Grundlagen "Web-Engineering"

World Wide Web

"Das **World Wide Web** (englisch für „weltweites Netz“, kurz Web oder WWW) ist ein über das **Internet** abrufbares System von elektronischen Hypertext-Dokumenten, sogenannten **Webseiten**, welche mit **HTML** beschrieben werden.

Sie sind durch **Hyperlinks** untereinander verknüpft und werden im Internet über die Protokolle **HTTP oder HTTPS** übertragen.

Der Benutzer kann dann den **Hyperlinks** auf der angezeigten Webseite folgen, die auf andere Webseiten verweisen. So ergibt sich ein weltweites Netz aus Webseiten. Das Verfolgen der Hyperlinks wird auch als **„Surfen im Internet“** bezeichnet.

Die Webseiten enthalten meist **Texte**, oft mit **Bildern** und **grafischen Elementen** illustriert. Häufig sind auch **Videos**, **Tondokumente** oder **Musikstücke** eingebettet."

Zum Aufrufen von Inhalten aus dem World Wide Web wird ein **Webbrowser** benötigt, der z. B. auf einem **PC** oder einem **Smartphone** läuft. Mit ihm kann der Benutzer die auf einem beliebigen, von ihm ausgewählten **Webserver** bereitgestellten Daten herunterladen und auf einem geeigneten Ausgabegerät wie einem **Bildschirm** oder einer **Braillezeile** anzeigen lassen.

(vgl. https://de.wikipedia.org/wiki/World_Wide_Web)

- "Hyperlinks" / URL
- Client - Server
- HTTP
- Webbrowser
- PC / Smartphone
- HTML
- Texte, Bilder, Video, Ton

Engineering

Engineering is the practice of using **natural science**, **mathematics**, and the **engineering design** process to solve **technical problems**, increase **efficiency** and **productivity**, and **improve systems**.

The **engineering design process** is a common series of steps that engineers use in creating **functional products and processes**. The process is **highly iterative**[...]

It is a **decision making process** in which the **basic sciences** [...] are **applied** to convert resources optimally **to meet a stated objective**.

Among the fundamental elements of the design process are the **establishment of objectives and criteria**, **synthesis**, **analysis**, **construction**, **testing** and **evaluation**.

(vgl. https://en.wikipedia.org/wiki/Engineering, https://en.wikipedia.org/wiki/Engineering_design_process)

- **Anwendung von Wissenschaften**
- Lösen von technischen Problemen, erhöhen der Effizienz, Optimierung von Systeme
- **Iterativer Prozess**
- Entscheidungsverfahren
- Bestimmen der Anforderungen und **Analyse**, **Herstellung**, **Überprüfung** und **Evaluation**

"Hyperlinks" / URL

```
https://www.philipackermann.de:80/books/web.html?language=de#chapter7
```

Client - Server

- Client: Browser -> Eher einfache Maschine (Software und Hardware), meistens in direktem Zugriff des Users
- Server: Webserver -> eher leistungsfähig (Software und Hardware), i.d.R. im Rechenzentrum

HTTP

```
$ telnet www.perdu.com 80
Trying 208.97.177.124...
Connected to www.perdu.com.
Escape character is '^]'.

GET / http/1.1
Host: www.perdu.com

HTTP/1.1 200 OK
Date: Sat, 17 Aug 2013 12:14:56 GMT
Server: Apache
Accept-Ranges: bytes
X-Mod-Pagespeed: 1.1.23.1-2169
Vary: Accept-Encoding
Cache-Control: max-age=0, no-cache
Content-Length: 204
Content-Type: text/html

<html><head><title>Vous Etes Perdu ?</title></head><body><h1>Perdu sur l'Interne
t ?</h1><h2>Pas de panique, on va vous aider</h2><strong><pre> * <----- vous
&ecirc;tes ici</pre></strong></body></html>

Connection closed by foreign host.
$
```

