HTML - Hypertext Markup Language	CSS - Cascading Style Sheets	JavaScript(ProgrammierSpr) (ECMAScript Name der Sprachdefinition)	$Loopen \ \ddot{u}ber \ for (vari=0; i < items.length; i++), for (constitem \ of \ Array.from (items)),$
Struktur	Vorteile: Trennung Struktur/Layout, Fluides Design, unterschiedliche Ausgabemedien	Include: <script>alert("test")</script> →head, <script src="foo.js"></script> → body	for(const item of items)
html <html lang="en"><head><meta charset="utf-8"/><title>Title>/title></</td><td>unterstützen, weniger copy&paste - wiederverwenden, Styles auslagern in Css-Dateien</td><td>

</td><td>querySelector() (erster Match), querySelectorAll() (gibt alle/nodelist zurück) → const
nav = document.getSelector('.class') oder const container = document.getSelector('#id');</td></tr><tr><td>head><body>Hello World</body></html> - Charset erste 1024Byte im File-Chunk →</td><td>Inline: Style: Im <head> Besser
Performance da 1 HTTP Anfrage weniger <head><style> span{ background: red;}</td><td>Primitives: String, number, boolean, null, undefined, symbol → immutable let foo=5; Objects: Plain Objects, Arrays, Regex, Functions → mutable typeof([]) → object</td><td><pre><button id="addButton">add item</button></pre></td></tr><tr><td>(überprüfen im browser network tab → response header), iFrame, UTF-8 variable length</td><td><pre></pre> <pre></pre> <pre><</td><td>Var/let: var mit var redeklaieren ist ok, let redeklarieren in gleichem Scope/Block nicht</td><td><ul id="list-cont"></td></tr><tr><td><meta name="" content=""/> - name= description, author, publisher, keywords HTML5 Elements</td><td>Selektoren</td><td>Rechenregeln: Punkt vor Strich & von links nach rechts. Str+Val = Str, Val+Str=String</td><td><pre>const listCont = document.getElementById('list-cont'); const liItems = document.getElementsByTagName('li');</pre></td></tr><tr><td><div><nav><main><section><aside><article><header><footer><figure><figcaption></td><td>Universal * { } Typ a { } Attribut input[type=submit] { }</td><td>"4" + "2" = "42"</td><td><pre>document.getElementById('addButton').addEventListener('click', () => {</pre></td></tr><tr><td>Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein End-Tag - optional slash (
 Void Elemente - kein Elemente - optional slash (
 Void Elemente - optional slash (
 Void Elemente - optional slash (
 Void Ele</td><td>ID (performant) #my-id { } (eindeutig) Klassen .my-class { } (mehrfach)</td><td>"4" + 2 = "42" 4- "2" = 2 "4" - 2 = 2 5 + +"2" = 7 <math>\rightarrow</math> + "2" = 2 5 -, *, / "a" = NaN Vergleichen: immer mit === (verhindert Typumwandlung von primitiven DT)</td><td><pre>const li = document.createElement('li'); li.appendChild(document.createTextNode(liItems.length + 1)); listCont.appendChild(li);</pre></td></tr><tr><td>area, base, br, col, embed, hr, img, input, keygen, link, meta, param, source, track, wbr</td><td>Nachfahren p a (irgendwann) Kind p > a (p ist direkter Parent)</td><td>"123" == 123 <math>\rightarrow</math> t "123" === 123 <math>\rightarrow</math> f [] == false <math>\rightarrow</math> t null == false <math>\rightarrow</math> f [1,2]=="1,2"<math>\rightarrow</math>t</td><td>})</td></tr><tr><td>Nicht verwendbare Zeichen</td><td> Geschwister p ~ a (irgendwann auf Ebene) Nachbarn p + a (a direkt nach p) div.box → alle div's mit class "box", .box.box1 → Elemente welche beide Klassen haben</td><td>"aa" < "ab" <math>\rightarrow</math> t [] == "" <math>\rightarrow</math> true {} == true <math>\rightarrow</math> f {} == "" <math>\rightarrow</math> false undef===undef</td><td>DOM Manipulation</td></tr><tr><td>Zeichen müssen ersetzt werden & <math>\rightarrow \&</math>amp; <math>\langle \rightarrow \&</math>lt; <math>\rangle \rightarrow \&</math>gt; "<math>\rightarrow \&</math>quot;</td><td>Element-Selektor → auf ein Element (z.b. div)</td><td>Boolean: false = (0, "", null, undefined, NaN), true = (Alles andere, "0", "false", [],)</td><td>EventTarget (addEventListener(), removeEventListener()), Node: prop: childNodes,</td></tr><tr><td>Includes</td><td>Attributselektoren <math>a[href]\{\}, [title = Berlin]\{\}, [href = "http"] \rightarrow Case Sensitive</math></td><td><math>!(\text{null}) \rightarrow \text{true}; "a" < "b" \rightarrow \text{true} (ASCII val); [] = "" \rightarrow \text{true}; boolean(0) \rightarrow \text{false}</math></td><td>nodeValue, firstChild, nextSibling. Fkt: appendChild(), removeChild(), Element: prop: id, className/classList, innerHTML, attributes. Fkt: get-/set-/toggleAttribute(), closest().