

DO YOU REALLY NEED AN SPA?

WIE ASTRO SCHNELLE, INTERAKTIVE WEBSITES MIT WENIGER
JAVASCRIPT-BALLAST ERMÖGLICHT

Vasco Schwarze · JUG Saxony Day 2026 · 25.9.2026

Timeseries of JavaScript Bytes

Source: <http://archive.org>

Zoom 1m 3m 6m YTD 1y 3y All

Nov 2011 → Sep 2026





JavaScript Bloat in 2024

Translations: [Korean](#) [Russian](#)

Second-guessing the modern web

The emerging norm for web development is to build a React single-page application, with server-rendering. The two key elements of this architecture are something like:

- The main UI is built & updated in JavaScript using React or something similar.
- The backend is an API that that application makes requests against.

Stranger Than Usual

This idea has really swept the internet. It started with a few major popular websites and has crept into corners like marketing and blogs.

I'm increasingly skeptical of it.

Der Gegenangriff gegen Single Page Applications

Ein altes Thema in diesem Blog, häufig von Rants begleitet, ist der übermäßige Einsatz von Javascript auf Websites. Insbesondere wenn Grundfunktionen der Website nicht mehr gehen, wenn man kein Javascript aktiviert hat. So habe ich mich zum Beispiel 2010 auf meinem alten Blog darüber aufgeregt, dass bei einer

Tom MacWright

Writing ←

Reading

Photos

Projects

Drawings

Micro

About

[Archiv](#) [Kategorien](#) [Tags](#) [Kontakt](#) [Feed](#)

Just fucking use HTML

Hey, dipshit! You know what loads faster than your bloated, overengineered mess? Plain, unadulterated HTML. And you know what doesn't break every motherfucking Tuesday? HTML that just fucking works. Why the fuck are you overcomplicating things, you masochistic fuck? You're out here acting like you're building the next goddamn moon landing when all you need is a button and some text.

justfuckingusehtml.com

DIE ZENTRALE FRAGE

"Brauchen wir dafür wirklich eine SPA?"

Oder geht es auch leichtgewichtiger?

WORUM ES HIER NICHT GEHT

- reines Astro-Tutorial
- Framework-Vergleiche (außer ein paar spezifische Metriken)
- KI

WER BIN ICH?



Vasco Schwarze
Software-Entwickler
sciota GmbH

sciota

EIN KURZER BLICK IN DIE VERGANGENHEIT...

WARUM WURDEN SPAS ZUM STANDARD?

- flüssigere UX
- Nutzung von Hardware-Fortschritten
- geringere Serverlast
- klares mentales Modell
- riesiges Ökosystem
- Team-Gewohnheit

DIE TECHNISCHEN KOSTEN

- JS benötigt
- Größere Bundles
- Verzögertes Laden der Seite
- schwieriges SEO
- Accessibility-Einschränkungen



LÄSST SICH DAS MESSEN?

LÄSST SICH DAS MESSEN?

NA KLAR!

LÄSST SICH DAS MESSEN?

CORE WEB VITALS

- LCP < 2,5sek
- INP < 200ms
- CLS < 0,1

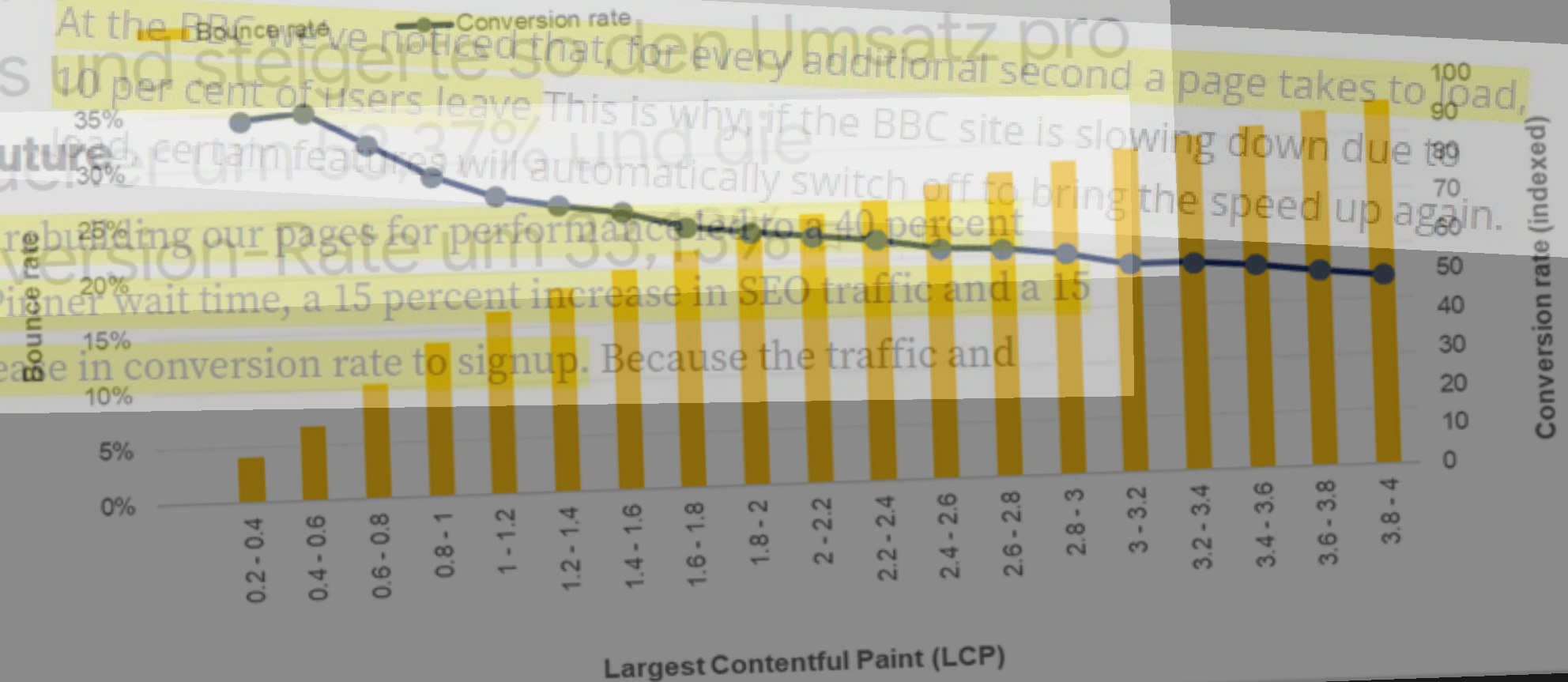
WARUM INTERESSIERT UNS DAS ÜBERHAUPT?