Verbindungsaufbau zum Server

HTTP-Anfrage

Serverantwort: Header

Serverantwort: Nachrichtenrumpf

Verbindungsende

Webbrowser

Browser Wars
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
Others (Opera, Safari, PSP...)
Firefox, Mozilla, Camino, etc.
Netscape classic
Internet Explorer for Windows

StatCounter Global Stats
Browser Market Share Worldwide from Jan 2009 - Mar 2023
70%
56%
42%
28%
14%
0%
statcounter
Jan 2010
Jan 2011
Jan 2012
Jan 2013
Jan 2014
Jan 2015
Jan 2016
Jan 2017
Jan 2018
Jan 2019
Jan 2020
Jan 2021
Jan 2022
Jan 2023
Chrome
IE
Firefox
Safari
Opera
Android
UC Browser
Samsung Internet
Edge Legacy
Other (dotted)

StatCounter Global Stats
Desktop vs Mobile vs Tablet vs Console Market Share Worldwide from Jan 2009 - Mar 2023
100%
80%
60%
40%
20%
0%
statcounter
Jan 2010
Jan 2011
Jan 2012
Jan 2013
Jan 2014
Jan 2015
Jan 2016
Jan 2017
Jan 2018
Jan 2019
Jan 2020
Jan 2021
Jan 2022
Jan 2023
Desktop
Mobile
Tablet
Console

PC / Smartphone

Webseiten strukturieren mit HTML

```html
<!DOCTYPE html>
<html lang="de">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Titel</title>
</head>
<body>

</body>
</html>
```

Aufbau von Webapplikationen

Software Stacks

LAMP

MERN

MongoDB, Express.js, React, Node.js

Programmiersprachen

https://survey.stackoverflow.co/2023/#section-most-popular-technologies-programming-scripting-and-markup-languages, 23.04.24

Datenbanken

https://survey.stackoverflow.co/2023/#section-most-popular-technologies-databases, 23.04.24

Cloud Plattformen

https://survey.stackoverflow.co/2023/#section-most-popular-technologies-cloud-platforms, 23.04.24

Web Frameworks

https://survey.stackoverflow.co/2023/#section-most-popular-technologies-web-frameworks-and-technologies, 23.04.24

State of JavaScript 2022, 21.06.2023

Texte, Bilder, Video, Ton

Werkzeuge

Tools.md

Inspektor | Konsole | Debugger | Netzwerkanalyse | Stilbearbeitung | Laufzeitanalyse | Speicher

HTML durchsuc

```
<!DOCTYPE html>
<html lang="de">  scroll
  <head> ··· </head>
  <body>
    <header> ··· </header>
    <nav> ··· </nav>
    <div class="clear"></div>
    <main> ··· </main>
    <footer> ··· </footer>
  </body>
</html>
```

html > body

Stile filtern :hov .cls

```
Element {                                          Inline
}
body, h1, h2, h3, h4,   dark.css:7 @(prefers-color-scheme: dark)
a, p, blockquote cite {
    color: #ddd;
}
body {   dark.css:2 @(prefers-color-scheme: dark)
    background-color: #333;
}
body {                                             style.css:8
    font-family: sans-serif;
}
body {                                             normalize.css:23
    margin: 0;
}
```

Geerbt von html

```
html {                                             normalize.css:11
    line-height: 1.15;
    -webkit-text-size-adjust: 100%;
}
```

Layout | Berechnet | Änderungen | Schriftarten

Flexbox

Flex-Behälter oder -Element auswählen, um fortzufahren.

Raster

Raster hervorheben

- ul.wp-block-gallery.columns-2.is-cropped
- ul.wp-block-gallery.columns-3.is-cropped
- ul.wp-block-gallery.columns-3.is-cropped
- ul.wp-block-gallery.columns-3.is-cropped

Anzeigeeinstellungen für Raster

- [x] Zeilennummern anzeigen
- [ ] Bereichsnamen anzeigen
- [ ] Unendliche Linien