</td></tr><tr><td><pre></pre> </pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><pre></pre><p</td><td> </td><td>Number: alles Floats, <math>0.1+0.2=0.3 \rightarrow false(Check Number.isSafeInteger()) + Wert \rightarrow num parseInt("1.7xy") parseFloat("1.2ab") "1ab" = <math>+(true) = 1, +(x13*) = 13, +(x13*) = 13</math></math></td><td>querySelector(), querySelectorAll(). Parent: children, first/lastElementChild, append(),</td></tr><tr><td><pre></pre> <pre></td><td>Startet mit Wort [a =v] (v alleinstehend oder -) Startet mit [a^=v]</td><td>= 1 = 1.2 NaN NaN</td><td>remove(). HTMLElement: dataset, style, hidden</td></tr><tr><td><pre><script src="main.js" async></script> (immer schliessen! Sonst alles als Script interpretiert. Im Footer einbinden)</pre></td><td> Beinhaltet Wort v [a~=v] (v alleinstehend, -) Beinhaltet Wort v [a*=v]</td><td><math>NaN = parseInt("abc"), typeof(NaN) \rightarrow 'number' NaN ==(=) NaN \rightarrow fals \rightarrow isNaN(x)</math></td><td><pre>const newEl = doc.createElement('div'); newEl.appendChild(document.createTextN- newEl2.innerText = 'World'; docu-</pre></td></tr><tr><td>Content Model</td><td>Pseudoelemente für Inline-Block, Listenpunkte, Tabellenüberschriften</td><td> 0/0, Error Wert typeof(null) → 'object' o=/hsr/i; typeof(o) → object Infinity: 3/0, Math.pow(2,10000) Negativ Infinity: -Math.pow(2,10000)</td><td><pre>ode ('Hello')); document.getElement- ById("container").appendChild(newEl); pendChild(newEl2)</pre></td></tr><tr><td>caption, col, colgroup, dd, dt, head, html, legend, li, optgroup, option
source, tbody, th, thead, tr</td><td><pre>::before, ::after, ::first-line, ::first-letter, ::selection .question::before{ content: '?', } Pseudoklassen</pre></td><td>String: length, slice(), trim(), indexOf(), includes(), `Mein Name ist \${name}` \rightarrow Inhalt</td><td>innerHTML: setzt HTML Content innerText: sichtbarer Text, textContent: komplet Text El.</td></tr><tr><td></td><td>Strukturell: :first-child, :nth-child(2), :nth-child(2n) (jedes 2te) :nth-child(odd) (alle</td><td>als JS interpr. Immutable <math>\rightarrow</math> s="hsr"; s[2]="x" <math>\rightarrow</math> "hsr" cannot be changed</td><td><div id="el" class="box alert important"></div></td></tr><tr><td>Section Root Flow body, td div. dl. footer, form, header, main ol. p.</td><td>geraden), nth-of-type(), p:first-of-type, p:last-of-type(), p:only-of-type(), div:empty</td><td>Undefined: Var. nicht definiert oder nicht initialisiert. typeof(myVar) != 'undefined' &&</td><td><pre><script>const el = document.getElementByld("el"); console.log(el.className); // box alert important console.log(el.classList); // DOMTokenList(3) ["box", "alert", "important"] </script></pre></td></tr><tr><td>fieldset div, dl, footer, form, header, main ol, p, table, ul</td><td>Dynamisch: :hover, :active, :focus, :visited</td><td>myVar.a == undefined; typeof(undefined) <math>\rightarrow</math> undefined
Null: Wert von Var, null == undefined <math>\rightarrow</math> true, null === undefined <math>\rightarrow</math> false, null == false/</td><td>DomTokenList hat Funktionen: add / remove / toggle / contains / Replace</td></tr><tr><td>Phrasing</td><td>Diverse: p:lang(en), :not(p), :is(header, main, footer) → header, main oder footer</td><td>true <math>\rightarrow</math> false, null == false <math>\rightarrow</math> false, [] == ![] <math>\rightarrow</math> true, typeof(null) <math>\rightarrow</math> object</td><td>DOM Events auf Ereignisse reagieren, Caputre-, Target, Bubble- Phase</td></tr><tr><td>Sectioning area, br. em. input (hidden,</td><td>Performance A > B wenn möglich A B vorziehen. A > #ID ist unnötig da #ID sowieso eindeutig ist. Auf</td><td>Array: keine fixe Länge, start 0. arr[0] = 'test' / arr.push('test') / arr.lenght / arr[key] /</td><td>Events: function myFunc(event){}; document.getElementByld('id').onclick = myFunc / button.addEventListener("click", myFunc); / button.removeEventListener('click', myFunc);</td></tr><tr><td>arcicle side map, span, strong Interactive a, input, label, select,</td><td>Vererbung verlassen .navigation li a ist schlecht, besser nur .navigation Universelle</td><td>arr.forEach(function(element, index){}); / arr.map(function(element){}); /</td><td>Registrieren: Listener, onclick, inline – Prüfung inline verwenden < button</td></tr><tr><td>textarea</td><td>Regeln wie [href]{} vermeiden</td><td>arr.filter(function(element){return element > 5;}}; / arr.every(function(element){return element > 5;}}; /] instanceof Array → true</td><td>onclick="alert('3')">3</button> oder <input onkeydown=«console.log(event.key)»> oder</td></tr><tr><td>Heading Embedded audio (controls), embed,</td><td>Box Model margin (ausseabstand), padding (innenabstand)</td><td>terieren: default = Zugriff via index, in = iteriert über Index, of = über Werte</td><td><input onchange=«console.log(event.target.value)»></td></tr><tr><td>hgroup,
h1-h6 audio, img. iframe, img (usemap),
video (controls)</td><td>Border-box: definiert fixe grösse, padding & border werden davon abgezogen</td><td>for(var i=0;i<arr.length;i++){ for(var p in arr){ for(var v of arr) {</td><td>click, dblclick, mousedown, mouseup, mouseover, mouseenter, mouseleave,