Rakuten 24 investierte in Core Web

Vitals und steigerte so den Umsatz pro

Results & future

The result of rebuilding our pages for performance led to a 40 percent decrease in Pinner wait time, a 15 percent increase in SEO traffic and a 15 percent increase in conversion rate to signup. Because the traffic and



WARUM INTERESSIERT UNS DAS ÜBERHAUPT?

- bessere UX -> zufriedene Nutzer :)
- Conversion/Retention
- Nachhaltigkeit

ZEIT FÜR EINEN VERGLEICH...

DIE BASELINE-APP: REZEPTBUCH

- Startseite mit ausgewählten Rezepten
- Übersicht mit Textsuche & Kategorie-Filter
- Detailseite mit Portionsrechner & Kochmodus-Checkliste
- Favoriten-Toggle mit *localStorage*-Persistenz

React 19 + React Router 7 · Vite-Build · ein JS-Bundle für die komplette App

Kochen ohne Kompromisse

Eine Vielzahl durchdachter Rezepte, live berechnete Portionen und ein Kochmodus, der dich Schritt für Schritt begleitet.

[Alle Rezepte entdecken](#)

Ausgewählte Rezepte



DESSERT



Weiche Schokoladentarte

Zart schmelzende Tarte mit dunkler Schokolade auf knusprigem Keksboden.

35 Min.



VEGETARISCH



Nudel-Risotto mit Erbsen

Cremiges One-Pot-Gericht aus Hörnchennudeln, Erbsen und würzigem Frischkäse.

25 Min.



VEGETARISCH



Blumenkohl-Gratin mit Kurkuma

Goldener Blumenkohl-Auflauf mit Kurkuma, Joghurt und geriebenem Käse.

40 Min.



FLEISCH



Kurz mariniertes Butter Chicken

Cremiges, kurz mariniertes Butter-Hähnchen in würziger Tomatensauce.

75 Min.



TECHNISCHER AUFBAU DER BASELINE

```
1 default function App() {  
2   return (  
3     <RecipesProvider>  
4       <Routes>  
5         <Route element={<Layout />}>  
6           <Route index element={<Home />} />  
7           <Route path="rezepte" element={<Recipes />} />  
8           <Route path="rezepte/:slug" element={<RecipeDetail />} />  
9           <Route path="ueber-uns" element={<About />} />  
10          <Route path="*" element={<NotFound />} />  
11        </Route>  
12      </Routes>  
13    </RecipesProvider>  
14  );  
15 }
```

