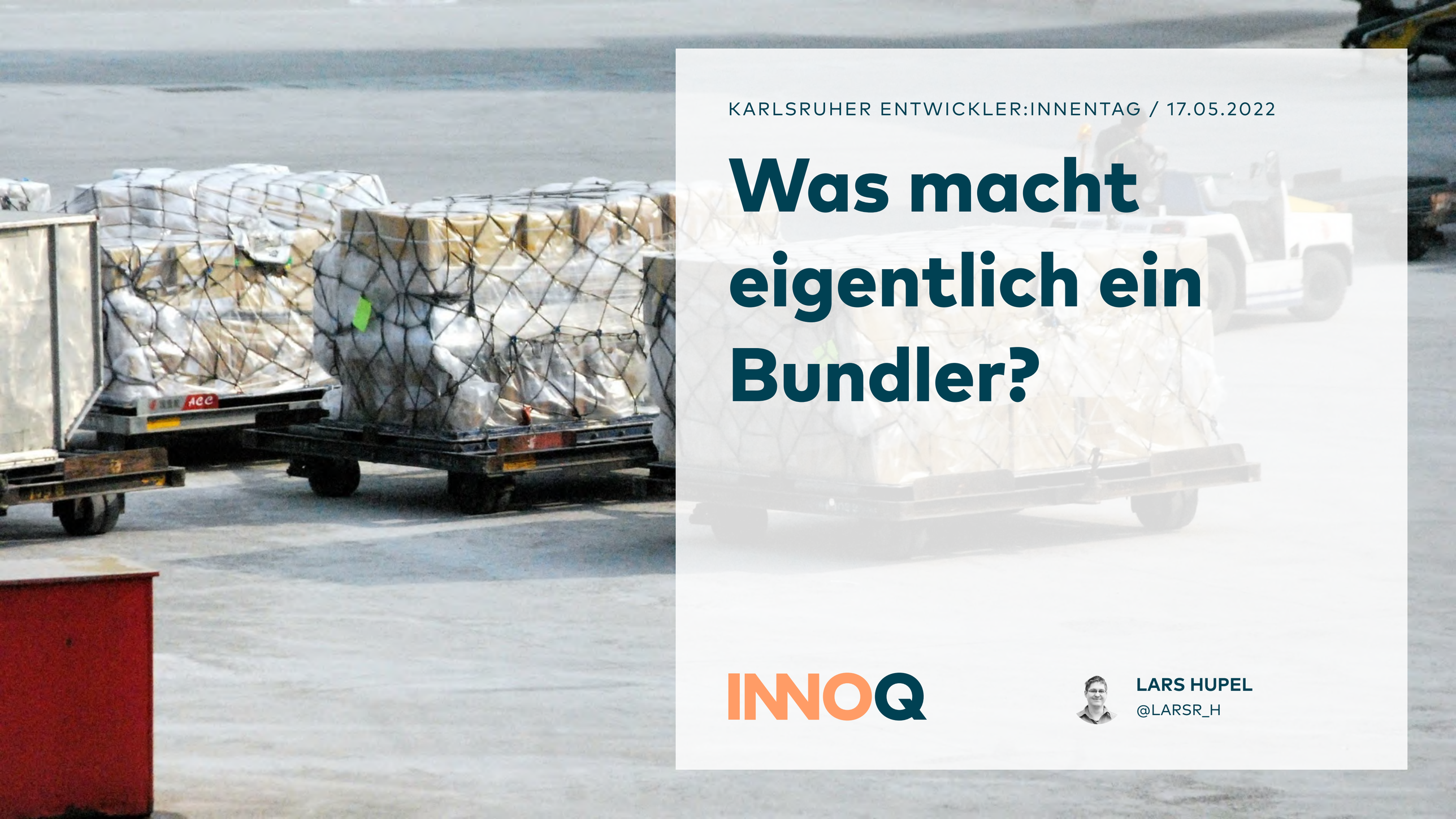


KARLSRUHER ENTWICKLER:INNENTAG / 17.05.2022

Was macht eigentlich ein Bundler?