mousemove, keyup, keydown, keypress, copy, cut, paste, focus, select, load, resize, scroll</td></tr><tr><td></td><td>Outline: gleich wie border, vergrössert das Element aber nicht Global-reset: "{ padding: 0; margin: 0}</td><td>console.log(arr[i]);</td><td>target (zeigt auf das effektive Element, für welches der Event gilt)</td></tr><tr><td>Metadata</td><td>Short writing: margin: 1px 2px 3px 4px (top right bottom left)</td><td>Object: Sammlung von properties. var person = {name: 'Tim', hallo: function(){}}; var name = person("name"); person.alter = 10;</td><td><u>currentTarget</u> (zeigt auf das aktuelle Element, wo sich der Event registriert hat)</td></tr><tr><td>link, meta, style, title</td><td>Display</td><td>function showData(pers)[for(const x in pers) [console.log(x + ' = ' + pers[x]); }} // K+V</td><td>DOM Lifecycle - 3 Zustände, HTML oben nach unten Laden, kennt Elemente noch nicht</td></tr><tr><td></td><td>Block (füllt ganze zeile), inline (erlaubt margin/padding), inline-block (erlaubt</td><td>Functions: function calc(fn, a, b) { fn(a, b); } const add2 = function add(a) { return</td><td>loading: Dokument wird geladen. Interactive: Dokument geladen aber gewisse ressourcen noch nicht → durch Event "DOMContentLoaded" repräsentiert. complete:</td></tr><tr><td>Text Flow: <h1>-<h6> <small> <abbr> <blockquote>
 (Wichtig)</td><td>margin/padding/width/height), none (entfernt), visibility:hidden (space bleibt)</td><td>function(b) { return a+ b; }} typeof(function() { return "a"; } <math>\rightarrow</math> function</td><td>Event "load"</td></tr><tr><td><address></address> <time datetime="2016-02-11 12:45"> </time></td><td></td><td>Offene ParamListe: arguments beinhaltet Params, arr.from(arguments).join("\n"); Rest Param: function foo(name,params) {con.log(2, params.join(";")); → "foo;bar"</td><td>"DOMContentLoaded": für Libraries/Skripts die nicht wissen wie sie genutzt werden.</td></tr><tr><td>Phrasing: <cite> (title of work) <code> (Betonung) <mark> <i><hr/></td><td>reserviert), static (standard-positionierung im Fluss)</td><td>.name - name der funclength - anzahl params. No function Overloading → wird</td><td>"Load": nötig falls aufgrund von Elementgrössen Aktionen ausgelöst werden müssen</td></tr><tr><td>Hyperlinks</td><td>Float Text um Bild fliessen lassen – float: left/right</td><td>überschrieben. Defualt Werte function x(name = "foobar")</td><td><u>Defer/Async:</u> <u>defer</u> (Skript files nach parse von Dokument ausführen, aber vor
"DOMContentLoaded"-Event, Reihenfolge gewährleistet), <u>async</u> (files asynchron laden,</td></tr><tr><td>Report</td><td>Cascading - Kaskade</td><td>Lamda: arr.filter(elem => elem > 5); / arr.map(x => x*x); Context always on same Obj.</td><td>aber vor "load"-Event, Reihenfolge nicht gewährleistet)</td></tr><tr><td>Lists</td><td>Spezifität: Für jeden ID-Selektor: A++, für jeden Attribut-/Klassenselektor oder</td><td>Scope: Sichtbarkeit von Variablen & Funktionen, jede Func. & Obj. generiert neuen Sc. In Scope Zugriff auf dessen Vars, globale Vars & parent Scope vars, (closure). Variablen</td><td><pre>doc.addEventListener("DOMContentLoaded", () => console.log(document.readyState));</pre></td></tr><tr><td> <ol start="2" reversed> <dl><dt>term</dt> <dd>details</dd>
Tables</td><td>Pseudoklasse: B++, für jeden Typselektor & Pseudoelement C++. Universalselektor * wird ignoriert und :not() wird nicht mitgezählt, der Inhalt der Klammer aber schon. Die</td><td>ohne "var", "let", "const" →global scope angelegt, sonst aktueller scope. globalObject.b</td><td>Attribute werden auf Properties im JS gemapped</td></tr><tr><td><caption>Rang</caption><thead>NameZeit</td><td>Endwerte bestimmen die Reihenfolge → je mehr Punkte desto höher gewertet.</td><td>Block scope: Var mit "var" haben keinen Block scope → ausserhalb des Blocks { }</td><td><pre>class → «class name» bzw. «class list» for html. <input type="button" value="value-content" class="info-class" required/></pre></td></tr><tr><td>>>>>></td</td><td>* /* A=0, B=0, C=0, Spezifität 0 0 0 */ h1 /* A=0, B=0, C=1, Spezifität 0 0 1 */</td><td>verwendbar <math>\rightarrow</math> { var x = 2 } //x CAN be used here. Vars. mit "let" haben Block Scope <math>\rightarrow</math> { let x=2 } //x Can't be used here. Redeklaration einer "var" Var. Innerhalb eines Blocks</td><td><pre>const input = document.querySelector("input"); console.log(input.type, input.value, input.className); input.value = "123"</pre></td></tr><tr><td><tfoot>Total</tfoot> - 2 Kolonen zusammennehmen</td><td>ul li /* A=0, B=0, C=2, Spezifität 0 0 2 */