App.tsx

TECHNISCHER AUFBAU DER BASELINE

```
1 export function RecipesProvider({ children }: { children: ReactNode }) {
2   const [recipes, setRecipes] = useState<Recipe[]>([]);
3   const [status, setStatus] = useState<Status>("loading");
4
5   useEffect(() => {
6     let cancelled = false;
7
8     fetch("/recipes.json")
9       .then((res) => {
10         if (!res.ok) {
11           throw new Error(`Fehler beim Laden der Rezepte`);
12         }
13         return res.json() as Promise<Recipe[]>;
14       })
15       .then((data) => {
16         if (cancelled) return;
17         setRecipes(data);
18         setStatus("ready");
19       })
20       .catch(() => {
21         if (cancelled) return;
22         setStatus("error");
23       });
24 }
```

context/RecipesContext.tsx

TECHNISCHER AUFBAU DER BASELINE

```
1 export default function RecipeDetail() {
2   const { slug } = useParams<{ slug: string }>();
3   const { recipes, status } = useRecipes();
4   const { isFavorite, toggleFavorite } = useFavorites();
5
6   const recipe = recipes.find((r) => r.slug === slug);
7
8   const [servings, setServings] = useState<number | null>(null);
9   const [checkedSteps, setCheckedSteps] = useState<Set<number>>(() => new Set());
10
11   const activeServings = servings ?? recipe?.servings ?? 1;
12
13   const scaledIngredients = useMemo(() => {
14     if (!recipe) return [];
15     return recipe.ingredients.map((ing) => ({
16       ...ing,
17       scaledAmount: scaleAmount(ing.amount, recipe.servings, activeServings),
18     }));
19   }, [recipe, activeServings]);
20
21   function toggleStep(index: number) {
22     setCheckedSteps((prev) => {
23       const next = new Set(prev);
```

pages/RecipeDetail.tsx

TECHNISCHER AUFBAU DER BASELINE

```
1  const STORAGE_KEY = "recipe-app:favorites";
2
3  function readFavorites(): Set<string> {
4    try {
5      const raw = localStorage.getItem(STORAGE_KEY);
6      return raw ? new Set(JSON.parse(raw)) : new Set();
7    } catch {
8      return new Set();
9    }
10 }
11
12 let favorites = readFavorites();
13 const listeners = new Set<() => void>();
14
15 function writeFavorites(next: Set<string>) {
16   favorites = next;
17   localStorage.setItem(STORAGE_KEY, JSON.stringify([...next]));
18   listeners.forEach((listener) => listener());
19 }
20
21 function subscribe(listener: () => void) {
22   listeners.add(listener);
23   return () => listeners.delete(listener);
```

hooks/useFavorites.ts

DER OUTPUT

dist/index.html	0.49	kB
dist/assets/index-DdRMf1ka.css	6.68	kB
dist/assets/index-XPSA8tZd.js	270.08	kB



DIE BASELINE

Metrik	recipe-react (SPA)
Build-Output gesamt (dist/)	284,7 KB
LCP (Rezepteübersicht)	146 ms
übertragene Datenmenge (Rezepteübersicht)	89,8 KB
LCP ("Über uns")	68 ms
übertragene Datenmenge ("Über uns")	89,8 KB

DAS MUSS DOCH BESSER GEHEN...

JA, ZUM BEISPIEL MIT



ASTROS PHILOSOPHIE

„Ship Zero JavaScript by Default“

ISLANDS ARCHITECTURE

Header (interaktive Insel)

Sidebar (statisch)

Hauptinhalt (statischer Text, Bilder
etc.)

Karussell (interaktive Insel)

Footer (statisch)

ÄHNLICHE CONTENT-FIRST ANSÄTZE

- **Qwik** - Resumability
- **htmx** - HTML-Attribute statt Komponenten
- **Next.js** - Partial Prerendering / RSC
- **Astro** - unser Beispiel heute

WEITERE COOLE ASTRO-FEATURES

- Scoped CSS
- Content Collections
- Optimierte Bilder
- Einbindung von JS-Framework-Komponenten
- i18n
- View Transitions

ANATOMIE EINER .ASTRO-DATEI

```
1 <!doctype html>
2 <html lang="de">
3   <head>
4     <meta charset="UTF-8" />
5     <title>Rezeptbuch</title>
6   </head>
7   <body>
8     <main>
9       <h1>Rezepte</h1>
10      <p>Hier steht der Inhalt</p>
11    </main>
12  </body>
13 </html>
```

ANATOMIE EINER .ASTRO-DATEI

```
1 ---
2 ---
3 <!doctype html>
4 <html lang="de">
5   <head>
6     <meta charset="UTF-8" />
7     <title>Rezeptbuch</title>
8   </head>
9   <body>
10    <main>
11      <h1>Rezepte</h1>
12      <p>Hier steht der Inhalt</p>
13    </main>
14  </body>
15 </html>
```

ANATOMIE EINER .ASTRO-DATEI

```
1  ---
2  const num = 3 + 4;
3  ---
4  <!doctype html>
5  <html lang="de">
6    <head>
7      <meta charset="UTF-8" />
8      <title>Rezeptbuch</title>
9    </head>
10   <body>
11     <main>
12       <h1>Rezepte</h1>
13       <p>Hier steht der Inhalt: {num}</p>
14     </main>
15   </body>
16 </html>
```

```
1 <!doctype html>
2 <html lang="de">
3   <head>
4     <meta charset="UTF-8" />
5     <title>Rezeptbuch</title>
6   </head>
7   <body>
8     <main>
9       <h1>Rezepte</h1>
10      <p>Hier steht der Inhalt: 7</p>
11    </main>
12  </body>
13 </html>
```