Box-Modell

margin 0
border 0
padding 0
823×2397.2

Webseiten gestalten mit CSS

Layoutkonzepte

- http://info.cern.ch/hypertext/WWW/TheProject.html
- Framesets
- Tabellen
- Cascading Style Sheets (CSS)
- Fixed vs. Liquid Layout
- Responsive Webdesign
- Device Agnostic
- Mobile First

Box Model

- `box-sizing: content-box` : `width` bezieht sich nur auf den content (blau, `300x150`)
- `box-sizing: border-box` : `width` bezieht sich auf content + padding + border (blau, violett, grau, `300+2*40+2*5` für die Breite)

Einheiten

- Absolute Grössen: `px` (`cm` , `mm` , ...) -> sparsam verwenden
- Relative Grössen
  - `em` : Schriftgrösse des Elternelements
  - `rem` : Schriftgrösse des Wurzelelements
  - `vw` , `vh` : viewport breite, viewport höhe

Farben

CIE 1931 Farbraum

Vergleich Farbräume

Farben in CSS

sRGB Farbraum

- Farbnamen: `color: darkblue;`
- Hex-Werte: `color: #ffa500;`
- RGBA-Werte (mit Deckkraft): `color: rgba(169, 169, 169, 0.5)`
- HSL-Werte (Hue, Saturation, Lightness): `color: hsl(60, 100, 50)`

Alle sichtbaren Farben

- LCH (Lightness Chroma Hue / Opacity): `color: lch(29.2345% 44.2 27 / 0.5)`
- Oklch: `color: oklch(40.1% 0.123 21.57)`
- CIELAB (Lightness, red-green, blue-yellow): `color: lab(29.2345% 39.3825 20.0664);`
- Oklab: `color: oklab(40.1% 0.1143 0.045);`

https://codepen.io/web-dot-dev/pen/poZgXxy

Webseiten interaktiv machen mit JavaScript

vgl.: Douglas Crockford (2018): How JavaScript Works, virgule solidus

How Class Free Works

- Klassen sind syntaktischer Zucker, d.h. sie bieten keine Funktionalität, die nicht mit anderen Mitteln erreicht werden kann.
- Sie verhalten sich anders als Klassen in C++, Java oder C#. Das kann verwirrend sein.

"Composition over Inheritance"

- Vererbung ist weniger zentral als manche Sprachen oder Kurse vermitteln.
- Vererbung bringt auch einige Probleme mit sich, da die Klassen sehr eng gekoppelt sind und nicht explizit klar ist, welche Methoden aufgerufen werden.
- Komposition ist sehr leistungsfähig.

Closures

Verschachtelte Funktionen können auf Variablen aus den äusseren Funktionen zugreifen. Auch nach deren Ausführung.

```
function init() {
  var name = "Mozilla"; // name is a local variable created by init
  function displayName() {
    // displayName() is the inner function, that forms the closure
    console.log(name); // use variable declared in the parent function
  }
  displayName();
}
init();
```

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Closures

Folgende Struktur wird empfohlen:

```
function counter_constructor() {
    // private property
    let counter = 0;

    // composition
    const reuse = other_constructor();

    function up() {
        counter -= 1;
        return counter;
    }

    function down() {
        counter += 1;
        return counter;
    }

    // freeze to make the object immutable
    return Object.freeze({
        // make functions up and down public
        up,
        down,
        // expose goodness property from another object
        goodness: reuse.goodness
    })
}
```

Asynchronität

- JavaScript wurde primär für User-Interaktionen entwickelt.
- Asynchronität ist deshalb ein zentrales Sprachfeature.
- Es gibt verschiedene Möglichkeiten für asynchronen Code:
  - Callbacks
  - Promise
  - `async` / `await`

`async` / `await` ist verwirrend, weil damit Code produziert wird, der synchron aussieht, aber asynchron funktioniert.

