Browser und Rendermechanik

Browser

- DER Client im Web
- Human Machine Interface für Webanwendungen
- Laufzeitumgebung für Clientseitige Sprachen
  - JS, HTML, CSS und andere
- Oft zentralste Software für Nutzer
- Wichtige Komponenten sind
  - Schnittstellen
  - Entwicklertools
  - Render Engines
  - Laufzeitumgebungen

Browser API - Übersicht

- Manipulation von HTML (DOM)
- Manipulation von Medien (Canvas, WebGL)
- ...
- Erlauben Zugriff über
  - Javascript
  - HTML
  - CSS
  - HTTP Protokoll
  - Betriebssystem

Browser API - Javascript Integration

- Viele API können über Javascript genutzt werden
- Integration über Javascript Klassen
- Erlauben sehr aufwändige Anwendungsfälle
- Erlaubt Reaktion auf Eingaben

>> https://www.w3schools.com/tags/tryit.asp?filename=tryhtml5_ev_onkeyup2

>> https://www.w3schools.com/html/tryit.asp?filename=tryhtml5_canvas_tut_path2

>> https://www.w3schools.com/html/tryit.asp?filename=tryhtml5_geolocation

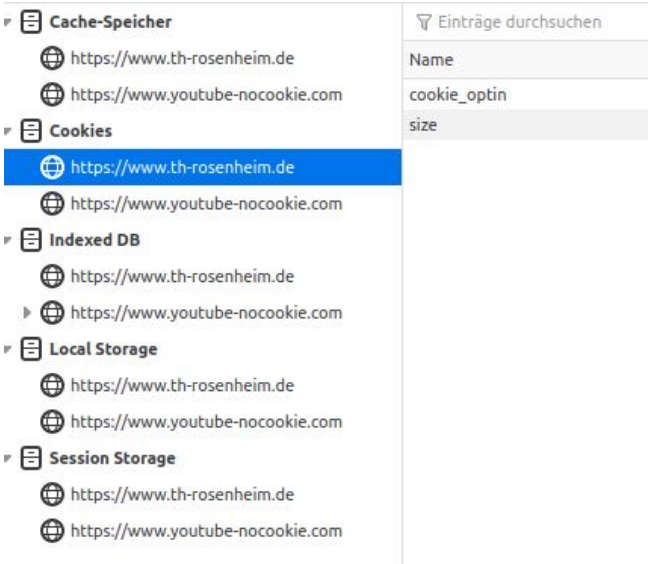
>> https://www.w3schools.com/html/tryit.asp?filename=tryhtml5_sse

Browser API - zum Betriebssystem

- Liefert Webseiten Informationen über Hardware und Betriebssystem
  - User-Agent Header, “Locale”
- Erlaubt Steuerung von Hardware des Systems
  - Vibrations-API
  - Geolocation (GPS Sensor)
- Zugriff aufs Dateisystem
  - Nur sehr beschränkt möglich!
    - Z.B. bewusste Auswahl von Dateien zum Upload