ANATOMIE EINER .ASTRO-DATEI

```
1 ---
2 const names = ["John", "Paul"];
3 ---
4 <!doctype html>
5 <html lang="de">
6   <head>
7     <meta charset="UTF-8" />
8     <title>Rezeptbuch</title>
9   </head>
10  <body>
11    <main>
12      <h1>Rezepte</h1>
13      <ul>
14        {names.map(name => <li>{name}</li>)}
15      </ul>
16    </main>
17  </body>
18 </html>
```

```
1 <!doctype html>
2 <html lang="de">
3   <head>
4     <meta charset="UTF-8" />
5     <title>Rezeptbuch</title>
6   </head>
7   <body>
8     <main>
9       <h1>Rezepte</h1>
10      <ul>
11        <li>John</li>
12        <li>Paul</li>
13      </ul>
14    </main>
15  </body>
16 </html>
```

KOMPONENTEN WIEDERVERWENDEN

ExampleText.astro

```
1 <p class="text">  
2   etwas Text  
3 </div>
```

RecipesPage.astro

```
1 ---  
2 import ExampleText from "./ExampleText.astro";  
3 ---  
4  
5 <ExampleText />
```

KIND-INHALT EINBETTEN MIT SLOTS

ExampleText.astro

```
1 <p class="text">
2   <slot />
3 </div>
```

RecipesPage.astro

```
1 ---
2 import ExampleText from "./ExampleText.astro";
3 ---
4
5 <ExampleText >
6   etwas Text
7 </ExampleText/>
```

PROPS

ExampleText.astro

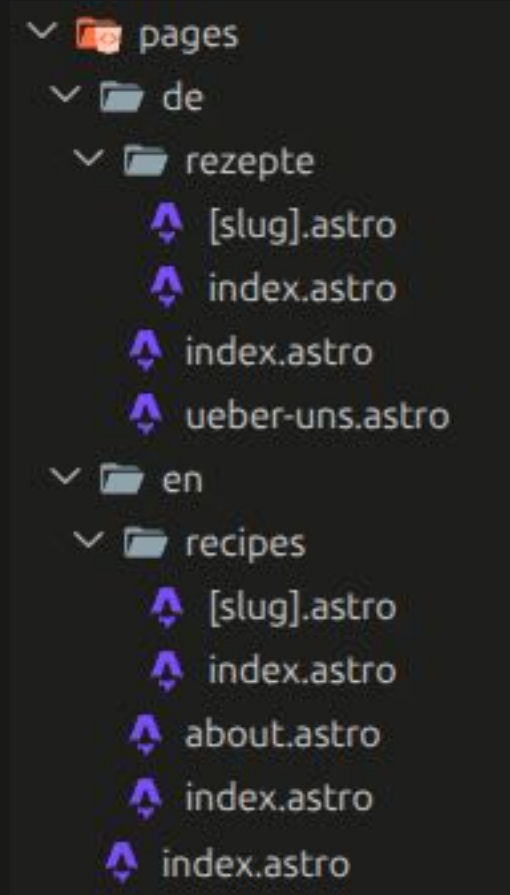
```
1 ---
2 type Props = {
3   text: string;
4 }
5
6 const { text } = Astro.props;
7 ---
8
9 <p class="text">
10   {text}
11 </div>
```

RecipesPage.astro

```
1 ---
2 import ExampleText from "../ExampleText.astro";
3 ---
4
5 <ExampleText text={"etwas Text"} />
```

REZEPTBUCH APP - VERSION 2

FILE-BASED ROUTING



Über uns

Rezeptbuch ist ein Demo-Projekt, das im Rahmen eines Konferenzvortrags entstanden ist. Es vergleicht zwei technische Ansätze für dieselbe kleine Rezepte-App: eine klassische React Single Page Application und diese Astro-App mit Islands-Architektur.

Beide Varianten bieten identische Inhalte und Funktionen: Rezeptsuche, Portionsrechner, Favoriten und einen Kochmodus. Der Unterschied liegt ausschließlich darin, wie viel JavaScript zum Anzeigen und Bedienen der Seite notwendig ist.

PAGES

```
1 ---
2 import BaseLayout from '../..../layouts/BaseLayout.astro';
3 import AboutView from '../..../components/views/AboutView.astro';
4 import { useTranslations } from '../..../i18n/utils';
5
6 const lang = 'de' as const;
7 const t = useTranslations(lang);
8 ---
9
10 <BaseLayout lang={lang} title="Über uns" description={t('about.p1')} >
11   <AboutView lang={lang} />
12 </BaseLayout>
```