INNOQ

LARS HUPEL
@LARSR_H


Vlad Yaitskiy, CC-BY-SA 2.0
innoq.com/assistance

webpack

rollup.js

Snowpack

PARCEL

esbuild

An extremely fast JavaScript bundler

BABEL

TS

React

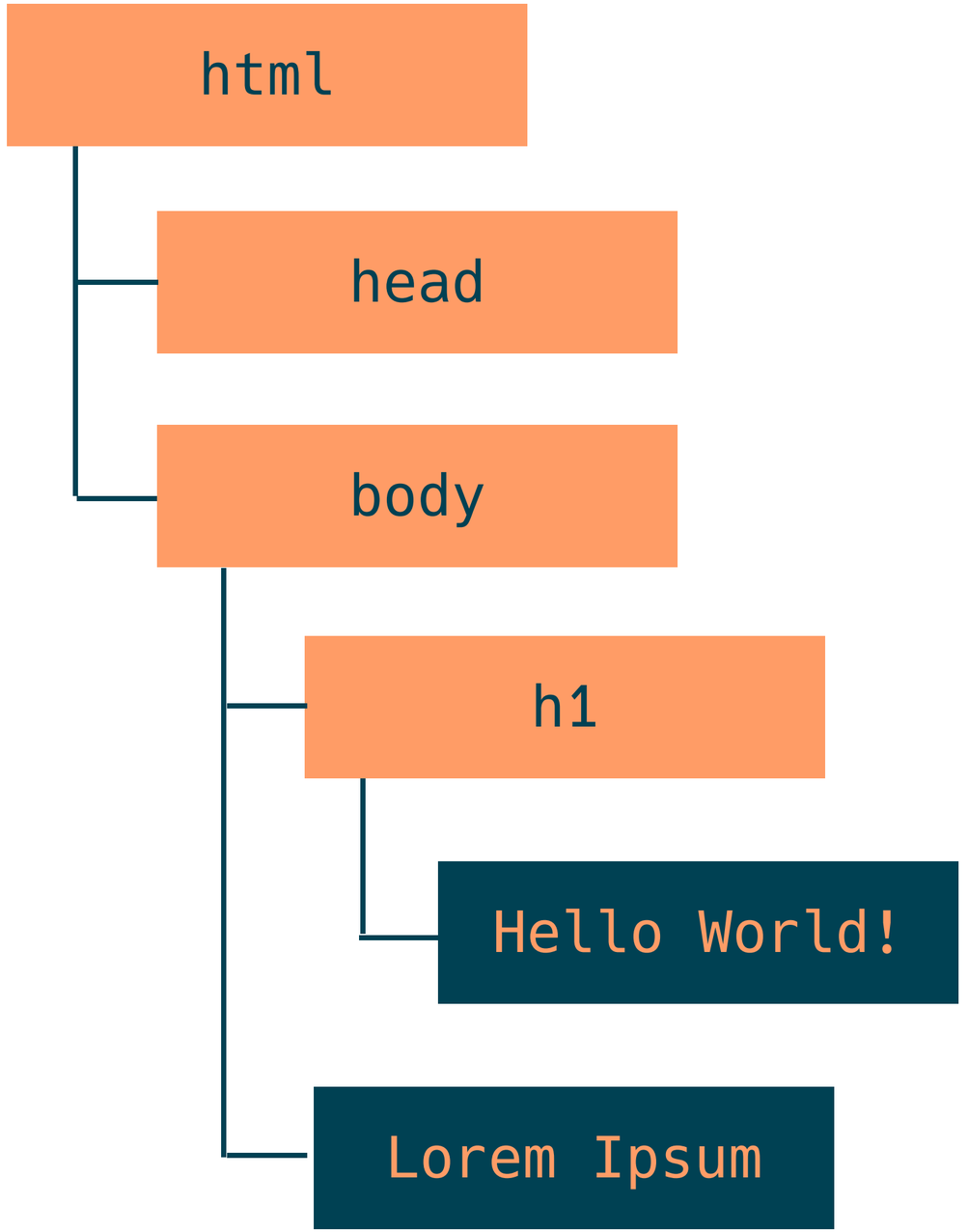

CONFIGURING
WEBPACK
PROGRAMMING
imgflip.com

Kapitel 1

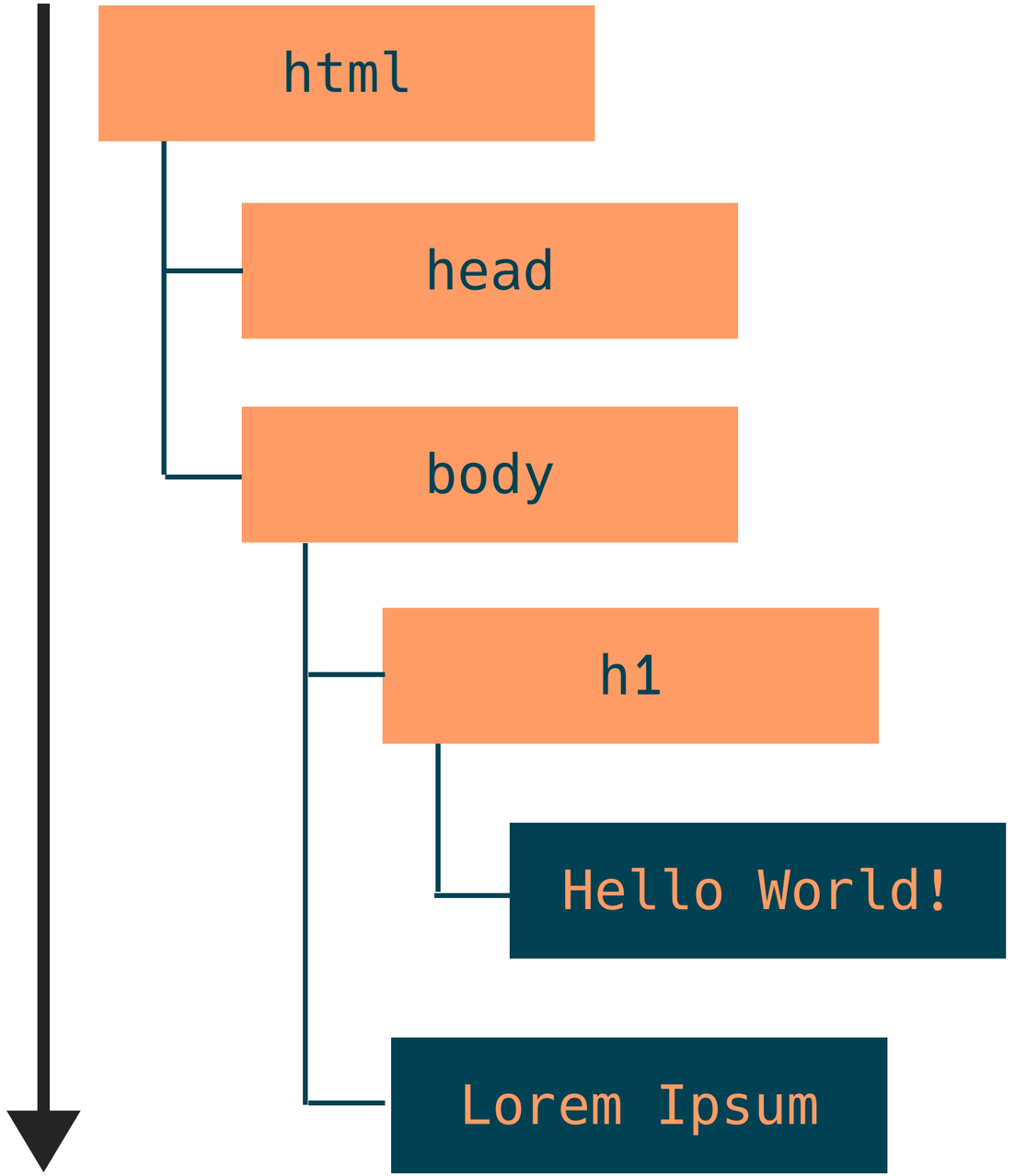
HTML

HTML-Grundlagen

```
<!DOCTYPE HTML>
<html>
  <head></head>
  <body>
    <h1>Hello World!</h1>
    Lorem Ipsum
  </body>
</html>
```