a::after /* A=0, B=0, C=2, Spezifität 0 0 2 */</td><td>redeklariert auch die Var. ausserhalb des Blocks. Redeklaration einer "let" Var. Nicht.</td><td><pre>input.setAttribute("myProp", "myVal"); input.removeAttribute("checked") eigene Attribute & Boolean Attribute mit set-/removeAttribute(); anpassen;</pre></td></tr><tr><td>Images</td><td>p:first-child /* A=0, B=1, C=1, Spezifität 0 1 1 */ [a:not([href]) /* A=0, B=1, C=1, Spezifität 0 1 1 */</td><td>Browser <math>\rightarrow</math> globales Objekt ist window. Kein File-Scope, alles im globalen Scope</td><td>Data-Attribute: Datenaustausch HTML ↔ JS. Name in HTML kebab-case & im Dictonary</td></tr><tr><td><pre></td><td>ul.nav [href] /* A=0. B=2. C=1. Spezifität 0 2 1 */</td><td>This Context: Als methode von Obj. object.foo(); <math>\rightarrow</math> this = object. Funktion mit new new foo(); <math>\rightarrow</math> this = undefined, unbound Funktion foo(); <math>\rightarrow</math> this = globales Objekt. Jeder</td><td>«dataSet» camelCase z.b. data-factory-id → dataset.factoryId. Für metainfo & wichtig für</td></tr><tr><td>Audio/Video/Object/SVG</td><td>#author /* A=1, B=0, C=0, Spezifität 1 0 0 */
#editor p /* A=1, B=0, C=1, Spezifität 1 0 1 */</td><td>Funkt. kann mit apply() oder bind() Context gesetzt werden – andere Regeln ignoriert.</td><td>Event-Bubbling. Bsp JS: left:delectriccars" data-columns="3" data-index-number="12314" data-par-"</td></tr><tr><td><audio viueo/object/3v3</p> <audio controls autoplay loop muted><source src="" type="audio/ogg"/></audio></td><td></td><td>foo.call({counter:123}); <math>\rightarrow</math> this = 123. Mit bind() kann Context vorgegeben werden</td><td>ent="cars"></article>; <script>console.log(document.getElementById("electric-</td></tr><tr><td><video poster="y.jpg"><source type="video/mp4" src=""/>Fallback link</td><td>Dekl. Im Browser (kleinste Prio), 2. Dekl. Des Benutzers, 3. Dekl. Des Autors, 4. Wichtige Dekl. Des Autors (!important), 5. Wichtige Dekl. Des Benutzers (grösste Prio)</td><td>(andere Regeln ignoriert ausser bei new). Bei Lambda ist Context ist immer auf dasselbe Obj gebunden – (lexical) parent context. const fn = fn2.bind{{name:"foo"}};</td><td>cars").indexNumber);</script> //12314 CSS: attr() liest Inhalt aus, Attribut-Selektoren können die Werte selektieren.</td></tr><tr><td><video id="tiNspire" poster="y.png" controls> <source src="x.mp4" type="video/mp4"></td><td>→ innerhalb von jedem Punkt wird nach Spezifität sortiert. Gleich → Reihenfolge wins</td><td>casseibe Obj gebunden – (lexical) parent context. const in = in2.bind({name: roo }); fn3(); <math>\rightarrow</math> this = obj mit foo;</td><td>article::before { content: attr(data-parent); } article[data-columns='3'] { width: 400 px; }</td></tr><tr><td><track src=".x.en.vtt" kind="subtitles" srclang="en" default></video></td><td>Vendor prefixes</td><td>Use strict: eliminiert "fails silently". Eliminiert probleme die code optimierung</td><td>Validation - ohne JS kann die Fehlermeldung & Aussehen nicht angepasst werden</td></tr><tr><td><object type="image/svg+xml" data="xy.svg"></obj> Frames - inhalt aus Web einbinden</td><td>-webkit- (chrome) -moz-(FF) -ms-(IE) -o-(old opera)</td><td>verhindern, Security wird leicht verbessert. Anfang vom File oder Funktion (vererbt)</td><td>& Abhängigkeiten nicht geprüft werden. Falls CSS anpassen: novalidate Attr. & error</td></tr><tr><td><pre><figure></figure></figure></figure></pre></td><td>Standard immer als letztes angeben, da sonst ignoriert!</td><td>Hoisting: JS verschiebt alle Deklarationen (let a) von Methoden / Variablen & Funktions Definitionen, an den Anfang des aktuellen Scopes - die Initialisierung (a=5) nicht.</td><td><pre>{color: «» } valueElement.addEventListener('input', function (event) {</pre></td></tr><tr><td>Forms - "for" von label für "id" von input - input sendet value="" mit button nicht!</td><td>Einheiten/Grössen px: fix, em: relativ zu Parent, rem: relativ zu Root, vm/vh/vmin/vmax: % von Viewport</td><td>«const» & «let» Variablen werden ge-«hoisted» aber vor der Deklaration der Variable</td><td>if (valueElement.validity.tooShort) {</td></tr><tr><td><pre><form><fieldset><legend>Autor</>Felder</fieldset><fieldset><legend>Rechtliches</pre></td><td>Counter body { counter-reset: my-counter;}</td><td>(ursprünglicher Ort) darf diese nicht zugegriffen werden (tem-poral dead zone). Bei</td><td><pre>valueElement.setCustomValidity('Viel zu kurz!'); } else {</pre></td></tr><tr><td>legend></fieldset></form></td><td>h3:before {counter-increment: my-counter; content: counter(my-counter);}</td><td>einer «Function-Expressions» wird die Zuweisung nicht verschoben! Hoisting Scope Creates global properties</td><td><pre>valueElement.setCustomValidity("); } });</pre></td></tr><tr><td><label>Name <input type="hidden text search password url tel email checkbox radio date color range" name="name" placeholder="stadt" required autofocus /></label></td><td>Font & Color</td><td>var Declaration Function Yes</td><td>Node.js - Javascript auf Server</td></tr><tr><td><pre><label>Text<textarea name="profiltext" rows="10" cols="100"></textarea></label></pre></td><td>font-family: erste Schrift wird gewählt, font-style: italic, oblique, font-variant: small-</td><td>1et Temporal dead zone Block No