pages/de/ueber-uns.astro

PAGES

```
1 ---
2 import { getCollection } from 'astro:content';
3 import BaseLayout from '../layouts/BaseLayout.astro';
4 import HomeView from '../components/views/HomeView.astro';
5 import { useTranslations } from '../i18n/utils';
6
7 const lang = 'de' as const;
8 const t = useTranslations(lang);
9
10 const allRecipes = await getCollection('recipes');
11 const featured = allRecipes.filter((r) => r.data.featured).sort((a, b) => a.id.localeCompare(b.id));
12 ---
13
14 <BaseLayout lang={lang} title="Rezeptbuch" description={t('home.subtitle')}>
15   <HomeView lang={lang} featured={featured} />
16 </BaseLayout>
```

pages/de/index.astro

LAYOUT

```
1  ---
2  import Header from "../components/Header.astro";
3  import type { Lang } from "../i18n/ui";
4
5  type Props = {
6    lang: Lang;
7    title: string;
8    description: string;
9  };
10
11  const { lang, title, description } = Astro.props;
12  ---
13
14  <!doctype html>
15  <html lang={lang}>
16    <head>
17      <meta charset="UTF-8" />
18      <meta name="viewport" content="width=device-width, initial-scale=1.0" />
19      <meta name="generator" content={Astro.generator} />
20      <title>{title}</title>
21      <meta name="description" content={description} />
22    </head>
23    <body>
```

layouts/BaseLayout.astro

ÜBER UNS - INHALT

```
1 ---
2 import { useTranslations } from '../..//i18n/utils';
3 import type { Lang } from '../..//i18n/ui';
4
5 type Props = {
6   lang: Lang;
7 }
8
9 const { lang } = Astro.props;
10 const t = useTranslations(lang);
11 ---
12
13 <div class="page">
14   <h1>{t('about.title')}</h1>
15   <p>{t('about.p1')}</p>
16   <p>{t('about.p2')}</p>
17 </div>
18
19 <style>
20   .page {
21     max-width: 640px;
22   }
23
```

components/views/AboutView.astro

SCOPED CSS

```
1  ---
2  import { useTranslations } from '../i18n/uti
3  import type { Lang } from '../i18n/ui';
4
5  type Props = {
6    lang: Lang;
7  }
8
9  const { lang } = Astro.props;
10 const t = useTranslations(lang);
11 ---
12
13 <div class="page">
14   <h1>{t('about.title')}</h1>
15   <p>{t('about.p1')}</p>
16   <p>{t('about.p2')}</p>
17 </div>
18
19 <style>
20   .page {
21     max-width: 640px;
22   }
23
```

```
<div class="page" data-astro-cid-gtkanwr>
  <h1 data-astro-cid-gtkanwr>Über uns</h1>
  <p data-astro-cid-gtkanwr>
    Rezeptbuch ist ein ...
  </p>
  <p data-astro-cid-gtkanwr>
    Beide Varianten bieten ...
  </p>
</div>

<style>
.page[data-astro-cid-gtkanwr] {
  max-width: 640px;
}

.page[data-astro-cid-gtkanwr] h1[data-astro-c
  margin-bottom: 1rem;
}
</style>
```

CONTENT COLLECTIONS

```
1 import { defineCollection } from "astro:content";
2 import { glob } from "astro/loaders";
3 import { z } from "astro/zod";
4
5 const recipes = defineCollection({
6   loader: glob({ pattern: "**/*.yaml", base: "./src/content/recipes" }),
7   schema: z.object({
8     title: z.object({
9       de: z.string(),
10      en: z.string(),
11    }),
12    description: z.object({
13      de: z.string(),
14      en: z.string(),
15    }),
16    category: z.enum(["Vegetarisch", "Fleisch", "Dessert", "Vegan"]),
17    cookTime: z.number().positive(),
18    difficulty: z.number().int().min(1).max(5),
19    servings: z.number().int().positive(),
20    featured: z.boolean().default(false),
21    image: z.string(),
22    ingredients: z.array(
23      z.object({
```

content.config.ts

CONTENT COLLECTIONS

```
content/recipes
├── blumenkohl-gratin-mit-kurkuma.yaml
├── butter-chicken.yaml
├── gruenes-brokkoli-curry.yaml
├── hackbaellchen-curry.yaml
├── nudel-risotto-mit-erbsen.yaml
├── schweinsoehrchen-mit-zimtzucker.yaml
├── tomaten-gefüllt-mit-oliven.yaml
└── weiche-schokoladentarte.yaml
```

CONTENT COLLECTIONS

```
1 title:
2   de: "Blumenkohl-Gratin mit Kurkuma"
3   en: "Cauliflower Gratin with Turmeric"
4 description:
5   de: "Goldener Blumenkohl-Auflauf mit Kurkuma, Joghurt und geriebenem Käse."
6   en: "Golden roasted cauliflower bake with turmeric, yogurt, and grated cheese."
7 category: "Vegetarisch"
8 cookTime: 40
9 difficulty: 1
10 servings: 4
11 featured: true
12 image: "/images/recipes/blumenkohl-gratin-mit-kurkuma.jpg"
13 ingredients:
14   - name: "Blumenkohl"
15     amount: 1
16     unit: "Stück"
17   - name: "Kurkumapulver"
18     amount: 2
19     unit: "EL"
20   - name: "Griechischer Joghurt"
21     amount: 250
22     unit: "g"
23   - name: "Geriebener Käse"
```