Browser API - Lokale Speicheroptionen

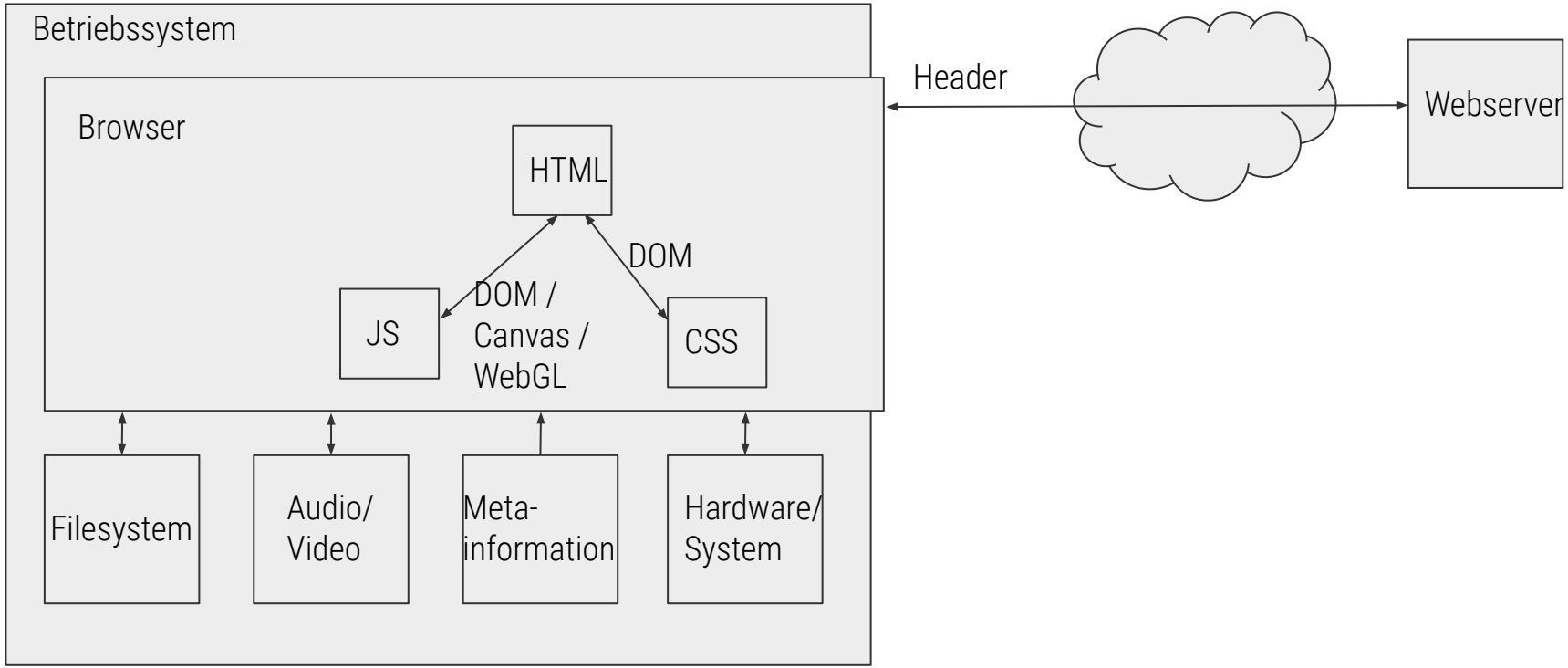
- Mehrere Optionen lokal Informationen abzulegen
- Klar getrennt nach Webseiten (Domains)
- Spezialisiert nach Anwendungszweck
  - Bieten unterschiedliche Lebensdauer
    - Session Lifetime von Session Storage
  - Bieten unterschiedliche API
    - ZQuery Funktion von Indexed DB
    - Request Bundling von Cookies
    - …
  - Unterschiedliche Größenbeschränkungen
  - …
- Nutzung kommt auf Verwendungszweck an!



Browser API - zu HTML/CSS

- Erlauben (simple) Reaktion auf Eingaben
  - Z.B. “:hover” CSS Selektor
  - >> https://www.w3schools.com/cssref/tryit.asp?filename=trycss_sel_hover
- Integration von Mediendateien
- Geräteinformation
  - Z.B. Gerätetyp über Mediaqueries
  - >> https://www.w3schools.com/cssref/tryit.asp?filename=trycss3_media_bg

