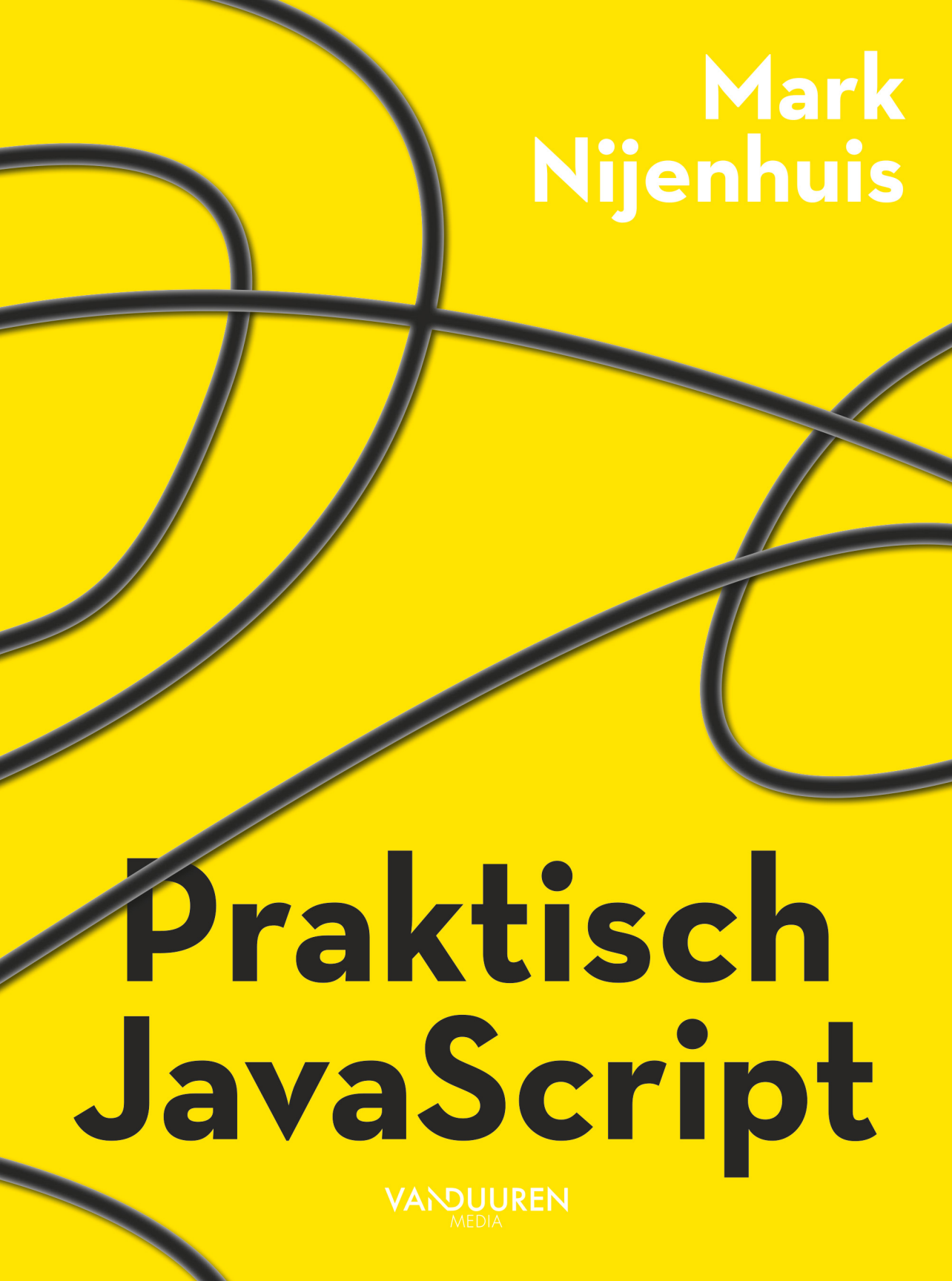
Mark
Nijenhuis
Praktisch
JavaScript
VANDUUREN
MEDIA

Praktisch JavaScript

Mark Nijenhuis

VANDUUREN
MEDIA

Inhoud

Hoofdstuk 1

Inleiding: context en geschiedenis

JavaScript is een van de meest invloedrijke en veelzijdige programmeertalen die de wereld van webontwikkeling heeft getransformeerd. Oorspronkelijk ontwikkeld door Netscape in de jaren negentig, heeft JavaScript zich snel ontwikkeld van een eenvoudige scripttaal voor webpagina's tot een krachtige, veelzijdige taal die aan de basis staat van moderne webapplicaties. Maar wat maakt JavaScript zo speciaal, en waarom is het zo belangrijk voor front-end development?