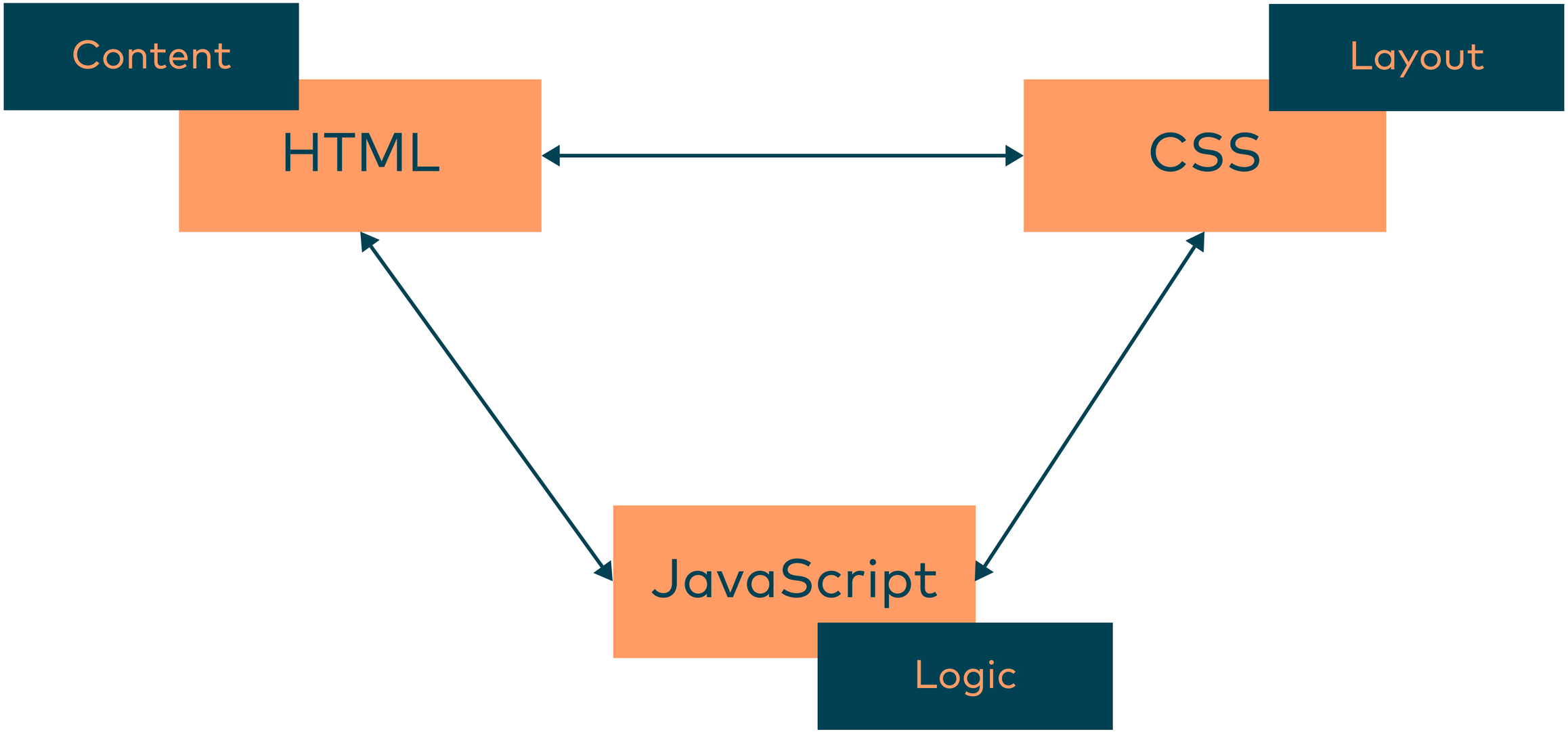


Browser arbeiten (meistens) linear



HTML kann viel mehr …

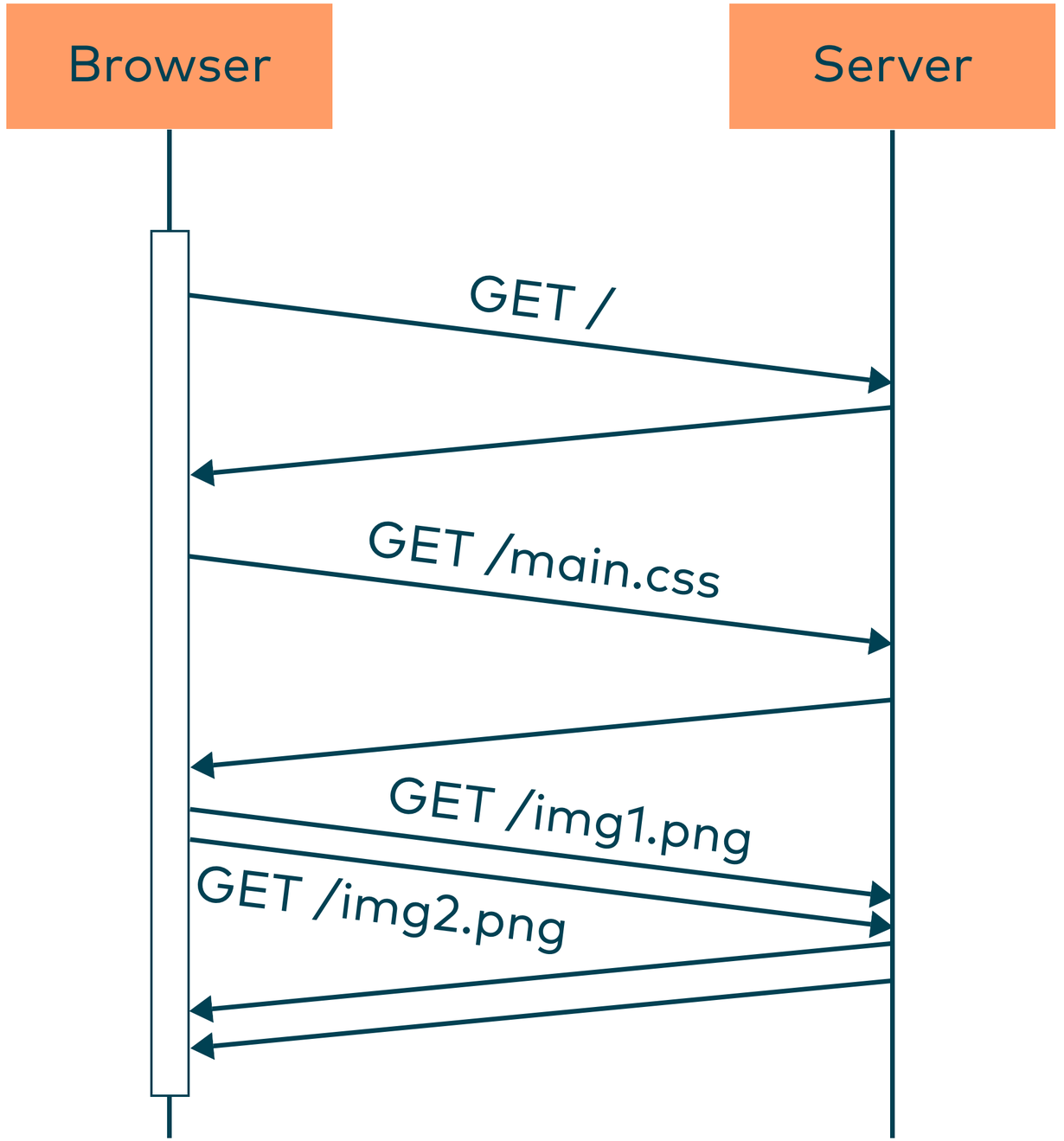
- JavaScript
- Grafiken
- Stylesheets
- Schriftarten
- `<iframe>`
- …