const Temporal dead zone Block No</td><td>Request: method (get, put, post), url (angefragte url)</td></tr><tr><td><pre><label>Nationalität <select name="nationality"><option value="chf" selected>Schweiz</pre></td><td>caps, font-size: Grösse, font-weight: bold, lighter, line-height: Zeilenhöhe, text-</td><td>function Complete Block Yes</td><td>Response: writeHead, setHeader, statusCode, statusMessage, write, end</td></tr><tr><td>option></select></label> <label for="imgs">bilder</label><input type="file" id="imgs" name="images" multiple/></td><td>decoration: underline, overline, line-through, color: red, #ff0000, rgb(255, 0, 0), text-
align: center</td><td>class No Block No</td><td>JSON - String, Number, Boolean, Array, Object & null. NaN & Infinity werden zu null</td></tr><tr><td><pre><label>Herr <input type="radio" name="gender" value="male" checked></label></pre></td><td>List</td><td>import Complete Module-global No</td><td>Helpers: JSON.parse & JSON.stringify - {"alter":"18", "name":"Regula"} - double quotes! Array: nicht verwenden, da JSON Hijacking und keine Erweiterung möglich</td></tr><tr><td><label>Frau <input type="radio" name="gender" value="female" ></label></td><td>list-style-type: circle/disc/square/'-'/none list-style-position: inside / outside</td><td>Regex: /muster/flags Flags: g(lobal),i(gnore case), m(ultiline), u(nicode)</td><td>HTTP (Hyptertext transfer protocol) - immer höchste http Ver. Wird verwendet</td></tr><tr><td><input type="<mark>search</mark>" name="search-modell" list="<mark>list</mark>" id="search-modell"/> <<mark>datalist</mark> id="<mark>list</mark>"> <option value="Option1">Text</option></datalist></td><td>list-style-image: url('xy.gif'), list-style: circle inside url("/images/arrow.svg");</td><td>'John Smith'.replace(/(\w+)\s(\w+)/, '\$2, \$1'); // Smith, John</td><td>URI Schema: 5 Teile - <schema-name>:<hierarchical part>[?<query>][#fragment]</td></tr><tr><td><pre>cdatalist id= list > <option value= Option1 >Text</option></pre>/datalist> <button type="submit reset">Search</button></td><td>Table</td><td>str.split(regx); // array of strings / str.match(regx); // null/string / regx.test(str); // bool Bubbling: (event.target = effektives Event Objekt) Events werden im Tree nach oben</td><td>Bsp: foo://example.com:8042/over/there?name=ferret#nose</td></tr><tr><td>disabled - Wert wird nicht an Server übermittelt, readonly - Wert wird an Server</td><td>border: [width] [style] [color] border-collapse: collapse/separate border-spacing: [horizontal] [vertical] caption-side: top/bottom/ vertical-align: basline/top/middle/bot-</td><td>propagiert (disable → event.stopPropagation())</td><td>URI: besteht aus URN & URL URN: ist eine URI. Resource dauerhaft bezeichnen.</td></tr><tr><td>geschickt. Name & value Attribut wird an Server geschickt → [name]=[value]& <pre>cprogress>, <meter>, <output></pre></td><td>tom/ table tr:nth-child(even) <math>\{\} \rightarrow</math> jede gerade Zeile stylen</td><td>DOM (HTML → Parse HTML → DOM Tree → Render Tree)</td><td>Beschreibt was etwas ist. (uri: tel:+1-819-555-1212, urn: +1-819-555-1212) URL: ist eine URI. Beschreibt wo etwas ist und wie man dahin kommt. (https://t3n.de – http zeigt wo</td></tr><tr><td><pre><pre><pre><pre><select name="vehicle" multiple></pre></td><td>th[scope=row] {} → Header horizontal Stylen</td><td>Objekte: window (glob vars, console, history, etc.), document (Einstiegspunkt DOM,</td><td>die Ressource ist) Encoding: <math>// \rightarrow \%2F</math>, ' (space) <math>\rightarrow \%20</math>, '&' <math>\rightarrow \%26</math>, '#' <math>\rightarrow \%23</math>, '?' <math>\rightarrow \%3F</math></td></tr><tr><td><pre><optgroup label="car"><option value="bmw">bmw</option></optgroup><optgroup</pre></td><td>Form</td><td>.body, .height, .title), NewLine (im DOM Tree leerer Node, \n etfernen, als Kommentar) Suche: getElementById(), getElementsByClassName(), getElementsByTagName(),</td><td>Request - Method, Address (URL), Prot. Vers., Request Headers, Header/Body Separator</td></tr><tr><td></td><td></td><td>AUTHOR VELCTEMENT RVING L. VETT LEMENTS RVI (2000 2000 2000 2000 2000 2000 2000 20</td><td></td></tr><tr><td> label="moto"><option></option></option></option></option></option></option></option></option></option></option></option></option></option></option></option></option></option></option></option></option></option></option></option></option></option></td><td>Pseudoklasse: :invalid, :valid, :focus, :default, :in-range, :disabled, :enabled Pseudoelement: ::placeholder</td><td>getElementByName geben Collection zurück, kein Array! & erzeugen LiveQuery (passt</td><td>GET (data in url /search?query=abc), HEAD (transfer header and status line only), POST (data in request body), PUT (replaces all current representations), DELETE (delete all</td></tr></tbody></table></title></head></html>			