In front-end development is JavaScript onmisbaar geworden. Het is de enige programmeertaal die door alle moderne webbrowsers native wordt ondersteund, wat betekent dat het essentieel is voor het bouwen van interactieve webapplicaties. Bovendien is de kennis van JavaScript een fundament voor het leren van populaire front-end frameworks zoals React, Vue.js en Angular.

Dit handboek bestaat uit een aantal mini-projecten rond één centrale applicatie, waarbij je in elk project kennis maakt met één of meer aspecten van geavanceerd front-end development en direct leer toepassen in de praktijk.

Door deze projecten leer je niet alleen de taal, maar ook hoe je JavaScript kunt inzetten binnen moderne front-end frameworks en met veelgebruikte libraries. Daarnaast geven we ook aandacht aan TypeScript, een superset van JavaScript die je helpt om beter gestructureerde en onderhoudbare code te schrijven.

Wat maakt JavaScript uniek?

JavaScript onderscheidt zich van andere programmeertalen door verschillende kernkenmerken:

- **Interactiviteit** JavaScript brengt webpagina's tot leven door interactie met gebruikers mogelijk te maken. Denk aan formuliervalidaties, dynamische inhoudsupdates en animaties.
- **Platformonafhankelijkheid** JavaScript kan in elke moderne webbrowser worden uitgevoerd, zonder dat er extra software nodig is.
- **Breed toepassingsgebied** JavaScript heeft een breed scala aan toepassingen: van eenvoudige scripting tot het bouwen van complexe webapplicaties en server-side development.
- **Actieve gemeenschap** JavaScript heeft een grote en actieve gemeenschap die voortdurend bijdraagt aan de ontwikkeling van de taal en bijbehorende tools, frameworks en libraries.

Een korte geschiedenis

JavaScript werd in 1995 gecreëerd door Brendan Eich tijdens zijn werk bij Netscape Communications. Oorspronkelijk heette het "Mocha", later "LiveScript" en uiteindelijk "JavaScript" om te profiteren van de populariteit van Java destijds. Hoewel de naam suggereert dat het iets te maken heeft met Java, zijn de twee talen fundamenteel verschillend. JavaScript was bedoeld om webpagina's dynamischer te maken en interactie met de gebruiker mogelijk te maken zonder de pagina opnieuw te laden.

De evolutie van JavaScript

Vanaf het bescheiden begin als eenvoudige scripttaal is JavaScript geëvolueerd tot een volledige programmeertaal die aan zowel de front-end als de back-end (server sided JavaScript) van webontwikkeling kan worden gebruikt. Enkele belangrijke mijlpalen in deze evolutie zijn:

- **ES6 en latere versies** De introductie van ECMAScript 6 (ES6) in 2015 bracht een groot aantal nieuwe features zoals arrow functions, let en const, klassen, modules en veel meer, waardoor de taal moderner en robuuster werd.
- **Node.js** De introductie van Node.js in 2009 maakte het mogelijk om JavaScript ook op webservers te gebruiken, wat leidde tot de ontwikkeling van full-stack JavaScript-toepassingen.
- **Moderne frameworks en libraries** Bibliotheken zoals jQuery maakten het gemakkelijker om met JavaScript te werken, en moderne frameworks zoals React, Angular en Vue.js hebben de manier waarop we webapplicaties bouwen volledig getransformeerd.

De jaren van jQuery

In de vroege jaren van het web waren ontwikkelaars beperkt tot het gebruik van rudimentaire JavaScript om interactieve elementen aan hun websites toe te voegen. Dit veranderde in 2006 met de introductie van jQuery, een snelle, kleine en feature-rijke JavaScript-bibliotheek. jQuery maakte het aanzienlijk eenvoudiger om met DOM-elementen te werken, asynchrone verzoeken te doen (AJAX) en animaties te creëren. De bekendste slogan van jQuery, “Write less, do more”, vatte perfect samen wat jQuery deed: het vereenvoudigde veelvoorkomende taken en bood een consistente API die in alle browsers werkte.