Browser API - Übersicht

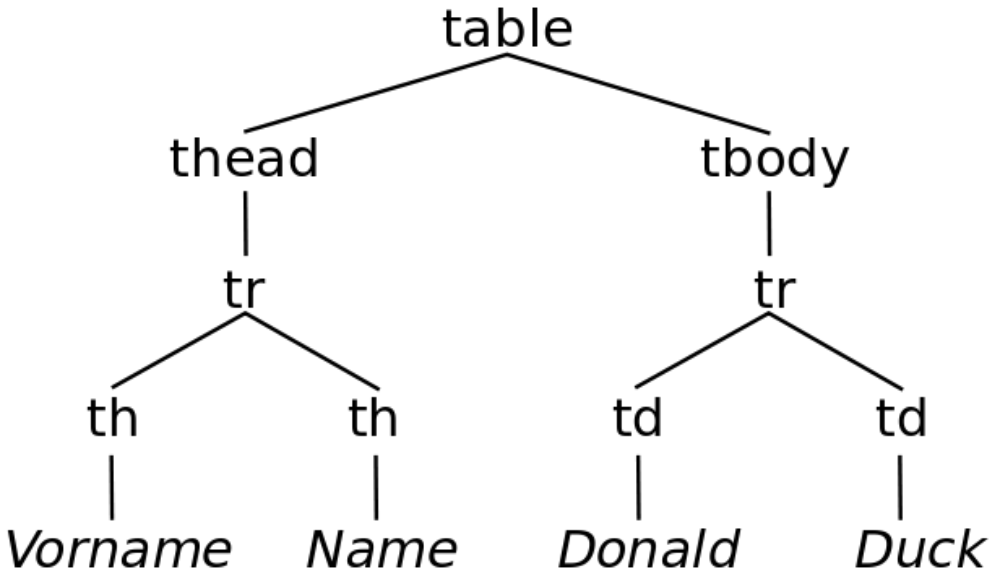


Browser API - DOM

- **D**ocument **O**bject **M**odel
- Programmierschnittstelle zum einheitlichen Zugriff auf HTML Elemente
  - Zu Manipulation durch Javascript
  - Auch in XML genutzt
- Stellt Struktur als abstrakten Baum bereit
- Hat auch JS Repräsentation von Elementen

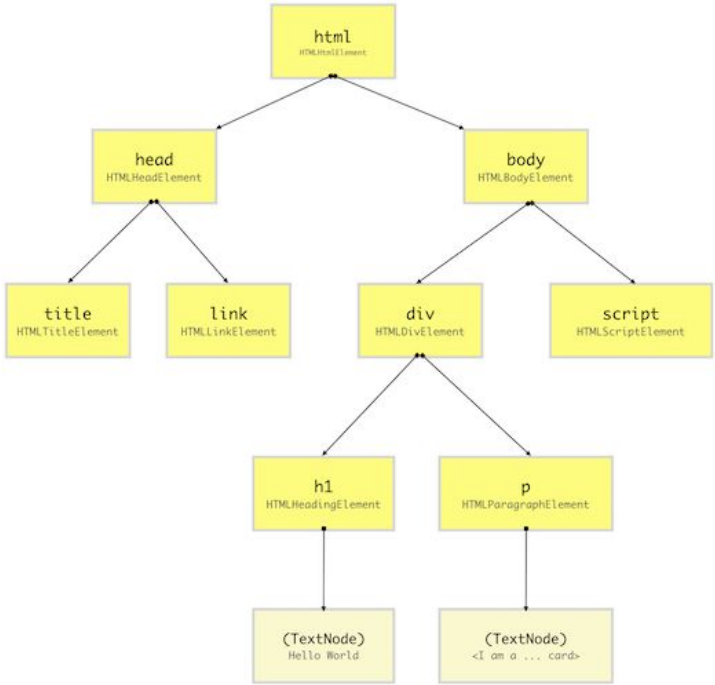
Browser API - DOM

```
<table>
  <thead>
    <tr>
      <th>Vorname</th>
      <th>Name</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Donald</td>
      <td>Duck</td>
    </tr>
  </tbody>
</table>
```



Browser API - DOM

- Aufgeteilt in sog. Nodes
- HTML Attribute und Inhalte haben Nodes im DOM
- DOM kann ein logische Bereiche geteilt sein
  - sog. Shadow DOM
  - Erlaubt das “Einsperren” von DOM Teilen
- Aber, nicht der ganze DOM wird gerendert
  - Abhängig von Anzeigeoptionen im CSS