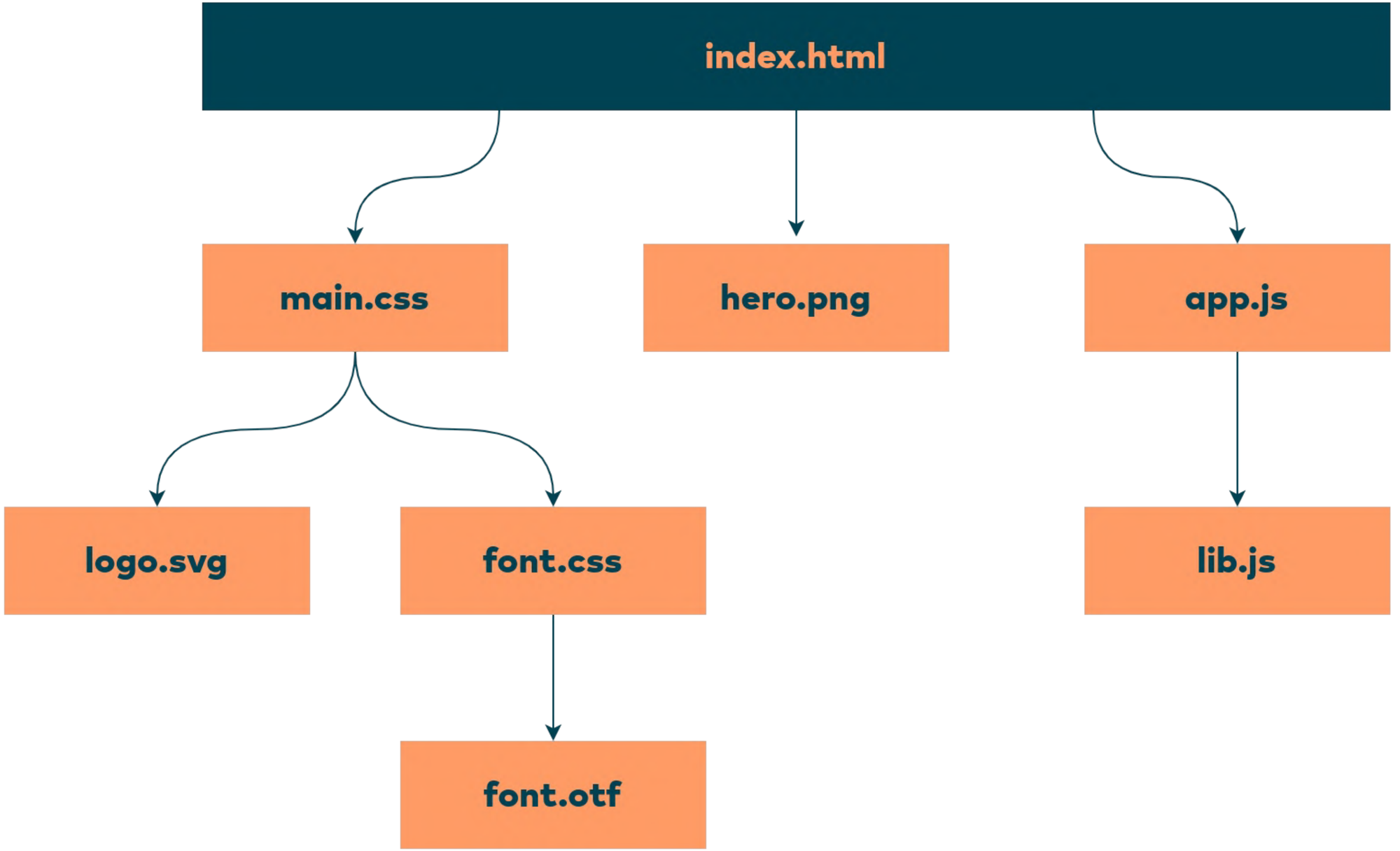

Content
HTML
CSS
Layout
JavaScript
Logic

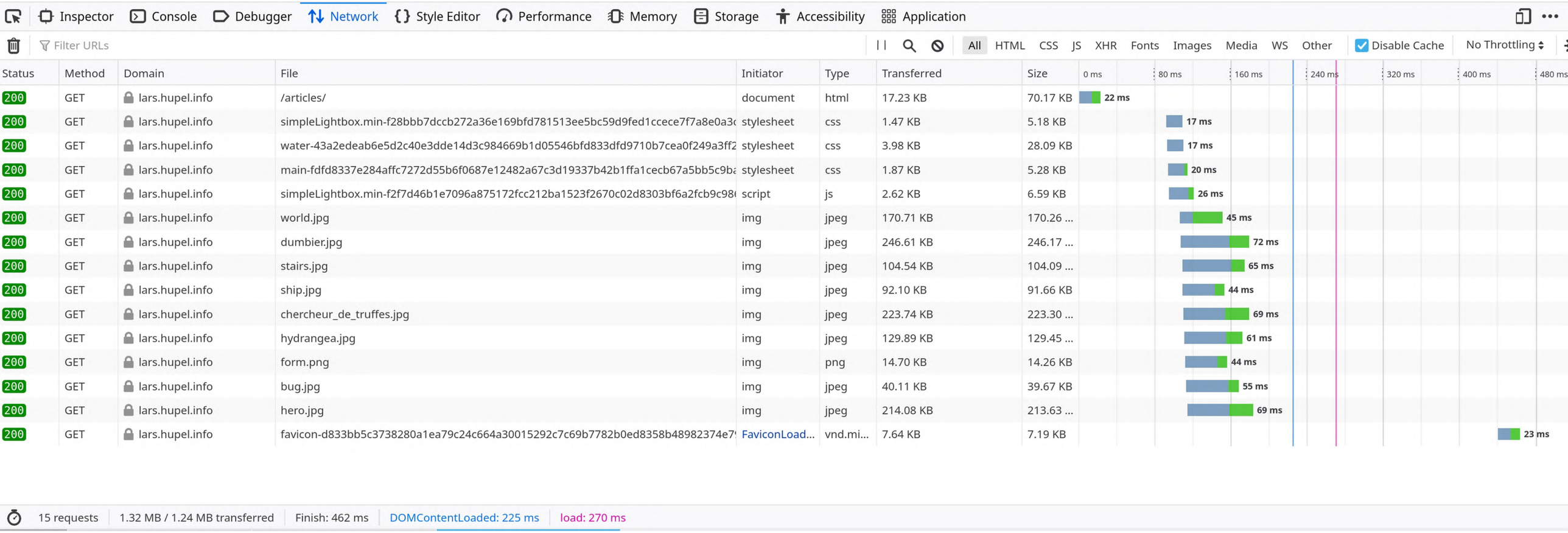
Kapitel 2

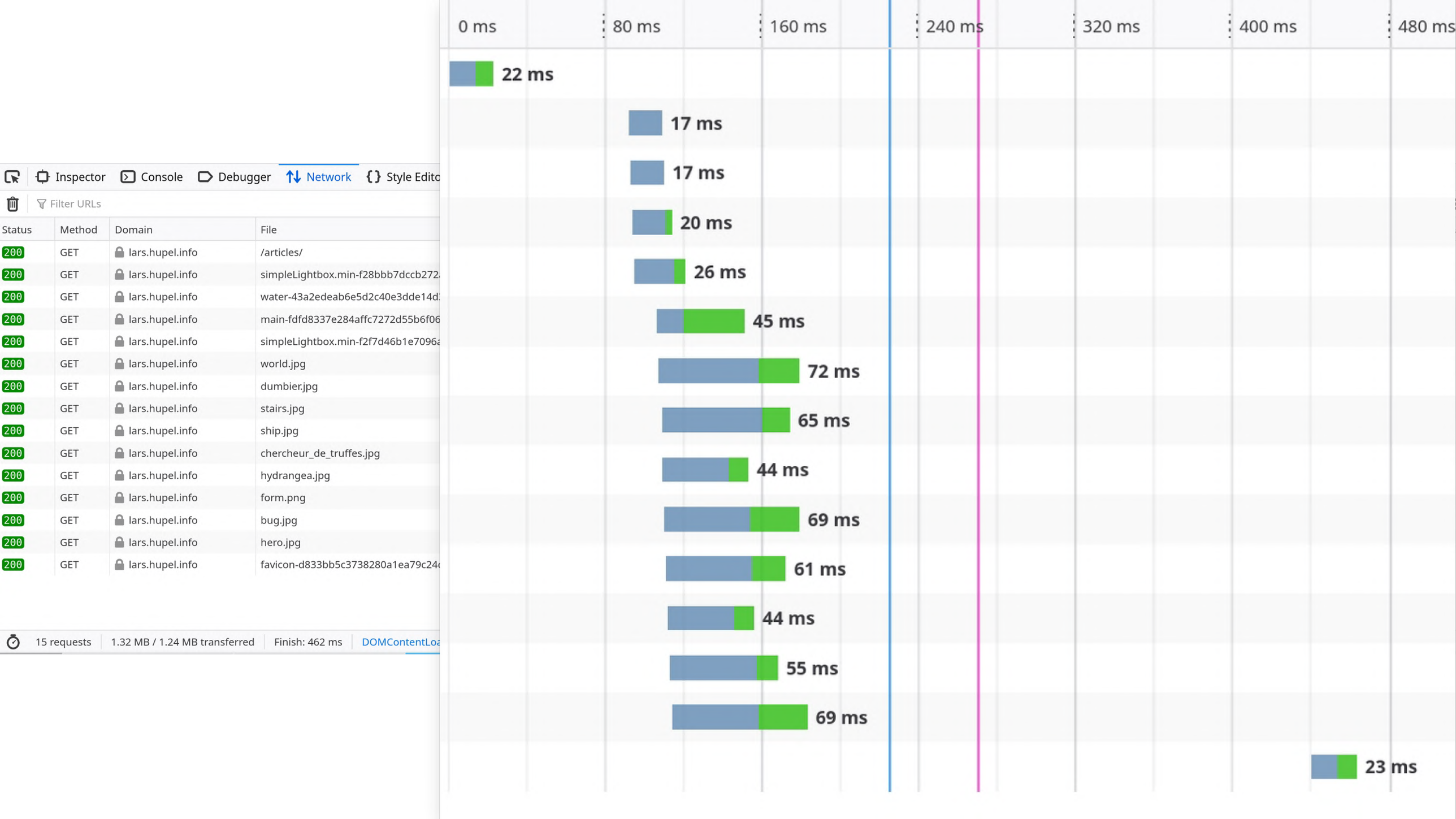
HTTP

HTTP-Grundlagen




index.html
main.css
hero.png
app.js
logo.svg
font.css
lib.js
font.otf


Inspector
Console
Debugger
Network
Style Editor
Performance
Memory
Storage
Accessibility
Application
Filter URLs
All
HTML
CSS
JS
XHR
Fonts
Images
Media
WS
Other
Disable Cache
No Throttling
Status
Method
Domain
File
Initiator
Type
Transferred
Size
0 ms
80 ms
160 ms
240 ms
320 ms
400 ms
480 ms
200 GET lars.hupel.info /articles/ document html 17.23 KB 70.17 KB 22 ms
200 GET lars.hupel.info simpleLightbox.min-f28bbb7dccb272a36e169bfd781513ee5bc59d9fed1ccece7f7a8e0a3c stylesheet css 1.47 KB 5.18 KB 17 ms
200 GET lars.hupel.info water-43a2edeab6e5d2c40e3dde14d3c984669b1d05546bfd833dfd9710b7cea0f249a3ff2 stylesheet css 3.98 KB 28.09 KB 17 ms
200 GET lars.hupel.info main-fdfd8337e284affc7272d55b6f0687e12482a67c3d19337b42b1ffa1cecb67a5bb5c9ba stylesheet css 1.87 KB 5.28 KB 20 ms
200 GET lars.hupel.info simpleLightbox.min-f2f7d46b1e7096a875172fcc212ba1523f2670c02d8303bf6a2fcb9c98 script js 2.62 KB 6.59 KB 26 ms
200 GET lars.hupel.info world.jpg img jpeg 170.71 KB 170.26 ... 45 ms
200 GET lars.hupel.info dumbier.jpg img jpeg 246.61 KB 246.17 ... 72 ms
200 GET lars.hupel.info stairs.jpg img jpeg 104.54 KB 104.09 ... 65 ms
200 GET lars.hupel.info ship.jpg img jpeg 92.10 KB 91.66 KB 44 ms
200 GET lars.hupel.info chercheur_de_truffes.jpg img jpeg 223.74 KB 223.30 ... 69 ms
200 GET lars.hupel.info hydrangea.jpg img jpeg 129.89 KB 129.45 ... 61 ms
200 GET lars.hupel.info form.png img png 14.70 KB 14.26 KB 44 ms
200 GET lars.hupel.info bug.jpg img jpeg 40.11 KB 39.67 KB 55 ms
200 GET lars.hupel.info hero.jpg img jpeg 214.08 KB 213.63 ... 69 ms
200 GET lars.hupel.info favicon-d833bb5c3738280a1ea79c24c664a30015292c7c69b7782b0ed8358b48982374e7 FaviconLoad... vnd.mi... 7.64 KB 7.19 KB 23 ms
15 requests
1.32 MB / 1.24 MB transferred
Finish: 462 ms
DOMContentLoaded: 225 ms
load: 270 ms


Inspector
Console
Debugger
Network
Style Editc
Filter URLs
Status
Method
Domain
File
200 GET lars.hupel.info /articles/
200 GET lars.hupel.info simpleLightbox.min-f28bbb7dccb272
200 GET lars.hupel.info water-43a2edeab6e5d2c40e3dde14d
200 GET lars.hupel.info main-fdfd8337e284affc7272d55b6f06
200 GET lars.hupel.info simpleLightbox.min-f2f7d46b1e7096a
200 GET lars.hupel.info world.jpg
200 GET lars.hupel.info dumbier.jpg
200 GET lars.hupel.info stairs.jpg
200 GET lars.hupel.info ship.jpg
200 GET lars.hupel.info chercheur_de_truffes.jpg
200 GET lars.hupel.info hydrangea.jpg
200 GET lars.hupel.info form.png
200 GET lars.hupel.info bug.jpg
200 GET lars.hupel.info hero.jpg
200 GET lars.hupel.info favicon-d833bb5c3738280a1ea79c24
15 requests
1.32 MB / 1.24 MB transferred
Finish: 462 ms
DOMContentLoa
0 ms
80 ms
160 ms
240 ms
320 ms
400 ms
480 ms
22 ms
17 ms
17 ms
20 ms
26 ms
45 ms
72 ms
65 ms
44 ms
69 ms
61 ms
44 ms
55 ms
69 ms
23 ms