Hoogtepunten van jQuery:

- **Eenvoudige DOM-manipulatie** Met methoden zoals `.hide()`, `.show()` en `.animate()` konden ontwikkelaars complexe interacties realiseren met weinig code.
- **Cross-browser compatibiliteit** jQuery zorgde ervoor dat code in alle grote browsers hetzelfde werkte, wat een grote uitdaging was in de tijd van browserinconsistenties.
- **AJAX-ondersteuning** jQuery maakte het makkelijk om asynchrone HTTP-verzoeken te doen, wat de basis legde voor dynamische, interactieve webpagina’s zonder dat ze volledig opnieuw herlaad hoefden te worden.

De opkomst van moderne frameworks

Hoewel jQuery veel heeft bijgedragen aan de webontwikkeling, waren er nog steeds beperkingen. Naarmate webapplicaties complexer werden ontstond de behoefte aan meer gestructureerde en schaalbare oplossingen. Dit leidde tot de opkomst van moderne JavaScript-frameworks zoals Angular, React en Vue.js.

Angular (2010)

AngularJS, geïntroduceerd door Google in 2010, was een van de eerste frameworks die een complete oplossing bood voor het bouwen van webapplicaties. Het bracht het concept van tweeweg databinding, dependency injection en een modulaire architectuur, waardoor ontwikkelaars robuustere en onderhoudbare applicaties konden bouwen.

- **Tweeweg databinding** Angular synchroniseerde automatisch de data tussen het model en de view, wat de ontwikkeling aanzienlijk vereenvoudigde.
- **Dependency injection** Dit maakte het mogelijk om services en componenten op een schaalbare manier te beheren.
- **Directives en componenten** Angular introduceerde een declaratieve manier om DOM-elementen te verrijken met gedrag.

React (2013)

React, ontwikkeld door Facebook en geïntroduceerd in 2013, bracht een paradigmaverschuiving in hoe we over UI-ontwikkeling denken. React introduceerde een component-gebaseerde architectuur en unidirectionele datastroom, wat leidde tot meer voorspelbare en eenvoudig te debuggen applicaties.

- **Componenten** React-componenten maken het mogelijk om UI-elementen te hergebruiken en te componeren, wat de ontwikkeling van complexe interfaces vereenvoudigt.
- **Virtual DOM** React introduceerde het concept van een *virtual DOM*, wat leidt tot snellere en efficiëntere updates van de user interface.
- **Ecosysteem** Met tools zoals React Router en statemanagementlibraries zoals Redux biedt React een compleet ecosysteem voor het bouwen van grootschalige applicaties.

Vue.js (2014)

Vue.js, ontwikkeld door Evan You en geïntroduceerd in 2014, combineert de beste aspecten van Angular en React in een lichtgewicht en flexibel framework. Vue is bijzonder populair geworden vanwege zijn eenvoud en krachtige features.

- **Reactieve databinding** Net als Angular biedt Vue tweerichtingsdatabinding, maar op een eenvoudiger en intuïtieve manier.
- **Component-gebaseerde architectuur** Net als React maakt Vue gebruik van een component-gebaseerde benadering, wat leidt tot herbruikbare en modulaire code.
- **Eenvoudig te Leren** Vue's leercurve is lager dan die van Angular en React, waardoor het een ideale keuze is voor zowel beginners als ervaren ontwikkelaars.

De toekomst van JavaScript-frameworks

De evolutie van JavaScript frameworks is nog lang niet voorbij. Met de voortdurende ontwikkeling van technologieën en de behoefte aan steeds complexere webapplicaties, zullen nieuwe tools en frameworks blijven opkomen. Moderne frameworks blijven zich ontwikkelen om beter te presteren, ontwikkelaarsproductiviteit te verhogen en gebruikerservaringen te verbeteren.