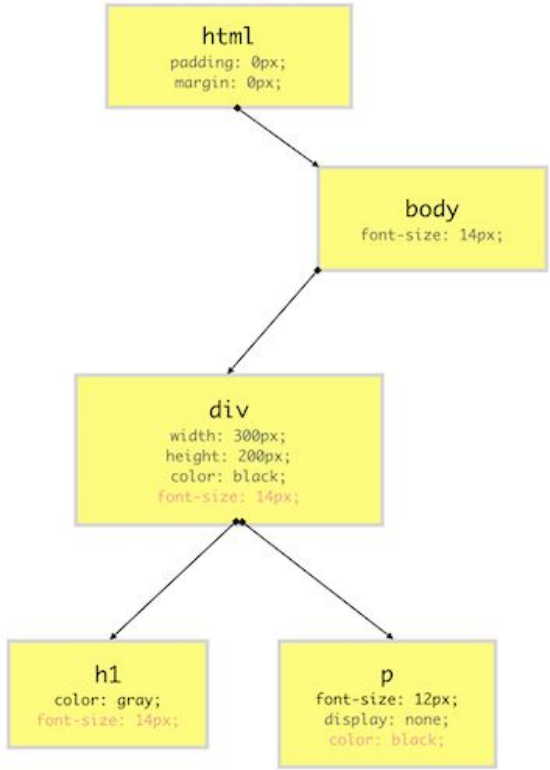


Render Prozess

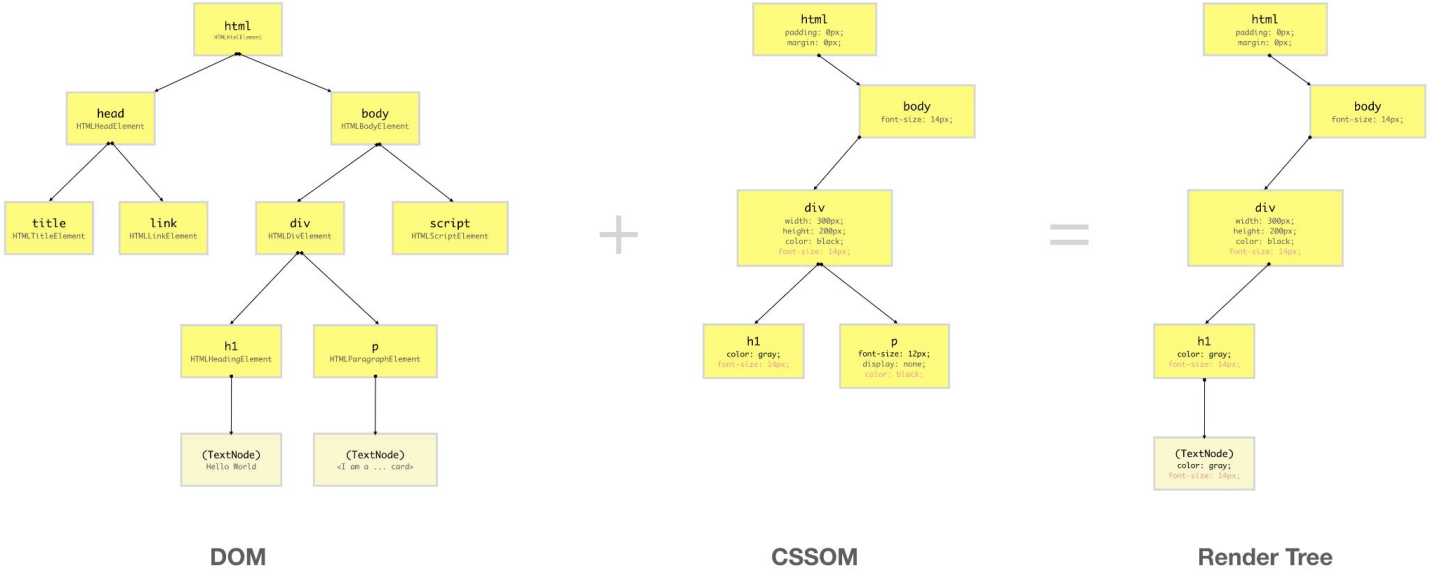
- Rendering ist ein mehrstufiger Prozess
- Hängt maßgeblich von HTML und CSS ab
  - (DOM und CSSOM)
- Wird jedes mal wiederholt bei Änderungen
- Hochoptimiert
  - Starke kategorisierung von Inhalten um Aufwand zu sparen
    - Layering, Tiling
- Für jeden Browser etwas anders
  - Abhängig von Render Engine und JS Laufzeitumgebung