CONTENT COLLECTIONS

BEI SCHEMA-VERSTOSS: BUILD-FEHLER

```
1 title:
2   de: "Blumenkohl-Gratin mit Kurkuma"
3 description:
4   de: "Goldener Blumenkohl-Auflauf mit Kurkuma, Joghurt und geriebenem Käse."
5 category: "Vegetarisch"
6 cookTime: 40
7 difficulty: 1
8 servings: 4
9 featured: true
10 image: "/images/recipes/blumenkohl-gratin-mit-kurkuma.jpg"
11 ingredients:
12   - name: "Blumenkohl"
13     amount: 1
14     unit: "Stück"
15   - name: "Kurkumapulver"
16     amount: 2
17     unit: "EL"
18   - name: "Griechischer Joghurt"
19     amount: 250
20     unit: "g"
21   - name: "Geriebener Käse"
22     amount: 150
23     unit: "g"
```

CONTENT COLLECTIONS

BEI SCHEMA-VERSTOSS: BUILD-FEHLER

```
[InvalidContentEntryDataError] recipes → blumenkohl-gratin-mit-kurkuma data does not match collection schema.
```

```
title.en**: **title.en: Required
```

```
description.en**: **description.en: Required
```

Hint:

See <https://docs.astro.build/en/guides/content-collections/> for more information on content schemas.

Error reference:

<https://docs.astro.build/en/reference/errors/invalid-content-entry-data-error/>

Location:

</home/vasco/projects/do-you-really-need-an-spa/examples/recipe-astro/src/content/recipes/blumenkohl-gratin-mit-kurkuma.yaml:0:0>

Stack trace:

at getEntryData (/home/vasco/projects/do-you-really-need-an-spa/node_modules/astro/dist/content/utils.js:126:9)

at async eval (/home/vasco/projects/do-you-really-need-an-spa/node_modules/astro/dist/content/loaders/glob.js:224:13)

CONTENT COLLECTIONS

```
1 ---
2 import { getCollection } from 'astro:content';
3 import BaseLayout from '../../layouts/BaseLayout.astro';
4 import HomeView from '../../components/views/HomeView.astro';
5 import { useTranslations } from '../../i18n/utils';
6
7 const lang = 'de' as const;
8 const t = useTranslations(lang);
9
10 const allRecipes = await getCollection('recipes');
11 const featured =
12   allRecipes
13     .filter((r) => r.data.featured)
14     .sort((a, b) => a.id.localeCompare(b.id));
15 ---
16
17 <BaseLayout lang={lang} title="Rezeptbuch" description={t('home.subtitle')}>
18   <HomeView lang={lang} featured={featured} />
19 </BaseLayout>
```

pages/de/index.astro

DYNAMISCHE GENERIERUNG VON SEITEN

```
1 ---
2 import { getCollection } from "astro:content";
3 import BaseLayout from "../../layouts/BaseLayout.astro";
4 import RecipeDetailView from "../../components/views/RecipeDetailView.astro";
5
6 const lang = "de" as const;
7
8 export async function getStaticPaths() {
9     const recipes = await getCollection("recipes");
10    return recipes.map((recipe) => ({
11        params: { slug: recipe.id },
12        props: { recipe },
13    }));
14 }
15
16 const { recipe } = Astro.props;
17 ---
18
19 <BaseLayout
20     lang={lang}
21     title={` ${recipe.data.title.de} - Rezeptbuch` }
22     description={recipe.data.description.de}
23 >
```

pages/de/rezepte/[slug].astro

DYNAMISCHE GENERIERUNG VON SEITEN

```
content/recipes
├── blumenkohl-gratin-mit-kurkuma.yaml
├── butter-chicken.yaml
├── gruenes-brokkoli-curry.yaml
├── hackbaellchen-curry.yaml
├── nudel-risotto-mit-erbsen.yaml
├── schweinsoehrchen-mit-zimtzucker.yaml
├── tomaten-gefüllt-mit-oliven.yaml
└── weiche-schokoladentarte.yaml
```

```
de
├── rezepte
│   ├── blumenkohl-gratin-mit-kurkuma
│   │   └── index.html
│   ├── butter-chicken
│   │   └── index.html
│   ├── gruenes-brokkoli-curry
│   │   └── index.html
│   ├── hackbaellchen-curry
│   │   └── index.html
│   ├── nudel-risotto-mit-erbsen
│   │   └── index.html
│   ├── schweinsoehrchen-mit-zimtzucker
│   │   └── index.html
│   ├── tomaten-gefüllt-mit-oliven
│   │   └── index.html
│   └── weiche-schokoladentarte
│       └── index.html
```

UND JETZT...
...ZEIT FÜR INTERAKTIVITÄT!