JavaScripts reis van jQuery naar moderne frameworks toont de kracht van innovatie en de voortdurende inspanningen van de ontwikkelaarsgemeenschap om de taal en haar toepassingen te verbeteren. In dit handboek gaan we dieper in op hoe je deze moderne tools en frameworks kunt gebruiken om robuuste, efficiënte en schaalbare webapplicaties te bouwen. We zullen je begeleiden door praktische mini-projecten die je de vaardigheden en kennis geven om JavaScript effectief te gebruiken in de context van moderne front-end development.

Waarom dit boek?

Het is essentieel om bij te blijven met de nieuwste technologieën en *best practices*. JavaScript, als de onbetwiste ruggengraat van moderne webapplicaties, speelt hierin een cruciale rol. Dit boek is bedoeld voor ontwikkelaars die al een basiskennis van JavaScript hebben en hun vaardigheden willen aanscherpen met de nieuwste technieken en frameworks.

Praktijkgerichte benadering

Een van de grootste uitdagingen bij het leren van nieuwe technologieën is het overbruggen van de kloof tussen theoretische kennis en praktische toepassing. Dit boek hanteert een unieke, praktijkgerichte benadering door de lezer te begeleiden via een centraal project: een geheugenspel dat steeds verder uitbreidt in complexiteit en functionaliteit. Elk onderdeel is zorgvuldig ontworpen om je stap voor stap door de belangrijkste concepten van JavaScript te leiden, met een focus op de integratie en het gebruik binnen populaire frameworks zoals React, Vue.js en Angular (in het project gebruiken we Vue.js maar bespreken ook overeenkomsten met Angular en hier en daar met React).

Breed scala aan onderwerpen

JavaScript is een veelzijdige taal met een breed scala aan toepassingen. Uitgaand van een basiskennis JavaScript bouwen we verder en bespreken we ES6+-features tot TypeScript, een superset van JavaScript die steeds populairder wordt vanwege zijn vermogen om de leesbaarheid en onderhoudbaarheid van code te verbeteren.

Toepassing van webcomponenten

We beginnen met een inleiding tot webcomponenten, custom elements, shadow-DOM en shadow-root, en laten zien hoe je een eenvoudige versie van een memorygame kunt omzetten naar een custom element. Dit hoofdstuk biedt een solide basis voor het begrijpen van encapsulatie en herbruikbaarheid in moderne webontwikkeling.

Declaratief programmeren

Een van de meest krachtige en efficiënte benaderingen in moderne softwareontwikkeling is declaratief programmeren. In plaats van stap voor stap te definiëren hoe een resultaat moet worden bereikt, beschrijft declaratief programmeren wat het gewenste eindresultaat is en laat het de onderliggende systemen de details afhandelen. In dit hoofdstuk gaan we dieper in op het concept van declaratief programmeren en laten we zien hoe dit principe kan worden toegepast binnen de context van JavaScript. We zullen verschillende technieken verkennen, waaronder het gebruik van moderne methoden zoals `.map()` en template literals, om leesbare, onderhoudbare en schaalbare code te schrijven. Door middel van een prak-

tisch voorbeeld, zoals het bouwen van een memorygame, laten we zien hoe een declaratieve benadering niet alleen de ontwikkeling vereenvoudigt, maar ook de robuustheid en flexibiliteit van je codebase verbetert.

State en state management

Een cruciaal aspect van elke interactieve webapplicatie is het beheer van de *state* – de verzameling gegevens die de huidige status van de applicatie vertegenwoordigen. State management betreft het volgen en bijwerken van deze gegevens op een gestructureerde en consistente manier. In dit hoofdstuk onderzoeken we de basisprincipes van state en state management binnen de context van JavaScript-applicaties. We behandelen verschillende technieken en patronen om de state effectief te beheren, waaronder het gebruik van state-objecten, asynchrone updates en het gebruik van de local storage. Aan de hand van het memory game project laten we zien hoe goed georganiseerd state management de leesbaarheid, onderhoudbaarheid en schaalbaarheid van je code aanzienlijk kan verbeteren. Dit hoofdstuk biedt een stevige basis om complexere applicaties te bouwen waarbij state management een centrale rol speelt.