Callback-Funktionen

- Callback-Funktionen werden als Parameter einer Funktion übergeben und von dieser aufgerufen.
- Die sogenannt "Callback-Hell", gemeint ist die Verschachtelung von Callbacks in Callbacks, sollte vermieden werden.

```
function foo(callback) {
    // some functionality

    callback(value);
}

foo((value) => {
    // runs after "some functionality"
})
```

Promise

Promises können klarer sein als Callbacks, sind aber auch weniger explizit und potenziell verwirrend.

```
const p1 = new Promise((resolve, reject) => {
    // some functionality

    resolve("Success!");
});
p1.then((value) => {
        // runs after "some functionality"
    }
);
```

Single-Page-Applikationen implementieren

Typische Dokumentstruktur

- **Docs**: README.md, CONTRIBUTING.md, LICENCE.md, etc
- **Configs**: `.gitignore`, `.talismanrc`, `commitlint.config.js`, `eslint.config.js`, `.dockerignore`, `jest.config.js`, `jsconfig.json`
- **Build**: `Dockerfile`, `docker-compose.yml`
- **Dependencies**: package.json, package-lock.json, node_modules
- **Render/Compilation outputs**: `/dist`, `/build`, `/coverage`
- Source Code: `/src`
- **Build Pipeline**: `.gitlab-ci.yml`

Progressive Web Apps

Local First

1. No spinners: your work at your fingertips
2. Your work is not trapped on one device
3. The network is optional
4. Seamless collaboration with your colleagues
5. The Long Now
6. Security and privacy by default
7. You retain ultimate ownership and control

https://www.inkandswitch.com/local-first/
https://localfirstweb.dev/

Die Performance von Webanwendungen optimieren

Browser Tools: Netzwerkanalyse

Browser Tools: Laufzeitanalyse

Firefox 117 – macOS 13.4.1 | Vollständiger Zeitraum (19s) | implementation=js&thread=g&v=10 | es Profil hochladen | Dokumentation

6 / 26 Tracks

1s 2s 3s 4s 5s 6s 7s 8s 9s 10s 11s 12s 13s 14s 15s 16s 17s 18s

Parent Process
PID: 44119

Network

Renderer

racoon.pages.dev
PID: 46861

Network

Aufrufbaum | Flammendiagramm | Stack-Diagramm | Markierungsdiagramm | Markierungstabelle | Netzwerk

Alle Frames | JavaScript | Native | Aufrufstack umkehren | Stacks filtern: Filterbegriffe eingeben

Vollständiges „https://racoon.pages.dev"

Gesamt (Teilmengen)		Eigene	
28%	90	4	EvaluateModule
12%	38	3	Br https://racoon.pages.dev/_app/immutable/nodes/2.34b35c83.js:80:708954
8.2%	26	—	le$6 https://racoon.pages.dev/_app/immutable/nodes/2.34b35c83.js:80:709020
7.9%	25	1	initGenrand https://racoon.pages.dev/_app/immutable/nodes/2.34b35c83.js:80:626929
7.6%	24	6	multiplication32 https://racoon.pages.dev/_app/immutable/nodes/2.34b35c83.js:80:626820
3.8%	12	—	addition32 https://racoon.pages.dev/_app/immutable/nodes/2.34b35c83.js:80:626759
3.8%	12	12	unsigned32 https://racoon.pages.dev/_app/immutable/nodes/2.34b35c83.js:80:626617
1.9%	6	6	unsigned32 https://racoon.pages.dev/_app/immutable/nodes/2.34b35c83.js:80:626617
0.3%	1	1	Ti https://racoon.pages.dev/_app/immutable/nodes/2.34b35c83.js:80:626430
0.6%	2	1	ke$2 https://racoon.pages.dev/_app/immutable/nodes/2.34b35c83.js:80:710306
0.3%	1	1	Ui https://racoon.pages.dev/_app/immutable/nodes/2.34b35c83.js:80:670038
0.3%	1	—	Di https://racoon.pages.dev/_app/immutable/nodes/2.34b35c83.js:80:637448
0.3%	1	1	Ni https://racoon.pages.dev/_app/immutable/nodes/2.34b35c83.js:80:632447
0.3%	1	1	ar https://racoon.pages.dev/_app/immutable/nodes/2.34b35c83.js:80:681221
0.3%	1	1	ir https://racoon.pages.dev/_app/immutable/nodes/2.34b35c83.js:80:687109
0.3%	1	1	Ci https://racoon.pages.dev/_app/immutable/nodes/2.34b35c83.js:80:621293
0.3%	1	1	_i https://racoon.pages.dev/_app/immutable/nodes/2.34b35c83.js:80:634977
6.3%	20	—	onNewScript resource://devtools/server/actors/thread.js:2034:13

unsigned32
https://racoon.pages.dev/_app/immutable/node ...