Render Prozess - CSSOM

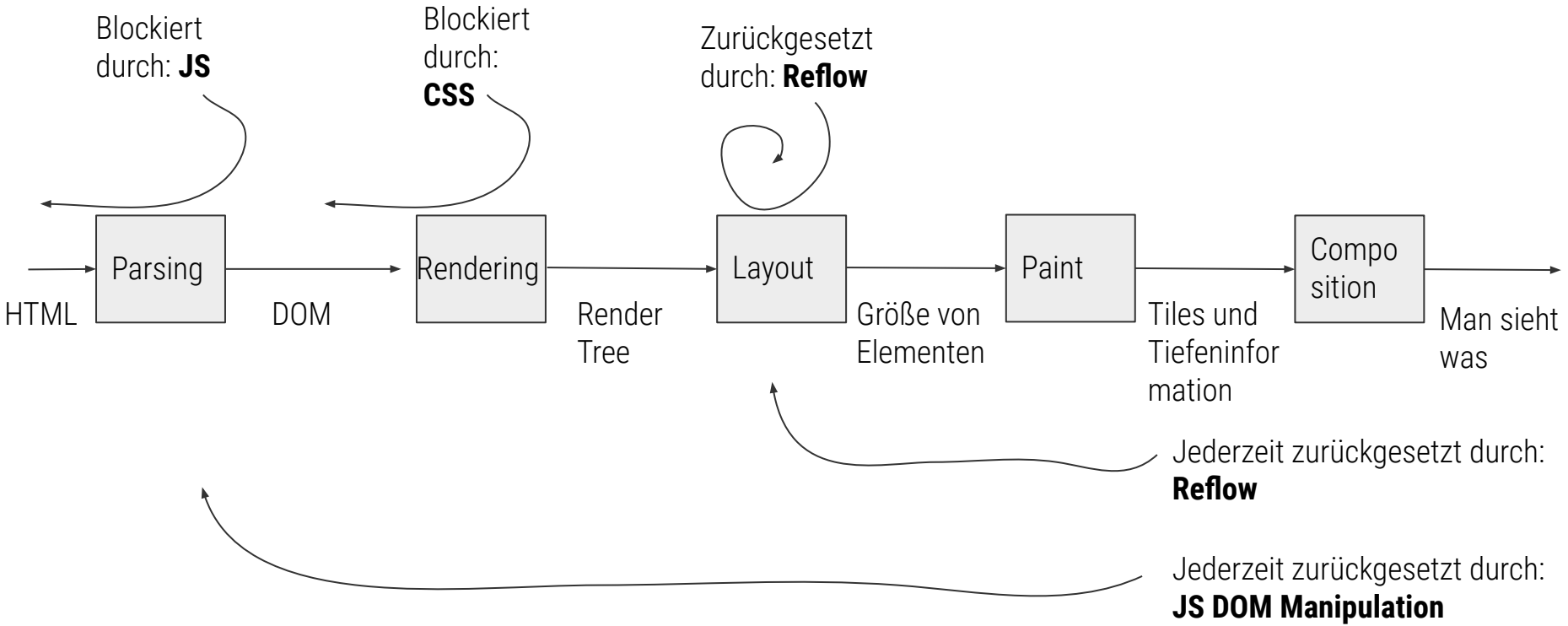
- CSSOM bestimmt Anzeigeoptionen von DOM Knoten
  - **C**SS **O**bject **M**odel
- Baum Repräsentation von HTML mit CSS Eigenschaften
- Nicht direkt manipulierbar
- Quellen sind jede Form von CSS
  - Sowohl extern als auch inline
- Wichtig für sog. Render Tree