KOCHMODUS-CHECKLISTE

Rezeptbuch

Start

Rezepte

Über uns

EN

← Zurück zur Übersicht



FLEISCH

Kurz mariniertes Butter Chicken

Cremiges, kurz mariniertes Butter-Hähnchen in würziger Tomatensauce.

🕒 75 Min. ●●●●○ 

KOCHMODUS-CHECKLISTE

```
1 ---
2 import Stars from "../Stars.astro";
3 import FavoriteToggle from "../FavoriteToggle.astro";
4 import PortionCalculator from "../PortionCalculator.astro";
5 import { useTranslations, localizedPath } from "../../i18n/utils";
6 import type { Lang } from "../../i18n/ui";
7 import type { CollectionEntry } from "astro:content";
8
9 type Props = {
10   lang: Lang;
11   recipe: CollectionEntry<"recipes">;
12 };
13
14 const { lang, recipe } = Astro.props;
15 const t = useTranslations(lang);
16 const { data } = recipe;
17 ---
18
19 <article class="page">
20
21   <div class="hero">
22     <img src={data.image} alt={data.title[lang]} class="hero-image" />
23     <div class="hero-info">
```

components/views/RecipeDetailView.astro

PORTIONSRECHNER

```
1  ---
2
3  ...
4
5  const { data } = recipe;
6  ---
7
8  <article class="page">
9
10   ...
11
12   <div class="content">
13     <section class="ingredients" aria-labelledby="ingredients-heading">
14       <h2 id="ingredients-heading">{t("recipe.ingredients")}</h2>
15       <PortionCalculator
16         servings={data.servings}
17         ingredients={data.ingredients}
18         lang={lang}
19       />
20     </section>
21
22   ...
23 </div>
```

components/views/RecipeDetailView.astro

PORTIONSRECHNER

```
1  ---
2  import { useTranslations } from '../i18n/utils';
3  import type { Lang } from '../i18n/ui';
4
5  type Props = {
6    servings: number;
7    ingredients: { name: string; amount: number; unit: string }[];
8    lang: Lang;
9  }
10
11  const { servings, ingredients, lang } = Astro.props;
12  const t = useTranslations(lang);
13  ---
14
15  <div class="portion-calculator" data-base-servings={servings}>
16    <div class="stepper">
17      <label for="servings-input">{t('recipe.servings')}</label>
18      <div class="stepper-controls">
19        <button type="button" data-decrement>--</button>
20        <input
21          type="number"
22          id="servings-input"
23          data-servings-input
```

components/PortionCalculator.astro

PORTIONSRECHNER

```
1 function scaleAmount(  
2   amount: number,  
3   baseServings: number,  
4   targetServings: number  
5 ): string {  
6   const scaled = (amount * targetServings) / baseServings;  
7   const rounded = Math.round(scaled * 100) / 100;  
8   if (Number.isInteger(rounded)) return String(rounded);  
9   return rounded.toFixed(rounded < 10 ? 2 : 1).replace(/\.?0+$/, "");  
10 }  
11  
12 function initPortionCalculator(root: HTMLElement) {  
13   const baseServings = Number(root.dataset.baseServings ?? "1");  
14   const input = root.querySelector<HTMLInputElement>("[data-servings-input]");  
15   const decrement = root.querySelector<HTMLButtonElement>("[data-decrement]");  
16   const increment = root.querySelector<HTMLButtonElement>("[data-increment]");  
17   const amounts = [...root.querySelectorAll<HTMLElement>("[data-amount]")];  
18   if (!input) return;  
19  
20   const render = (servings: number) => {  
21     input.value = String(servings);  
22     for (const el of amounts) {  
23       const baseAmount = Number(el.dataset.amount);
```

scripts/portion-calculator.ts

FAVORITEN-TOGGLE

```
1  ---
2  type Props = {
3    slug: string;
4  };
5
6  const { slug } = Astro.props;
7  ---
8
9  <button
10    type="button"
11    class="fav-btn"
12    data-slug={slug}>♥</button>
13  >
14
15  <script>
16    import "../scripts/favorite-toggle";
17  </script>
```

components/FavoriteToggle.astro

FAVORITEN-TOGGLE

```
1 const STORAGE_KEY = 'recipe-app:favorites';
2
3 function readFavorites(): Set<string> {
4   try {
5     const raw = localStorage.getItem(STORAGE_KEY);
6     return raw ? new Set(JSON.parse(raw)) : new Set();
7   } catch {
8     return new Set();
9   }
10 }
11
12 function writeFavorites(set: Set<string>) {
13   localStorage.setItem(STORAGE_KEY, JSON.stringify([...set]));
14 }
15
16 function initFavoriteToggle(button: HTMLButtonElement) {
17   const slug = button.dataset.slug;
18   if (!slug) return;
19
20   const sync = () => {
21     const active = readFavorites().has(slug);
22     button.textContent = active ? '♥' : '♡';
23     button.classList.toggle('is-active', active);
```

scripts/favorite-toggle.ts

SUCH- UND FILTERLEISTE GANZ KURZ IM ÜBERBLICK

Mom, can we have  React ?

