsten/grössten zurück.

5.3.3 clamp()

min val, actual size, max val.

5.4 Floats

Nicht mehr aktuell und nicht mehr zu verwenden.

5.5 Custom Attributes

```
--color-light-red: #ffeaea; // variable
--color: var(--color-dark-violet); // reference
```

5.6 Flexbox

Flex wird auf dem Container aktiviert und betrifft alle direkten Kind-Elemente.

5.7 Grid

6 Accessibility

Stolze Regeln: Bilder mit Alt-Text, Keine Informationen ausschließlich in Farbe dargestellt, Vorder- und Hintergrund bei reduzierter Farb- und Kontrastwahrnehmung in Standardansicht deutlich unterscheidbar, Skalierung der Schrift über Funktionen des Browsers möglich (arbeiten mit: em/rem),

6.1 Farbdesign

Die Farben für Farbenblinde simulieren, um unerkennbare Unterscheidungen zu vermeiden. Mehrfach-Codierungen verwenden: Fabre und Form, etc. **Farbkontraste:** (Wichtig für 50+), Level AA=4.5:1, AAA=7:1.

6.2 Zoombarkeit

Emfpehlung: Nicht unberbinen.

6.3 Animationen

Empfehlung: sollten abstellbar sein. Verhinderung der Auslösung von Epilepsie und Migräne.

6.4 Medien

Sollten immer einen Alternativen-Text haben.

6.5 Screen Reader

Keine Heading-Levels auslassen, Semantic Elements richtig nutzen, Skip-Links am Anfang der Seite.

7 Security

7.1 A01 Broken Access Control

7.2 A2 Cryptographic Failures

7.3 A03 Injection

7.4 A07 Identification and Authentication Failures

8 Testing

9 Animation

10 Internationalization

11 UX-Research

12 WebDevOps