HTTP-Performance

- seit 1991 zahlreiche Performance-Verbesserungen
- hauptsächlich getrieben von der Industrie
- Grundideen:
  - Anfragen parallelisieren
  - Verbindungen reduzieren
- nicht alle Anwendungen profitieren von HTTP/2+

1.1

Keepalive

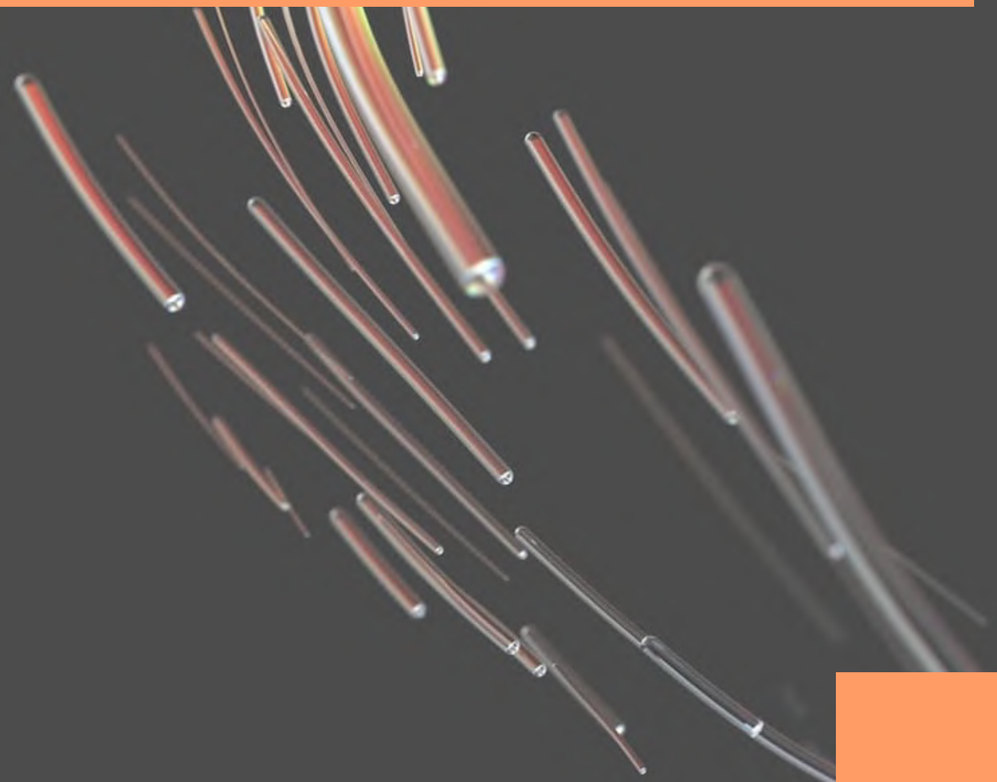
2.0

Multiplexing & Push

3.0

UDP+QUIC+TLS

Kapitel 3



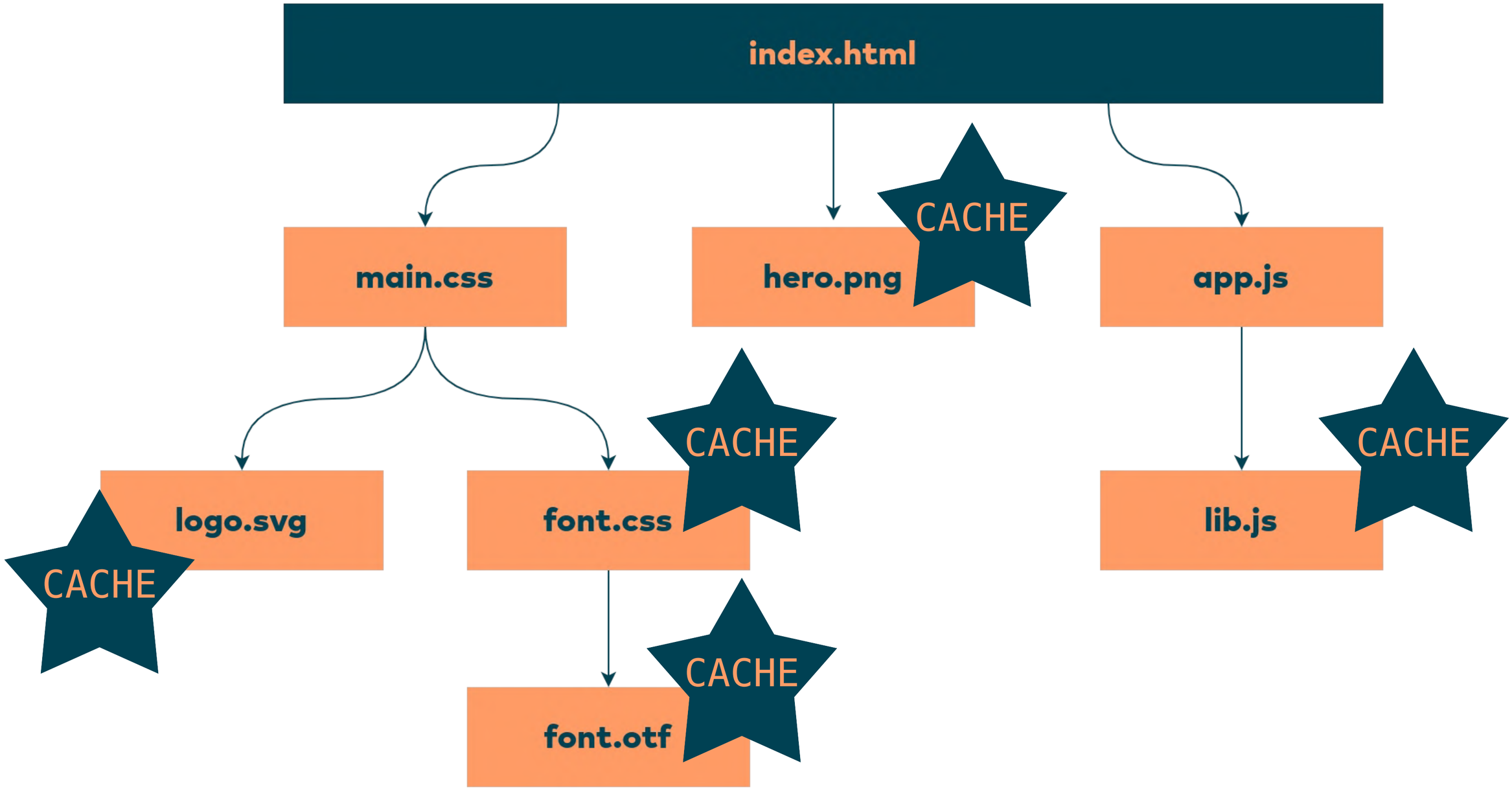
Caching

Ressourcen & Caching

HTTP/2+ ist cool, aber:

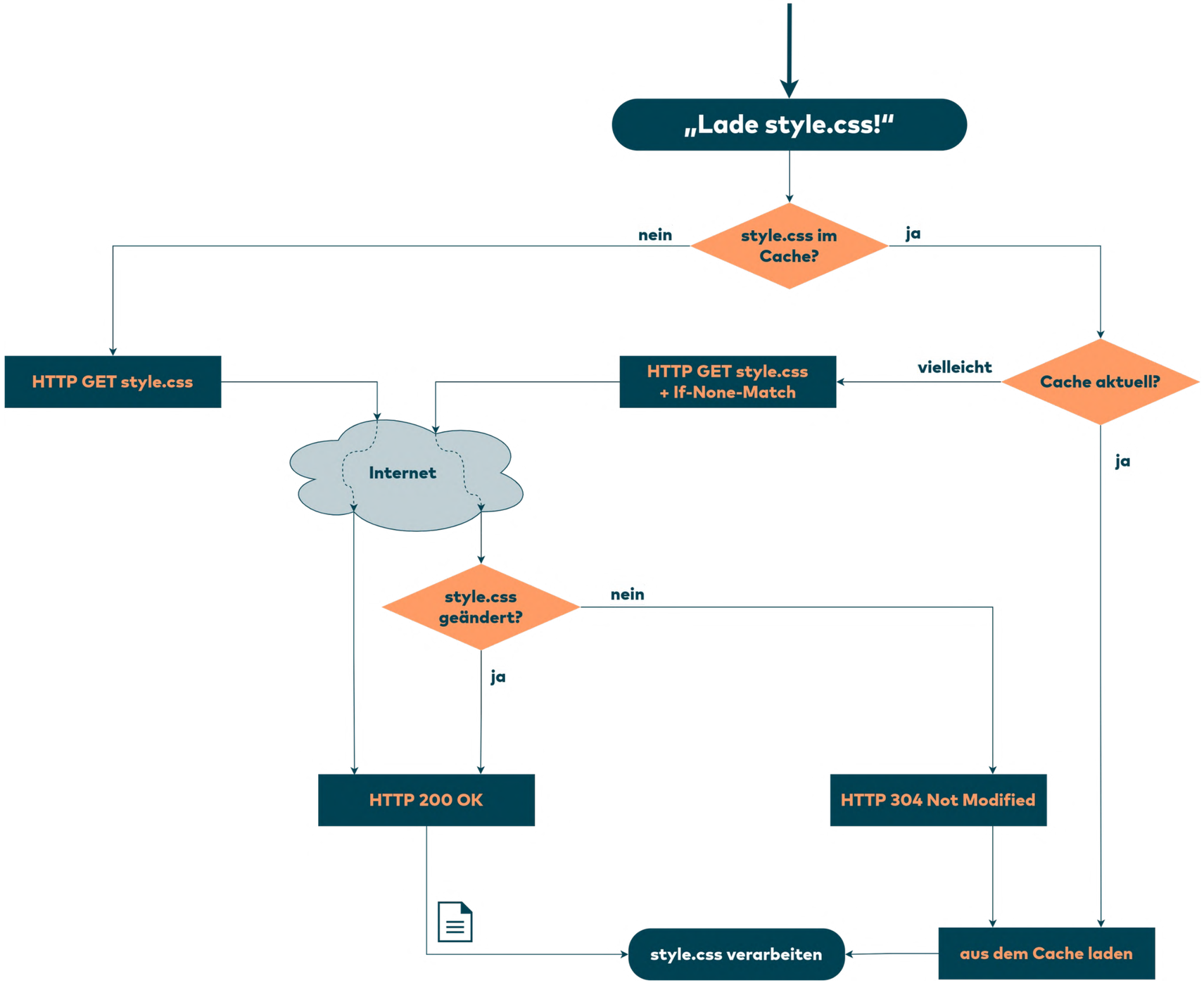
Je weniger Daten über die Leitung gehen müssen, desto besser!

→ Performance durch Caching

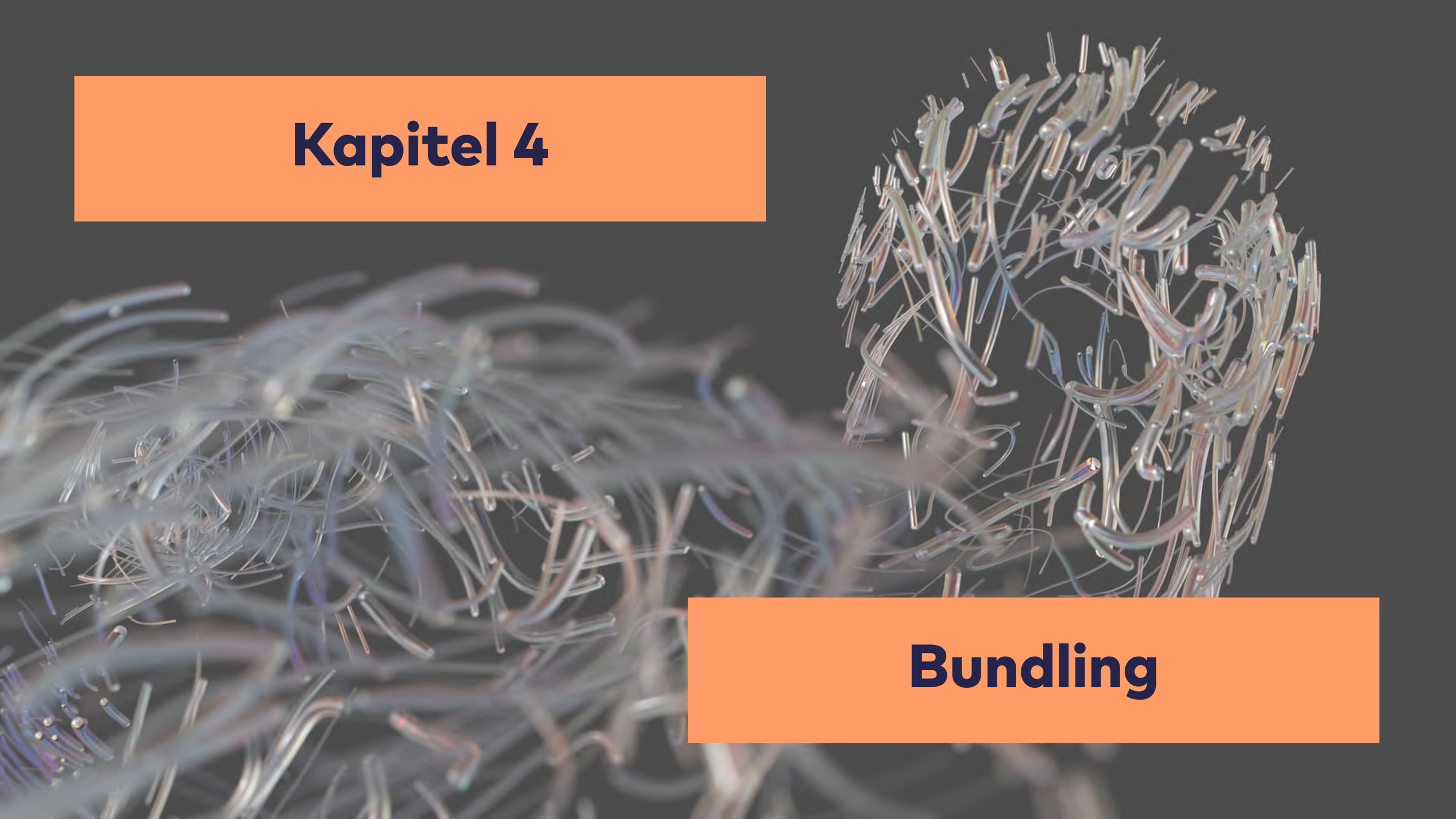

index.html
main.css
hero.png
app.js
CACHE
logo.svg
font.css
lib.js
CACHE
CACHE
CACHE
font.otf
CACHE

Wie funktioniert Caching?

- es gibt mehrere Arten von Caches:
  - Browser
  - CDN/Edge
  - Proxies
- Grundidee: zu jeder Anfrage wird die Antwort + Validität gespeichert
- Beispiele:
  - „für 1 Tag cachen ohne Rückfrage“
  - „cachen solange sich der Hash nicht geändert hat“
  - „unveränderlich → dauerhaft cachen“


„Lade style.css!“
style.css im Cache?
nein
ja
HTTP GET style.css
HTTP GET style.css + If-None-Match
vielleicht
Cache aktuell?
ja
Internet
style.css geändert?
nein
ja
HTTP 200 OK
HTTP 304 Not Modified
style.css verarbeiten
aus dem Cache laden

Kapitel 4



Bundling

Bundler zur Hilfe!

- Hauptaufgaben:
  - wir können wie vom Backend gewohnt entwickeln
  - das Resultat ist für Browser leicht verdaulich
- deswegen auch bekannt als „Frontend-Toolchain“ oder „Asset-Toolchain“