TypeScript

TypeScript heeft snel aan populariteit gewonnen als een krachtige uitbreiding van JavaScript. TypeScript biedt statische typecontrole en geavanceerde IDE-ondersteuning, waardoor ontwikkelaars eerder fouten kunnen opsporen en robuustere code kunnen schrijven. In dit hoofdstuk introduceren we TypeScript en laten we zien hoe het kan worden geïntegreerd in je JavaScript-projecten om de betrouwbaarheid en onderhoudbaarheid van je code te verbeteren. We bespreken de basisconcepten van TypeScript, zoals typen, interfaces en type-annotaties, en demonstreren de voordelen door onze bestaande JavaScript-code voor de memorygame om te zetten naar TypeScript. Door te leren werken met TypeScript krijg je een krachtig hulpmiddel in handen dat je helpt om complexere en foutbestendige applicaties te bouwen

Gebruik binnen frameworks

Webcomponenten kunnen ook naadloos geïntegreerd worden in frameworks zoals Angular, Vue.js en React. Dit boek legt uit hoe je webcomponenten kunt gebruiken binnen deze frameworks, met praktische voorbeelden en best practices. We bespreken ook populaire UI-libraries zoals Ionic en hoe je deze kunt gebruiken om moderne, responsieve webapplicaties te bouwen.

Uitdagingen en oplossingen

Bij het werken met webcomponenten kunnen er uitdagingen ontstaan, zoals het doorgeven van attributen aan child components en het opvangen van events van child components. Dit boek biedt concrete oplossingen en technieken om deze uitdagingen aan te pakken, zodat je effectief en efficiënt kunt werken met webcomponenten in je projecten.

Asynchroon programmeren

Op het moment dat we gebruik gaan maken van externe bronnen en deze benaderen via een web-API moeten we leren asynchroon te programmeren en dus leren werken met de Fetch()-API en promises. We gaan veel `async...await`-functies en methoden schrijven. Ook bespreken we Axios, een bibliotheek die meer opties biedt dan de vanilla Fetch()-API.

Datavisualisatie

Veel front-end applicaties zijn data-driven en bevatten datavisualisatie in verschillende vormen, bijvoorbeeld in de vorm van dashboards. We gaan de resultaten van de speler middels datavisualisatie weergeven met behulp van de JavaScript-bilbliotheek chartJS.

Meer state management

Om State management nog verder uit te bouwen en zelfs persistent te maken gaan we gebruik maken van een Pinia-store en Firebase als externe bron. We zorgen ervoor dat de gebruiker kan inloggen en zo altijd toegang heeft tot zijn of haar gegevens en de staat van het spel. Ook kan de gebruiker zijn gegevens aanpassen en bouwen we een notificatieservice.

PWA

Om het spel als app te kunnen installeren op een mobiel apparaat moeten we er een PWA van maken. We voegen een serviceworker en een manifest toe, en zaken als icons. Het eindresultaat is op elk mobiel apparaat te installeren!

Bonus

In aanvullende code (te downloaden van Github) wordt een multiplayervariant van dit spel geïntroduceerd. Die is niet geheel functionerend en vormt een mooie uitdaging om dit zelfstandig af te bouwen.

Debuggen en testen

Omdat debuggen en testen eigenlijk iets is wat je gedurende het hele proces doet en je hier misschien af en toe naar terug wilt bladeren hebben we dit als apart hoofdstuk opgenomen, aan het eind van het boek. Dat betekent dus niet dat je dit kunt skippen tot het eind; er staan waardevolle tips in die je tijdens het oefenen nodig hebt. Gebruik dit hoofdstuk dus als ‘EHBO’-kit tijdens het oefenen, met stappen en technieken die je helpen om codeproblemen snel te begrijpen en op te lossen en om je applicatie betrouwbaarder te maken. Elke ontwikkelaar heeft er baat bij om deze aanpakken vroeg in het proces toe te passen.

Toegankelijk voor ontwikkelaars met basiskennis

Dit boek is specifiek geschreven voor ontwikkelaars die al bekend zijn met de basisprincipes van JavaScript. We gaan ervan uit dat je bekend bent met datatypen, functies, arrays, objecten, operators, vergelijkingen, events, loops en wis-

kundige bewerkingen. Hierdoor kunnen we ons richten op meer geavanceerde onderwerpen en praktische toepassingen.