Render Prozess - Render Tree

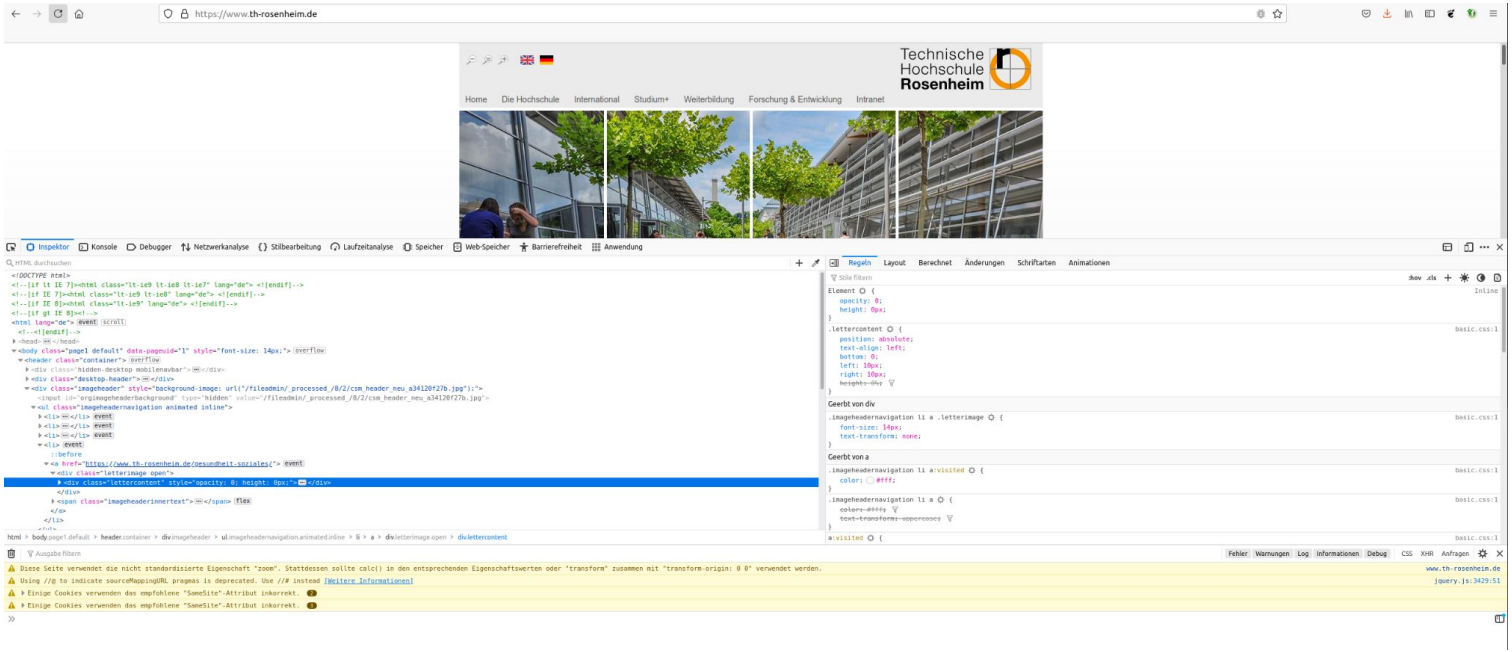
- 

Render Prozess



Browser Tools

- Browser bieten umfangreiche (Entwickler)-tools
  - Debugging, Profiling, Prototyping, ...



Browser Tools

- Inspektor
- Netzwerkanalyse
- Laufzeitanalyse
- Local Storage >> https://www.th-rosenheim.de/
- Barrierefreiheit >> https://www.dbsv.org/
- Konsole
  - >> https://www.bild.de/
  - >> https://www.spiegel.de/

Browser Extensions

- Erweitern Browser um zusätzliche Funktionalität
- Können Browser APIs nutzen
  - Haben oft vollen Zugriff auf Aktivitäten
    - Entfernen/Hinzufügen von Headern
    - Auslesen des lokalen Speichern
    - Mitlesen der Browser History
    - …
- Sehr wertvolle Werkzeuge
- Aber mit Vorsicht zu genießen

>> https://madweb.work/preprints/madweb21-paper16-pre_print_version.pdf

CSS Allgemein

- **C**ascading **S**tyle **S**heets (1995)
- Erlaubt separate Formatierung von HTML und XML
  - Trennung von Inhalt und Formatierung möglich
  - Unterschiedliche Formatierung abhängig von Anforderung
    - Über sog. Media Queries
- Referenziert Elemente (auch) über DOM
  - Sehr ausgefeilte Selektoren möglich
- Anweisungen legen Eigenschaften von Elementen fest
  - Werden anhand von Position im DOM überschrieben oder geerbt
    - Der mehr spezifische Selektor gewinnt

CSS - Syntax

- Selektor (-Liste)
- Anweisung
  - Beinhalten Eigenschaften

```
Selektor1 [, Selektor2 [, …] ] {
    Eigenschaft-1: Wert-1;
    …
    Eigenschaft-n: Wert-n[;]
}
/* Kommentar */
/* In eckigen Klammern stehen optionale Angaben */
```