HTML

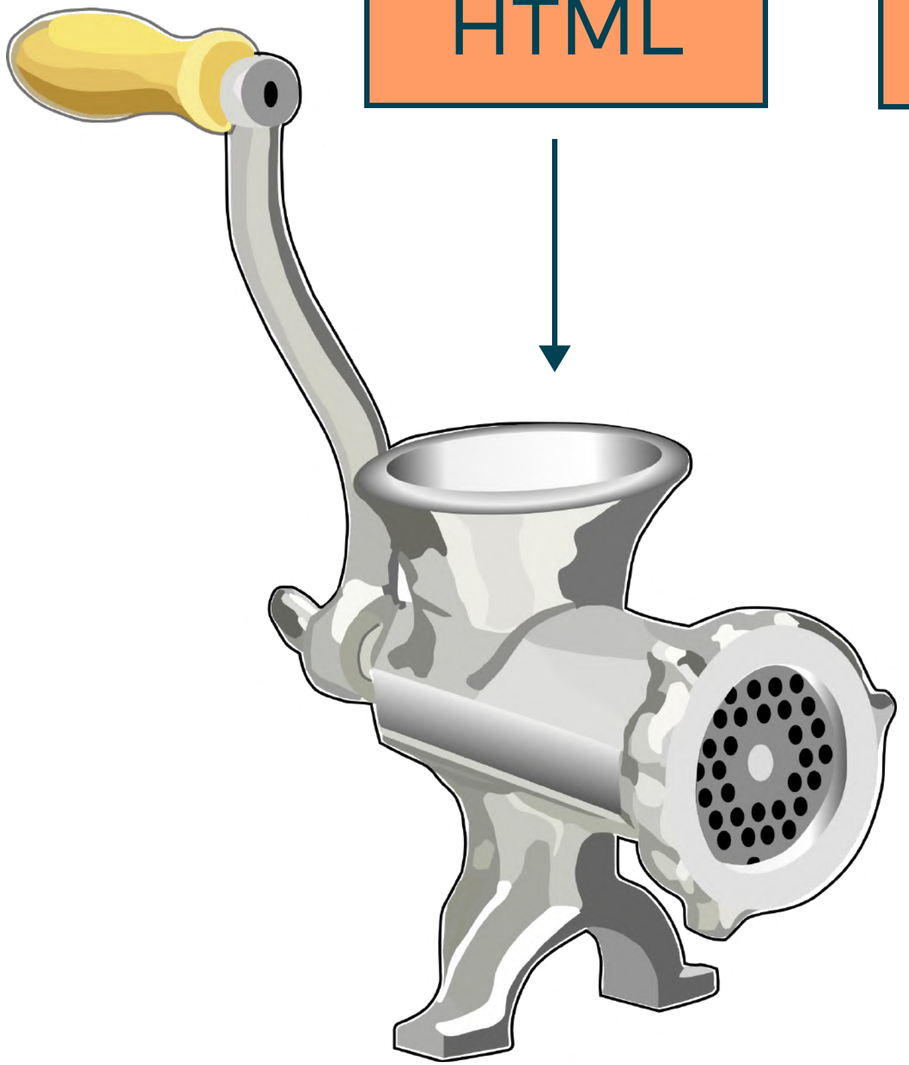
CSS

JS

TS

JSX

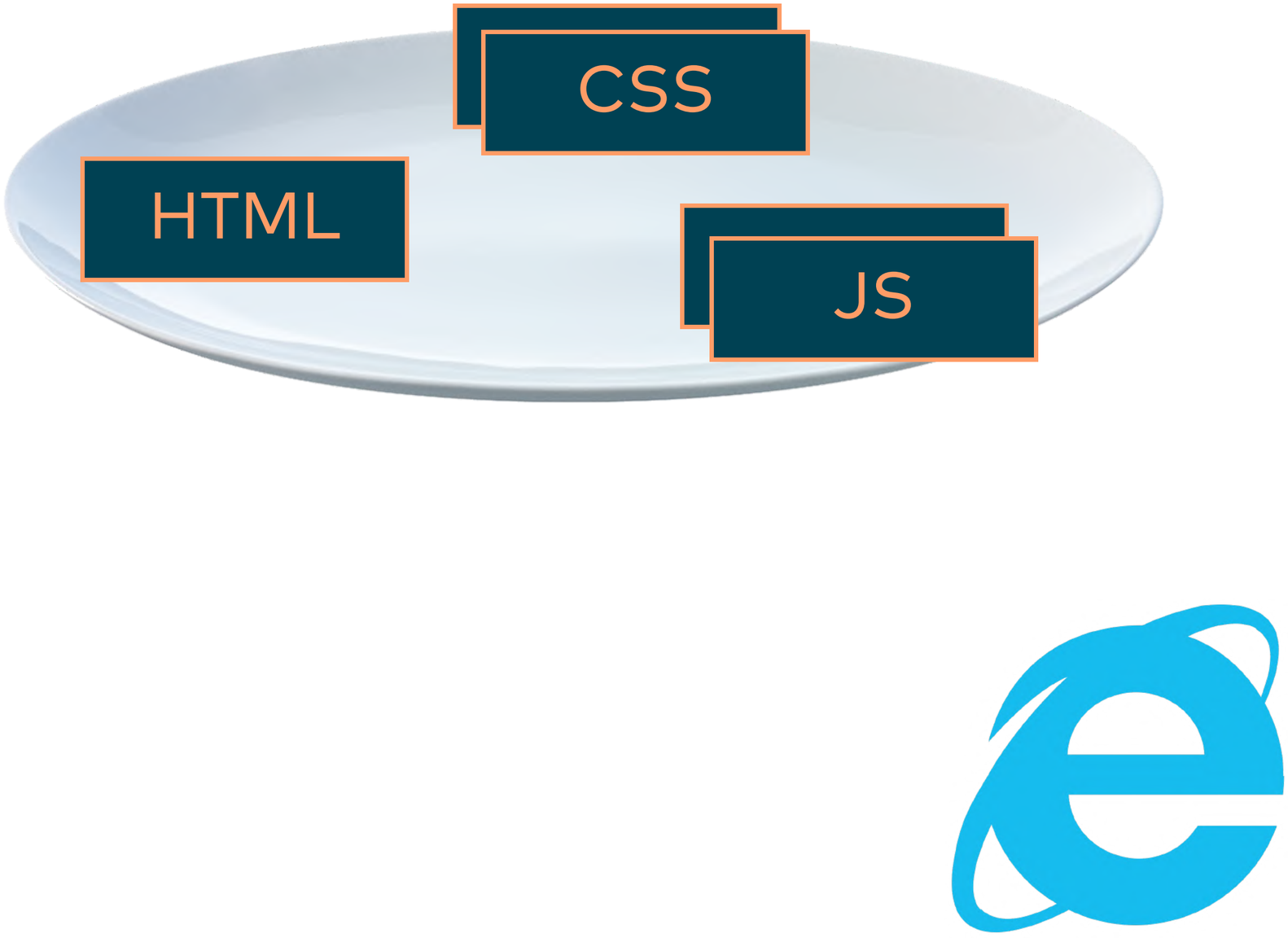
SCSS



HTML

CSS

JS


CSS
HTML
JS

Genereller Ablauf (nur JS)

1. Hauptmodul laden
2. Abhängigkeitsbaum konstruieren
3. zusätzliche Ressourcen laden und konvertieren
4. statische Analyse und Optimierungen
5. unnötige Abhängigkeiten entfernen („Treeshaking")
6. Code in (ältere Version von) JavaScript übersetzen
7. Code minifizieren

```
import React from "react";

const tree = (
  <ul>
    {
      items.map(item =>
        <li>{item}</li>
      )
    }
  </ul>
);
```

JSX

```
import React from "react";

const tree = (
  <ul>
    {
      items.map(item =>
        <li>{item}</li>
      )
    }
  </ul>
);
```

JSX

```jsx
import React from "react";

const tree = (
  <ul>
    {
      items.map(item =>
        <li>{item}</li>
      )
    }
  </ul>
);
```

```jsx
const tree = React.createElement(
  "ul",
  null,
  {
    children: items.map(item =>
      React.createElement(
        "li", null, item
      )
    )
  }
);
```

JSX

```
// lib.js
import {a} from "ext-lib";

export const b = 3;

export a;

// main.js
import {b} from "./lib.js";

console.log(b);
```

Dead code

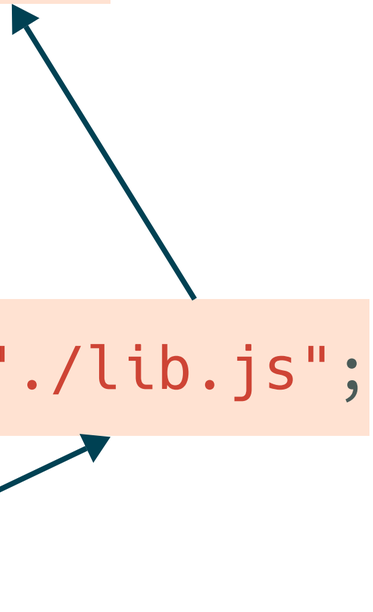
```
// lib.js
import {a} from "ext-lib";

export const b = 3;

export a;

// main.js
import {b} from "./lib.js";

console.log(b);
```



Dead code

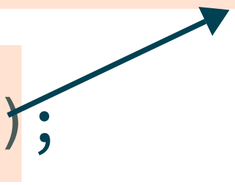
```
// lib.js
import {a} from "ext-lib";

export const b = 3;

export a;

// main.js
import {b} from "./lib.js";

console.log(b);
```



Dead code

```
// lib.js
import {a} from "ext-lib";

export const b = 3;

export a;
```

```
// main.js
import {b} from "./lib.js";

console.log(b);
```

```
// bundle.js
const b = 3;

console.log(b);
```

Dead code

```
class Person {
  constructor(name) {
    this.name = name;
  }

  hello() {
    return
      `Hello ${this.name}`;
  }
}
```

ES6+

```
class Person {
  constructor(name) {
    this.name = name;
  }

  hello() {
    return
      `Hello ${this.name}`;
  }
}
```