Up-to-date kennis

De technologieën en best practices in webontwikkeling veranderen voortdurend. Dit boek is geschreven met de meest recente informatie en technieken om ervoor te zorgen dat je kennis actueel blijft. We hebben de nieuwste versies van frameworks en tools gebruikt om ervoor te zorgen dat je met de meest relevante en effectieve methoden werkt.

Community en ondersteuning

Leren stopt niet bij het lezen van een boek. Daarom moedigen we je aan om deel te nemen aan de bredere ontwikkelaarscommunity. Door actief deel te nemen aan forums, meetups en online communities, kun je je netwerk uitbreiden, nieuwe inzichten opdoen en vragen stellen als je vastloopt. We bieden ook een begeleidende GitHub-repository waar je de code van de projecten kunt downloaden en kunt bijdragen aan discussies en verbeteringen.

Hoe dit boek te gebruiken

Dit boek is ontworpen voor ontwikkelaars met een basiskennis van JavaScript die hun vaardigheden willen uitbreiden en zich willen verdiepen in modern front-end development technieken. Hieronder vind je een overzicht van hoe je het meeste uit dit boek kunt halen:

Stapsgewijs leren

Het boek is opgebouwd in een logische volgorde, beginnend bij web componenten en geleidelijk voortbouwend naar complexere onderwerpen. Elk hoofdstuk is ontworpen om voort te bouwen op de kennis die in eerdere hoofdstukken is opgedaan. Het is daarom raadzaam om de hoofdstukken in volgorde door te werken, vooral als je nog niet bekend bent met JavaScript-bibliotheken en front-end frameworks.

Codevoorbeelden en oefeningen

Het boek bevat tal van codevoorbeelden en oefeningen die je kunt gebruiken om je begrip te testen. Zorg ervoor dat je deze oefeningen maakt en de codevoorbeelden uitvoert op je eigen computer. Experimenteer met de code, maak aanpassingen en zie wat er gebeurt. Dit is een uitstekende manier om je vaardigheden te versterken en te leren omgaan met echte problemen die je tegenkomt tijdens het coderen.

Begeleidende GitHub-repository

Bij dit boek hoort een begeleidende GitHub-repository waar je de code voor alle projecten kunt vinden. Deze repository wordt regelmatig bijgewerkt en is een waardevolle bron voor het volgen van de projecten in dit boek. Je kunt hier ook je eigen oplossingen uploaden, vragen stellen en bijdragen aan verbeteringen. Dit helpt je niet alleen bij het leren, maar geeft je ook ervaring met het werken in een gedeelde codebase, wat een belangrijke vaardigheid is voor elke ontwikkelaar.

Blijf up-to-date

Webontwikkeling is een dynamisch vakgebied dat voortdurend in beweging is. De technologieën en best practices die vandaag de dag populair zijn, kunnen morgen alweer veranderd zijn. We raden je aan om regelmatig op de hoogte te blijven van de nieuwste ontwikkelingen door blogs, nieuwsbrieven en tech-sites te volgen. Dit boek biedt een solide basis, maar het is belangrijk om je kennis continu bij te werken en nieuwe vaardigheden te blijven leren.

Feedback en verbetering

We waarderen je feedback over dit boek. Als je fouten ontdekt, suggesties hebt voor verbeteringen of gewoon je ervaringen wilt delen, aarzel dan niet om contact met ons op te nemen. Je input helpt ons om toekomstige edities van dit boek nog beter te maken en nuttiger voor lezers zoals jij.

Door deze richtlijnen te volgen haal je het maximale uit dit handboek. We hopen dat het je helpt om een sterke basis te leggen in JavaScript en moderne front-end development, en dat je geïnspireerd raakt om geweldige webapplicaties te bouwen. Laten we aan de slag gaan en de wereld van JavaScript verkennen!

Gratis bonushoofdstuk bij registratie

Wanneer je dit boek registreert (zie de informatie aan het begin van dit boek), ontvang je een mail met een link naar een gratis bonushoofdstuk: *Debuggen, testen en TDD in JavaScript-projecten*. Registreer je boek dus vandaag nog en download dit gratis hoofdstuk meteen.