Call node details

Traced running time		11.9ms
Traced self time		11.9ms
Running samples	4%	12
Self samples	4%	12

Categories	Running sample count	
JavaScript	100%	12

Categories	Self sample count	
JavaScript	100%	12

Implementation	Running sample count	
JS JIT (baseline)	17%	2
JS JIT (ion)	83%	10

Implementation — Self sample count

Bun

Bun.serve()
WebSocket
bun:sqlite
Server-side rendering React
HTTP requests per second (Linux x64)
66,706
32,921
13,967
Bun
v1.0
Deno.serve()
v1.36.2
Node.js
20.5.0
View benchmark

```
$ bun install
```

Bun is an npm-compatible package manager.

Installing dependencies from cache for a Remix app.

View benchmark

$ bun test
Bun is a test runner that makes the rest look like test walkers.
Bun
00.23s
Vitest
5x slower
01.91s
Jest+SWC
8x slower
03.07s
Jest+tsjest
18x slower
06.71s
Jest+Babel
20x slower
07.43s

https://bun.sh/

Energy, Time, Memory Comparision

Table 4: Normalized global results for Energy, Time, and Memory

Total					
	Energy (J)		Time (ms)		Mb
(c) C	1.00	(c) C	1.00	(c) Pascal	1.00
(c) Rust	1.03	(c) Rust	1.04	(c) Go	1.05
(c) C++	1.34	(c) C++	1.56	(c) C	1.17
(c) Ada	1.70	(c) Ada	1.85	(c) Fortran	1.24
(v) Java	1.98	(v) Java	1.89	(c) C++	1.34
(c) Pascal	2.14	(c) Chapel	2.14	(c) Ada	1.47
(c) Chapel	2.18	(c) Go	2.83	(c) Rust	1.54
(v) Lisp	2.27	(c) Pascal	3.02	(v) Lisp	1.92
(c) Ocaml	2.40	(c) Ocaml	3.09	(c) Haskell	2.45
(c) Fortran	2.52	(v) C#	3.14	(i) PHP	2.57
(c) Swift	2.79	(v) Lisp	3.40	(c) Swift	2.71
(c) Haskell	3.10	(c) Haskell	3.55	(i) Python	2.80
(v) C#	3.14	(c) Swift	4.20	(c) Ocaml	2.82
(c) Go	3.23	(c) Fortran	4.20	(v) C#	2.85
(i) Dart	3.83	(v) F#	6.30	(i) Hack	3.34
(v) F#	4.13	(i) JavaScript	6.52	(v) Racket	3.52
(i) JavaScript	4.45	(i) Dart	6.67	(i) Ruby	3.97
(v) Racket	7.91	(v) Racket	11.27	(c) Chapel	4.00
(i) TypeScript	21.50	(i) Hack	26.99	(v) F#	4.25
(i) Hack	24.02	(i) PHP	27.64	(i) JavaScript	4.59
(i) PHP	29.30	(v) Erlang	36.71	(i) TypeScript	4.69
(v) Erlang	42.23	(i) Jruby	43.44	(v) Java	6.01
(i) Lua	45.98	(i) TypeScript	46.20	(i) Perl	6.62
(i) Jruby	46.54	(i) Ruby	59.34	(i) Lua	6.72
(i) Ruby	69.91	(i) Perl	65.79	(v) Erlang	7.20
(i) Python	75.88	(i) Python	71.90	(i) Dart	8.64
(i) Perl	79.58	(i) Lua	82.91	(i) Jruby	19.84

https://haslab.github.io/SAFER/scp21.pdf