CSS - Selektoren

- Erlauben Definition von Wirkungsbereich angegebener Eigenschaften
- Können:

```css
/* <body> */
body {
    color: coral;
    font-size: 0.9em;
}

/* <a href=""> */
a {
    color: #2fcfaf;
    text-underline: none;
}
```

Elemente treffen

```css
/* <div id="navigation"> */
#navigation {
    font-size: larger;
}

/* <div class="card"> */
.card {
    color: azure;
    opacity: 0.9;
}
```

Attribute treffen

```css
/* <div class="card"> */
div.card {
    color: azure;
    opacity: 0.9;
}

/* <input type="text"> */
input[type=text] {
    font-size: larger;
}
```

Kombination aus beidem

CSS - Selektoren

- Können auch:

```
p:nth-child(odd) {
    background: red;
}

p:nth-child(even) {
    background: blue;
}

div:hover {
    opacity: 0.5;
}
```

```
div.menu, div.footer, .red{
    background-color: red;
}

/* <div><a/></div>*/
div span a {
    background-color: red;
}
```

DOM Bedingungen nutzen

Spezifisch und mehrdeutig sein

CSS - Arten von Eigenschaften

- CSS kann viele Dinge beeinflussen
  - Formatierung
  - Positionierung/Layouting
  - Verhalten
    - Animation
    - Reaktion auf Nutzerverhalten
- Klarer Fokus auf
  - Aussehen
  - Interaktion

>> https://www.w3schools.com/css/css_intro.asp

CSS - Integration

- CSS kann extern als auch inline integriert werden
  - Ähnlich JS
  - Ist Render-Blocking!
- Inline als Element oder Elementattribut

```
<head>
    <meta charset="UTF-8">
    <title>Formidable</title>
    <style type="text/css">
        label { display: inline-block; width: 210px; text-align: right; }
    </style>
    <link rel="stylesheet" href="/styles/base.css">
</head>
<body>
<form action="form.php" method="post" style="text-decoration: none">
    <label for="form_name">Name</label>
    <input type="text" id="form_name" name="name" required>
```

CSS - Priorisierung

- Cascading bedeutet Umgang mit Mehrfachselektion
- Priorisierung gibt an welche Eigenschaften angewandt werden

CSS priority scheme (highest to lowest)

Priority	CSS source type	Description
1	Importance	The "`!important`" annotation overwrites the previous priority types
2	Inline	A style applied to an HTML element via HTML "style" attribute
3	Media Type	A property definition applies to all media types, unless a media specific CSS is defined
4	User defined	Most browsers have the accessibility feature: a user defined CSS
5	Selector specificity	A specific contextual selector (`#heading p`) overwrites generic definition
6	Rule order	Last rule declaration has a higher priority
7	Parent inheritance	If a property is not specified, it is inherited from a parent element
8	CSS property definition in HTML document	CSS rule or CSS inline style overwrites a default browser value
9	Browser default	The lowest priority: browser default value is determined by W3C initial value specifications

CSS - Spezifizität

- Abhängig vom gewählten Selektor in Relation zum DOM
- Mehrstelliges System
  - Inline-Element . ID . Klasse . Elemententyp
- Je höher und je weiter link die Zahl, desto spezifischer

Selectors	Specificity
`h1 {color: white;}`	0, 0, 0, 1
`p em {color: green;}`	0, 0, 0, 2
`.grape {color: red;}`	0, 0, 1, 0
`p.bright {color: blue;}`	0, 0, 1, 1
`p.bright em.dark {color: yellow;}`	0, 0, 2, 2
`#id218 {color: brown;}`	0, 1, 0, 0
`style=" "`	1, 0, 0, 0

Quellen:

- https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Client-side_web_APIs/Introduction
- https://www.w3.org/TR/CSS1/#inheritance
- https://medium.com/jspoint/how-the-browser-renders-a-web-page-dom-cssom-and-rendering-df10531c9969
- https://developers.google.com/web/updates/2019/09/inside-browser-part3
- https://blog.logrocket.com/how-browser-rendering-works-behind-the-scenes-6782b0e9fb10/
- https://en.wikipedia.org/wiki/CSS
- https://www.w3schools.com/css/css_intro.asp
- https://en.wikipedia.org/wiki/CSS#Selector