current representations), CONNECT (tunnel to server), OPTION (communication options), Private / Public Members Inheritance dateSonos() { JavaScript Examples document.getElementById("songs").innerHTML = this.renderSongs(); class Bar { TRACE (message loop-back along the path to the target) class Bar { #value = 0; // #private field constructor(value) JS Scoping HTTP Response: Besteht aus Protocol Version, Status Code, Response Header, valuePublic = 0: //public field this.value = value:} const globalObj = typeof(global) != 'undefined' ? global : window; //File Scop istenOnListButtonClick() { Header/Body separator (2x crlf), Response Body (returning HTML code) constructor(value) { this.#value = value document.getElementBvId("songs").addEventListener("click", (event) => { lass Eon extends Bar var varA = 'varA': //File Scope Status Codes: 1xx (Informational), 2xx (Successful, 200 OK), 3xx (Redirection, 301 moved lobaler Sco if (event.target.tagName.toLowerCase() !== "button") { get prop() { return this.#value; } constructor(value function scope() { return: permanently), 4xx (Client error, 400 Bad request, 401 unauthorized, 403 forbidden, 404 set #prop(value) { this. #value = value} super(value): let b = 0; //Funktion Scope c = "ABC"; //globaler Scope varA = 'varAA'; a++; b++; not found). 5xx (Server error, 500 int. Server error) 9xx (non-standard codes) se super class methods/propertie: const songId = Shop.getSongIdOnParentOf(event.target); let getterValue = fooBar prop Host Header: Use same ip for multiple virtual hosts on a server. Host: www.host1.com hy using the super keyword console.log(g, varA, b, c); //1 'varAA' 0 ABC console.log(globalObj.a, globalObj.varA, globalObj.b, globalObj.c); //undefined undefined 0 'ABC' if(event.target.classList.contains("js-rating")) { Content-type entity header: deklariert parsing format für client/server in MIME-Type MVC songs.rateSongById(songId, event.target.dataset.delta);
this.updateSongs(); format (specifies media type (text, audio, application) and a subtype html → text/html) Benefits: löst View vom Model (logic), abstrahiert wie objekte agieren (Controller) } else if (event.target.classList.contains("add-to-basket")) { Model: Data & Business Ingic Hoisting Cookie wird auf Client gespeichert und bei iedem Request dem Server übergeben. Als View: Display data/state (html) }); var x = 1; var y = 2
function print(x) { HTTPOnly deklarieren → cookie wird nur für http/HTTPS requests verwendet. Als Secure Controller: Take user input, interaction function print(x) { console log(x); console log(y); deklariert → Cookie nur für HTTPS requests verwenden. between ui and bl (services), has routing console.log(x); console.log(v): Cookie: document.cookie = "username=Foo: expires=Thu, 18 Dec 2013 12": features static getSongIdOnParentOf(element) { var x = 3; var y = 4; let x = 3; let y = 4; Structure: bl/(animal.js, animal-service.js), while(!element.dataset.songId) {
 element = element.parentElement; AJAX - send/receive data asynchronously (in the background) print(); print(); common/(utils.js), dl/(food-strorage.js), ui/ x & v sind "undefined" Error \rightarrow x ist bereits im block scope print(x) deklariert. Communications JS ↔ Server → no page reloads needed (effektiv), mainly used in SPAs bootstrapper.js, ui/controllers/zoo-controller.js (eventHandlers) let nicht redeklarierbar. Ohne "let x=3" → x ist if(element === null) { XmlHttpRequest (XHR) provides JavaScript interface for asynchronous calls - ajax api throw new *Error*("Songlist has no song-id in a parent."); undefined, v is not defined → temporal dead zone Asynchronous modules Promises represents eventual completion (or failure) of async operation, handle callback const promise = fs.readFile('xy'); promise.then((content) => { //succeed }, (err)
=> { //failure }).catch(err => { //suc./failure generated err }); export expose variables/functions/classes to other modules, export class Food {} let Selektoren - select foo1 return element.dataset.sonoId: no: exmount (fook: import to import them, import (Food) from "/food is": successCallback = SC failureCallback = FC sscript type="module">import {bootstra as BS} from './bootstr.js'; BS.start();<script> <div id="foo2" class="foo"></div> doc.querySelectorAll('.frame > #foo1'); Handlebars templating & AJAX - Todo List Model Service: bl/music-service.is doc.querySelector('.frame :nththen(SC, FC) Operator wird verwendet um success / error events zu handlen. Verkettung Bootstrapper: ui/bootstrapper.js </div> 2, 3 → NodeList [div#foo1.foo] <script src="javascripts/todo-service.js" defer></script>
<script src="javascripts/index-controller.js" defer></script> child(1)'); //4 beliebig möglich, indem SC/FC ieweils wieder ein Resultat zurückgibt. chitd(1)'); //4
doc.querySelectorAll('div')[1]; //5
doc.querySelectorAll('div > div')[0]; //6 1,4,5,6 → <div id="foo1" export class MusicService import (MusicService) from '../bl/music-service.js'; function SC/FC(result) {...} - result enthält das erhaltene Resultat von promise constructor() [] (* import /IndexController\ from ' /controllers/index-controller is' class="foo"></div> <script id="todo-template" type="text/x-handlebars-template">
 <div class="todo-item" data-id="{{id}}"> .catch fängt Fehler von SC/FC ab. Wenn ein Fehler auftritt, sucht promise einen Operator This context - Ojbect/Function export class Bootstrapper (/* ... */ V ctass="todo-item" data-id="{{Id}}">
{##if done}}
<input type="checkbox" data-done="true" checked>
{{else}} in der Chain, welcher eine FC Fkt. unterstützt. Der Rückgabewert von FC wird verwendet, function makeCntFn(start) function makeCntObj(start) renderIndex() { return () => { return start++; } return {getNext: () => start++}; um die Kette wiederherzustellen. Wenn kein FC definiert ist → console.error Output gener. const musicService = new MusicService() controllers ui/controllers/index-controller.is const indexController = new IndexController input type="checkbox"> Promise export class IndexController (musicService): {{/if}} const cEn1 = makeCntEn(1): const cutOhit = makeCutOhi(2): constructor() () /* ... Wird new Promises in SC/FC zurückgegeben, wird ein weiterer asynchroner Schritt in die const cFn2 = makeCntFn(10); const cntObj2 = makeCntObj(20); {{title}} indexController.indexAction(): console.log(26, cntObj1.getNext()); //2
cntObj1.start = 0;
console.log(27, cntObj1.getNext()); //3
console.log(28, cntObj2.getNext()); //20 </div> console.log(22, cFn1()); //1 console.log(23, cFn1()); //10 console.log(24, cFn1()); //2 Kette hinzugefügt. Jeder promise repräsentiert die Vervollständigung vom vorherigen asyn. </script>
k href="stylesheets/style.css" rel="stylesheet"> const promise1 = new Promise((resolve, reject) => { setTimeout(() => { heads resolve('data resolved'); }, 1000); });
promise1.then((value) => { // successCallback console.log(`"\${value}"`); //"data Custom Validation <div id="todo-container"> </div> Wieso vernachlässigt? Niemand Verantwortlich, unklar wo starten, unklar über Outcome syg resolved" }, (err) => { // failureCallback }); <script> function checkFileSize() { <mark>minification</mark> (meta-informationen rausnehmen), grösse verkleinern mit lossless kompression (gzipdocument.addEventListener('DOMContentLoaded', () => indexAction());
</script> var FS = document.getElementById("FS"); Promise Handler: catch(failureCallback), then(successCallback, failureCallback). zonfli, br-brotli), replace animated sif with video. finally(finallyCallback) Vector vs Raster images: vector (geometric shapes), raster (imgs, photos, landscape) if (files.length > 0) {