No. There is  React ^{At}
Home

 React At home...

document.querySelectorAll(...)

FAZIT

INTERAKTIVITÄT GEHT AUCH OHNE FRAMEWORK!

BENCHMARK-VERGLEICH

Metrik	recipe-react (SPA)	recipe-astro
Build-Output gesamt (dist/)	284,7 KB	260,6 KB
LCP (Rezepteübersicht)	146 ms	93 ms
übertragene Datenmenge (Rezepteübersicht)	89,8 KB	5,9 KB
LCP ("Über uns")	68 ms	38 ms
übertragene Datenmenge ("Über uns")	89,8 KB	3,5 KB



ASTRO KANN MEHR ALS NUR STATISCH

- SSR
- Server Islands
- Actions
- Endpoints
- ...

Kochen ohne Kompromisse

Eine Vielzahl durchdachter Rezepte, live berechnete Portionen und ein Kochmodus, der dich Schritt für Schritt begleitet.

Alle Rezepte entdecken

Ausgewählte Rezepte



VEGETARISCH



Blumenkohl-Gratin mit Kurkuma

Goldener Blumenkohl-Auflauf mit Kurkuma, Joghurt und geriebenem Käse.

40 Min. ●○○○○



FLEISCH



Kurz mariniertes Butter Chicken

Cremiges, kurz mariniertes Butter-Hähnchen in würziger Tomatensauce.

75 Min. ●●●○○



VEGETARISCH



Nudel-Risotto mit Erbsen

Cremiges One-Pot-Gericht aus Hörnchennudeln, Erbsen und würzigem Frischkäse.

25 Min. ●○○○○



DESSERT



Weiche Schokoladentarte

Zart schmelzende Tarte mit dunkler Schokolade auf knusprigem Keksboden.

35 Min. ●●●○○

SERVERADAPTER

```
1 import { defineConfig } from "astro/config";
2 import node from "@astrojs/node";
3
4 export default defineConfig({
5   output: "static",
6   i18n: {
7     defaultLocale: "de",
8     locales: ["de", "en"],
9     routing: {
10       prefixDefaultLocale: true,
11       redirectToDefaultLocale: true,
12     },
13   },
14 });
```

astro.config.ts

SERVERADAPTER

```
1 import { defineConfig } from "astro/config";
2 import node from "@astrojs/node";
3
4 export default defineConfig({
5   output: "server",
6   adapter: node({ mode: "standalone" }),
7   i18n: {
8     defaultLocale: "de",
9     locales: ["de", "en"],
10    routing: {
11      prefixDefaultLocale: true,
12      redirectToDefaultLocale: true,
13    },
14  },
15 });
```

astro.config.ts

SERVERADAPTER

```
1 ---
2 export const prerender = true;
3
4 import { getCollection } from "astro:content";
5 import BaseLayout from "../../layouts/BaseLayout.astro";
6 import RecipeDetailView from "../../components/views/RecipeDetailView.astro";
7 ---
```

ACTIONS

- Typisierte, serverseitige Funktionen
- aufrufbar wie normale JS-Funktionen

SERVER ISLANDS

- dynamische, on-demand gerenderte Inseln

```
1 ---
2 import ServerHighlights from "../ServerHighlights.astro";
3 ---
4
5 ...
6
7 <ServerHighlights server:defer lang={lang} currentSlug={recipe.id}>
8   <p slot="fallback">{t("server.loading")}</p>
9 </ServerHighlights>
10
11 ...
```

components/views/RecipeDetailView.astro

SSR

- komplette Seite on-demand gerendert

USE CASES FÜR SERVER-FUNKTIONEN

- Authentifizierung & personalisierte Inhalte
- CAPTCHAs
- Häufig wechselnde, nicht vorab bekannte Daten



WANN SPAS DIE BESSERE WAHL SIND

→ KOMPLEXE, HOCHINTERAKTIVE ANWENDUNGEN MIT VIEL
CLIENT-STATE

z.B. Dashboards, Editoren, Tools mit dauerhaftem "App-Charakter"

EIN FRAGENKATALOG

für euer nächstes Projekt

1. Wie viel der Seite ist tatsächlich interaktiv vs. nur reiner Content?
2. Wird clientseitiger Zustand benötigt, der über viele Komponenten hinweg geteilt und synchron gehalten werden muss?
3. Ist SEO bzw. die Core Web Vitals ein kritischer Faktor?
4. Besteht bereits eine erhebliche Investition in ein bestehendes SPA-Setup?

Die entscheidende Frage ist nicht "Welches Framework?", sondern
"Wie viel JavaScript braucht diese Seite tatsächlich?"

Vasco Schwarze · vasco.schwarze@sciota.io · GitHub: VascoSchwarze

Slides & Code unter: <https://github.com/VascoSchwarze/do-you-really-need-an-spa>