ES6+

```
class Person {
  constructor(name) {
    this.name = name;
  }

  hello() {
    return
      `Hello ${this.name}`;
  }
}
```

```
var Person = (function () {
  function Person(name) {
    this.name = name;
  }
  Person.prototype.hello =
    function () {
      return "Hello ".
        concat(this.name);
    };
    return Person;
}());
```

ES6+

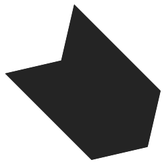
```
import "./styles.css";
```

CSS-in-JS

```
import "./styles.css";
```

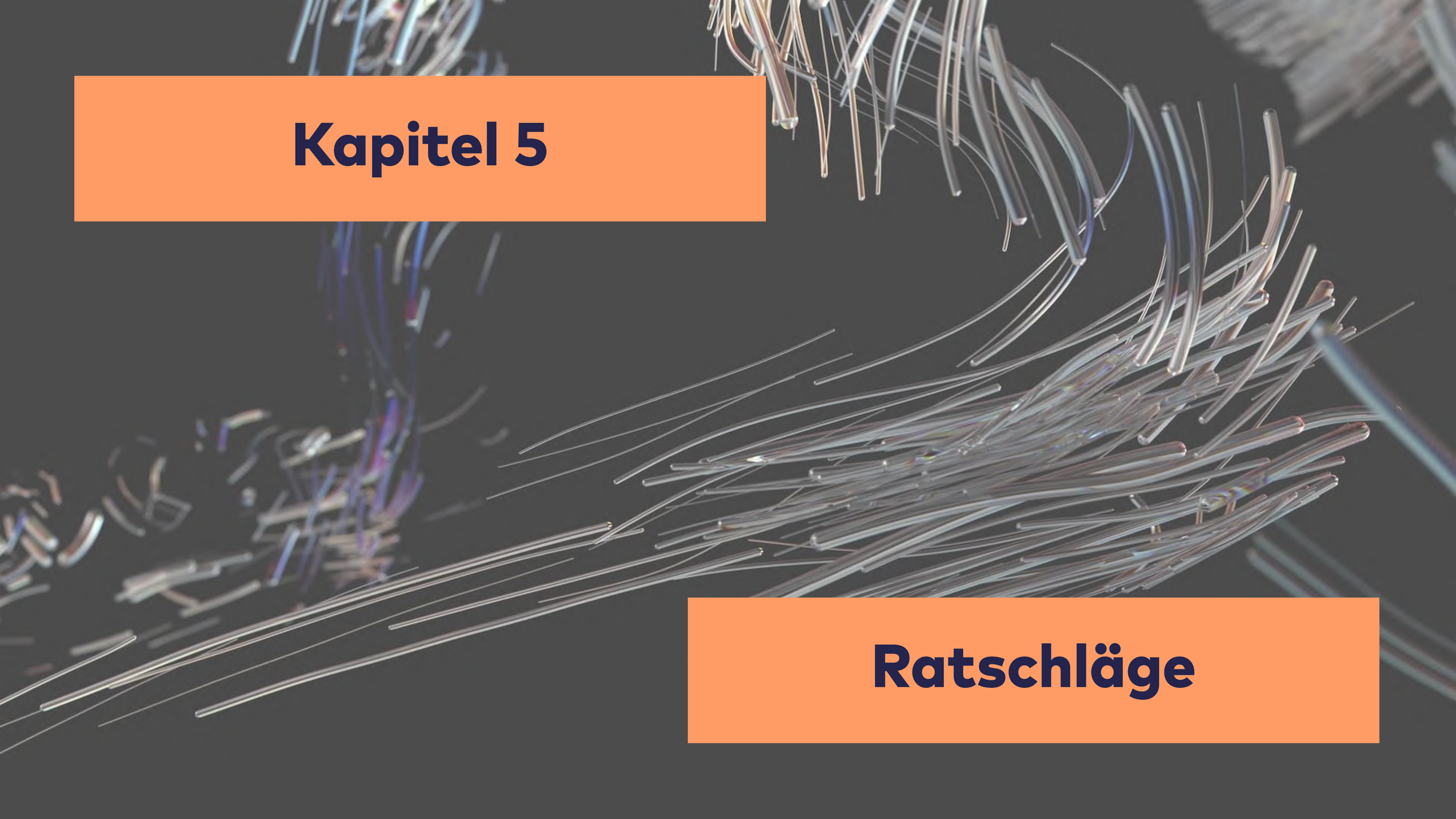
CSS-in-JS

```
import "./styles.css";
```



```
(()=>{"use strict";var t={918:(t,n,r)=>{var o=r(81),e=r.n(o),c=r(645);r.n(c)()(e()).push([t.id,"p {\n
color: red;\n}\n",""])},645:t=>{t.exports=function(t){var n=[];return n.toString=function(){return
this.map((function(n){var r="",o=void 0!==n[5];return n[4]&&(r+="@supports (".concat(n[4],") {")),n[2]&&
(r+="@media ".concat(n[2]," {")),o&&(r+="@layer".concat(n[5].length>0?" ".concat(n[5]):""," 
{")),r+=t(n),o&&(r+="}"),n[2]&&(r+="}"),n[4]&&(r+="}"),r})).join("")},n.i=function(t,r,o,e,c)
{"string"==typeof t&&(t=[[null,t,void 0]]);var a={};if(o)for(var i=0;i<this.length;i++){var u=this[i]
[0];null!=u&&(a[u]=!0)}for(var s=0;s<t.length;s++){var p=[].concat(t[s]);o&&a[p[0]]||(void 0!==c&&(void
0===p[5]||(p[1]="@layer".concat(p[5].length>0?" ".concat(p[5]):""," {").concat(p[1],"}")),p[5]=c),r&&
(p[2]?(p[1]="@media ".concat(p[2]," {").concat(p[1],"}"),p[2]=r):p[2]=r),e&&(p[4]?(p[1]="@supports
(".concat(p[4],") {").concat(p[1],"}"),p[4]=e):p[4]="".concat(e)),n.push(p))}},n}},81:t=>
{t.exports=function(t){return t[1]}}},n={};function r(o){var e=n[o];if(void 0!==e)return e.exports;var
c=n[o]={id:o,exports:{}};return t[o](c,c.exports,r),c.exports}r.n=t=>{var n=t&&t.__esModule?
()=>t.default:()=>t;return r.d(n,{a:n}),n},r.d=(t,n)=>{for(var o in
n)r.o(n,o)&&!r.o(t,o)&&Object.defineProperty(t,o,{enumerable:!0,get:n[o]})},r.o=
(t,n)=>Object.prototype.hasOwnProperty.call(t,n),r(918)})();
```

CSS-in-JS



Kapitel 5

Ratschläge

Frontend ist anders …

- „unscheinbare“ Architekturentscheidungen können große Auswirkungen haben
- gute Kenntnis von HTML, HTTP und Browsern sind essenziell
- Frontend wird leider oft nicht gut verstanden oder gar ignoriert


Schätzfrage:
Wie viele Abhängigkeiten
hat eine frische React-App?


Schätzfrage:
Wie viele Abhängigkeiten
hat eine frische React-App?
2782
SUMICH
LOTUS ROOT
VACUUM PACKED
Whole Mushrooms
CORNED BEEF
HAITAI
HAITAL
CHEESE CRACKER
FairPrice
Bleach
12 x 1 Litre
Rice Blend
6 Bags x 2.5kg
#56
Promotion Pack
Slimguard
QUAKER
桂格即食燕麦片
White Rice
4 Bags x 5kg
#1
FP Gold
4Bags x 5kg
GlucosCare® Tea
FairPrice
SARDINES IN TOMATO SAUCE
B.B:06 JUN 2019:4315717

Frontend-Frameworks haben erhebliche Komplexität.

Seid nicht faul und baut lieber Dinge von Hand!

Konfiguration der JS-Tools ist schwierig.

Macht kein Cargo Culting! Versteht, was ihr tut!

HTTP ist das Protokoll des Webs.

Arbeitet mit ihm, nicht gegen es!

HTML ist die Sprache des Webs.

Erfindet das Rad nicht neu! Der Browser ist euer Freund!

Frontend ist eine eigene Disziplin mit eigenen Regeln.

Betrachtet Frontend vom ersten Tag an in eurer Architektur!

Was zählt ist die Usability.

Messt Performance! Denkt immer an die User!

Q&A

INNOQ
www.innoq.com

Lars Hupel
lars.hupel@innoq.com
@larsr_h

innoQ Deutschland GmbH

Krischerstr. 100
40789 Monheim
+49 2173 333660

Ohlauer Str. 43
10999 Berlin

Ludwigstr. 180E
63067 Offenbach

Kreuzstr. 16
80331 München

Hermannstr. 13
20095 Hamburg

Erftstr. 15-17
50672 Köln

Königstorgraben 11
90402 Nürnberg