 if (files[0].size > 75000) { // Check the constraint Resolution: raster – high resolution → higher the size (not for vector images→unabhängig) Lossles Async: async function xv() - gibt immer ein Obi, von Promise zurück, Missing async breaks FS.setCustomValidity("The selected file must not be larger than 75 kB"); png, gif) vs lossy (jpg): lossy (eliminate unseen bits) - lossless (recreate exactly) JPG encoders: -- index-controller.js --> promise chain. Await: beim Aufruf einer async Function oder in einer async Fkt. guetzli (img 20-30% smaller than JPEG), WebP (25-34% smaller than JPEG) → guetzli better with const indexTemplateCompiled = Handlebars.compile(document.getElementById('todo-template').innerHTML); Codeausführung pausiert bei der Linie & wartet bis Promise fertig ist & value small imgs - WebP with large ones, zopfli (5-10% small than PNG), Progressive JPEG: load image sync function renderIndexView() { zurückgegeben wird. Missing await results in fire-and-forget calls successive waves, load whole image but first with bad quality before the quality improves → FS.setCustomValidity(""); const todos = await todoService.getAll();
todoContainer.innerHTML = todos.map(x => indexTemplateCompiled (x)).join('') async function openRead() { try {
 const fileContent = await fileSystem.readFile(',/toRead.txt', {encoding: 'utfdecreases page load time HTML5-srcset: listet alternative Bildgrößen als Set von Bildern, Browser r document.getElementById("FS").onchange = checkFileSize; kann aussuchen. Image optimization: prefer vector formats, minify δ compress sye assets, pick bes }); console.log(`\${fileContent}`);
} catch (err) { console.log(`Error: \${err}`); } } raster img format, experiment with quality settings for raster imgs, remove image metadata QUIC: Promise/Callback/Fetch can multiplex requests: one stream does not affect others. Reduces RTT from 3 to 1 Minifiction: unction initEventHandlers() { function ajaxCall(url){
 const promiseObi = new Promise(async sync function ajaxCall(url){ await openRead(): todoAddForm.addEventListener('submit', async (event) => {
 event.preventDefault(); remove unnecessary (spaces, new line, comments) character from source code → quicker try { Fetch API Bietet ein Interface für Zugriff & Bearbeitung von Teilen der HTTP-Pipeline. const response = await fetch(url. unction(resolve, reject) { download times, reduced bandwidth, reduced number of requests if files combined to one, names await todoService.add(todoTitleInput.value); try { {method: "GET"}); Arbeitet mit Promise API und async / await kann verwendet werden can be shortened Server Push: For each request to index.html, style.css and example.png is pushed renderIndexView(todoService); const response = await fetch(url. const data=await response.json(); to the client, location / { index index.html; http2 push /style.css; http2 push /example.png; } }); {method: "GET"});
 const data=await response.json(); return data: } catch(e) { throw new Error(e); Caching most important technology in keeping sites scalable and fast hata stream hergestellt
movies = await moviesResponse.json(); //wait for downloaded data, then parse todoContainer.addEventListener('click', async (event) => { resolve(data); ETag: Request 200 Response with ETag: "x234dff". Browser stores /file with key x234dff in cache const id = event.target.closest('.todo-item').dataset.id;
if (event.target.dataset.done) { } catch(e) { catch (e){ console.log(`Error occurred: \${e}`);
finally{ console.log(`Download finished`); } with max-age: 120s const btn = doc.getElementBvId("btn") reject(é): event.target.dataset.done = false; await todoService.change(id, {done: Boolean(false)}); btn.addEventListener("click", Landing Pages return promiseObi: function() {
 const URL="https://git.com/users/j";
 ajaxCall(URL).then(sucCb).then(tst); Request Options: fetch() has 2 parameter to set Settings & HTTP Headers l else f Common startup mistake: tell story of startup instead focus on customer journey event.target.dataset.done = true; Data Upload json to server Calling a cross-origin domain CORS Example const btn = doc.getElementById("btn"); Ouick wins: Beantwortung von Einwänden, focus on "vou" not "we". await todoService.change(id, {done: Boolean(true)}); fetch(serverUrl, { method: 'POST', //POST(new) / PUT(update) headers: {
'Content-Type': 'application/json' // must match the body data format. }, btn.addEventListener("click". DOs: define value offerin in one line, explain benefits (2-3 sentences), show difference from function() {
 const URL="https://git.com/users/j"; unction sucCb (successCode){ }); competitors, show solutions you offer, focus on language of the client, avoid superlatives Donts: no body: JSON.stringify(data) //JSON passed to server }) console.log(successCode): clear call to action (don't know where to click), not having a responsive design, too much chaos ajaxCall(URL).then(succCb, errCb); return successCode; fetch(crossOriginUrl, { method: 'POST', //Any HTTP method to use (enough whitespaces), inconsistent design, misplacement of sign-up form, lack of customer focus / cors allows requests to other domains credentials: 'include', // includes function indevAction() / unction succCb (successCode) / function tst(successCode){
 console.log(successCode); console.log(successCode); renderIndexView(): A/B testing: show 2 versions of same webpage, higher conversion wins. Process: find pages with low conversion/high drop-offs, identify goals (e.g. more signups), Hypothesis: how to achieve the function errCb(errorCode) { console.log(errorCode) Handlebars Templating goal?, create improvement, deploy 2 versions & wait for data, analyze data <!--todo-service.js -->
const HEADERS = { 'Content-Type': 'application/json' }; Separate representation logic (HTML) from program Code (JS) Fetch mit Promise then Chain <script id="entry" type="text/x-handlebars-template">title</script> Navigation - horizontal * { padding: 0; margin: 0; box-sizing: border-box; } fetch(url + '2' + nath, {credentials: 'include'}) {{#each this}}...{{/each}} - {{#if ...}}...{{else if ...}}...{{else}}...{{/if}} - {{@index}} (private var) async getAll() {
 const request = await fetch(BASE_URL); .then(response => response.json())
.then(processReturnedJsonFn); <nav>WebshopWarenkorlo
// li>
// ol></nav>
nav > ol { display: block; } - nav > ol > li { display: inline-block; } - a { HTML return await request.json(); Templat Asvnc/Await e (head) {{/each}} Fix: nav { position:fixed; top: 0px; bottom: 0px; left: 0px; right: 0px; } async function openRead() { async add(title) {
 const response = await fetch(BASE_URL, {method: 'POST', headers: HEADERS, </script> <div id="song-container"></div> * { box-sizing: border-box; } HTML ToolTip - show on hover, relative to button const fileContent = await fileSystem.readFile('./toRead.txt', body: JSON.stringify({title})});
 const data = await response.json(); {encoding: 'utf-8'});
console.log(`Read: \${fileContent}`); <button data-tooltip="click me please!">Click me</button> (body) <script> //am Ende des Body [data-tooltip] { position: relative; } - button:hover:after { content: attr(data-tooltip); position: absolute: top: 102%; left: 101%; } document.addEventListener('DOMContentLoaded',() => { indexAction();}) catch (err) console.log(`Error: \${err}`); async change(id, todo) { = document.getElementById("template").innerHTML; ıs Selektoren Styling const request = await fetch(`/api/todos/\${id}`, {method: 'PATCH', eaders: HEADERS, body: JSON.stringify(todo) }); = Handlebars.compile(tempSrc); dontion == nonleguisionset(compac Links ohne "href"-Attribut: a:not([href]) { text-decoration: line-through;} Song Shop - DOM + bubbling const data = await response.json(); Alle h1 nicht direkt unter dem Body: body * h1 { font-size: 8px; } const songs = new SongCollection([
new Song(2, "California Girls", "Beach Boys", 2, 5.00),
new Song(3, "How Deep Is Your Love", "Bee Gees", 1, 6.50),]);

return this.songs.getAllSortedByRating().map(song =>

data-song-id="\${song.id}"> <\li data-song-id="\${song.id}"> <h3>\${song.title} von \${song.artist} <button class="add-to-

function tryGetOpponentNameFromServer() {

if (opponentName) {

tryGetOpponentNameFromServer();

onentNameFromServerAsynch(opponentName => {

document.querySelector('#oppenent-name').innerHTML = opponentName; startShootingButton.disabled = false;

} else { setTimeout(tryGetOpponentNameFromServer, 10000); }

Classes - class XY { constructor() { this.member1 = "123"; }} - Fkt. Ohne function deklarieren As Expression: const foo = class { constructor() {} bar() { return "hello world":}}: static method: static classMethod() { return 'hello'; } → usage: Foo.classMethod(); Inheritance: If parent class has constructor → derived class must call super()

[{title: "title1", artist:"noname"}, {title:"foo",

Erster p nach h1: h1 + p { font-style: italic; }

Object = Copy by reference; Primitives = Copy by Value

Elemente durch 3 aber nicht 2 teilbar: ul > li:nth-child(3n):not(:nth-child(2n)) {

<time datetime="2016-02-11 12:45">Donnerstag, 11. Februar 2016, 12:45</time>

div[class="field occupied"] → divs mit den 2 Klassen, Reihenfolge wichtig

div.field.occupied → alle divs die diese beiden Klassen oder mehr haben