Maintainable Javascript Writing Readable Code Ebook

How JavaScript Works
Maintainable Javascript
Build Better Applications with Coding and Design Patterns
Coders at Work
Professional Ajax
Perl Best Practices
How to improve your JavaScript programs using functional techniques
Web Programming with HTML5, CSS, and JavaScript
Ten Guidelines for Future-Proof Code
Maintainable JavaScript
New features of ECMAScript 2015, 2016, and beyond
Why Smart Engineers Write Bad Code
Mastering the JavaTM Persistence API
The Principles of Object-Oriented JavaScript
Learning JavaScript Design Patterns
Building Maintainable Software, Java Edition
Develop reliable, maintainable, and robust JavaScript
Programming JavaScript Applications
Working Effectively with Legacy Code
Reflections on the Craft of Programming
The Problem with Software
Simplifying JavaScript
JavaScript: The Good Parts
Writing Modern JavaScript with ES5, ES6, and Beyond
Master the World's Most-Used Programming Language
The Good Parts
Writing Maintainable Unit Tests: Mastering the Art of Loosely Coupled Unit Tests
Turning Bad Code Into Good Code
Five Lines of Code
Async JavaScript
JavaScript: The Definitive Guide
Pro JPA 2
A JavaScript and jQuery Developer's Guide
Robust Python
How and when to refactor
with examples in C#
Writing Readable Code
Build Faster Web Application Interfaces
Object-Oriented JavaScript - Second Edition
JavaScript: Best Practice

CHANEL HOPE

How JavaScript Works "O'Reilly Media, Inc."
Peter Seibel interviews 15 of the most interesting computer programmers alive today in Coders at Work, offering a companion volume to Apress's highly acclaimed best-seller Founders at Work by Jessica Livingston. As the words "at work" suggest, Peter Seibel focuses on how his interviewees tackle the day-to-day work of programming, while revealing much more, like how they became great programmers, how they recognize programming talent in others, and what kinds of problems they find most interesting. Hundreds of people have suggested names of programmers to interview on the Coders at Work web site: www.codersatwork.com. The complete list was 284 names. Having digested everyone's feedback, we selected 15 folks who've been kind enough to agree to be interviewed: Frances Allen: Pioneer in optimizing compilers, first woman to win the Turing Award (2006) and first female IBM fellow Joe Armstrong: Inventor of Erlang Joshua Bloch: Author of the Java collections framework, now at Google Bernie Cosell: One of the main software guys behind the original ARPANET IMPs and a master debugger Douglas Crockford: JSON founder, JavaScript architect at Yahoo! L. Peter Deutsch: Author of Ghostscript, implementer of Smalltalk-80 at Xerox PARC and Lisp 1.5 on PDP-1 Brendan Eich: Inventor of JavaScript, CTO of the Mozilla Corporation Brad Fitzpatrick: Writer of LiveJournal, OpenID, memcached, and Perlbal Dan Ingalls: Smalltalk implementor and designer Simon Peyton Jones: Coinventor of Haskell and lead designer of Glasgow Haskell Compiler Donald Knuth: Author of The Art of Computer Programming and creator of TeX Peter Norvig: Director of Research at Google and author of the standard text on AI Guy Steele: Coinventor of Scheme and part of the Common Lisp Gang of Five, currently working on Fortress Ken Thompson: Inventor of UNIX Jamie Zawinski: Author of XEmacs and early Netscape/Mozilla hacker

Maintainable Javascript Pearson Education
This book is for experienced software developers who want to improve upon their existing skills in writing unit tests. You will learn how to build loosely coupled, highly maintainable and robust unit tests that are trustworthy and improve the overall code quality of your software applications. The content of this book is based on 15+ years of experience with Test-Driven Development. Although the examples in this book are written in C#, the principles and guidance are broadly applicable to other platforms and programming environments as well (Java, Python, JavaScript, etc.). You will be able to universally apply this knowledge throughout the rest of your career.

Build Better Applications with Coding and Design Patterns John Wiley & Sons
Looks at the principles and clean code, includes case studies showcasing the practices of writing clean code, and contains a list of heuristics and "smells" accumulated from the process of writing clean code.

Coders at Work "O'Reilly Media, Inc."
"Writing readable code"--Cover.

Professional Ajax Packt Publishing Ltd

Maintainable JavaScriptWriting Readable Code"O'Reilly Media, Inc."
Perl Best Practices "O'Reilly Media, Inc."
Does it seem like your Python projects are getting bigger and bigger? Are you feeling the pain as your codebase expands and gets tougher to debug and maintain? Python is an easy language to learn and use, but that also means systems can quickly grow beyond comprehension. Thankfully, Python has features to help developers overcome maintainability woes. In this practical book, author Patrick Viafore shows you how to use Python's type system to the max. You'll look at user-defined types, such as classes and enums, and Python's type hinting system. You'll also learn how to make Python extensible and how to use a comprehensive testing strategy as a safety net. With these tips and techniques, you'll write clearer and more maintainable code. Learn why types are essential in modern development ecosystems Understand how type choices such as classes, dictionaries, and enums reflect specific intents Make Python extensible for the future without adding bloat Use popular Python tools to increase the safety and robustness of your codebase Evaluate current code to detect common maintainability gotchas Build a safety net around your codebase with linters and tests
How to improve your JavaScript programs using functional techniques MIT Press
If you have a working knowledge of JavaScript and ECMAScript 6 (ES6), this practical guide will help you tackle modular programming to produce code that’s readable, maintainable, and scalable. You’ll learn the fundamentals of modular architecture with JavaScript and the benefits of writing self-contained code at every system level, including the client and server. Nicolás Bevacqua, author of Practical Modern JavaScript, demonstrates how to scale out JavaScript applications by breaking codebases into smaller modules. By following the design practices in this book, senior developers, technical leaders, and software architects will learn how to create modules that are simple and flexible while keeping internal complexity in check. Learn modular design essentials, including how your application will be consumed and what belongs on the interface Design module internals to keep your code readable and its intent clear Reduce complexity by refactoring code and containing and eliminating state Take advantage of modern JavaScript features to write clear programs and reduce complexity Apply Twelve-Factor App principles to frontend and backend JavaScript application development
Web Programming with HTML5, CSS, and JavaScript Apress
JavaScript is one of the easiest, most straightforward ways to enhance a website with interactivity. Sams Teach Yourself JavaScript in 24 Hours, 4th Edition serves as an easy-to-understand tutorial on both scripting basics and JavaScript itself. The book is written in a clear and personable style with an extensive use of practical, complete examples. It also includes material on the latest developments in JavaScript and web scripting. You will learn how to use JavaScript to enhance web pages with interactive forms, objects, and cookies, as well as how to use JavaScript to work with games, animation, and multimedia.
Ten Guidelines for Future-Proof Code Prentice Hall Professional
If you've used a more traditional object-oriented language, such as C++ or Java,

JavaScript probably doesn't seem object-oriented at all. It has no concept of classes, and you don't even need to define any objects in order to write code. But don't be fooled—JavaScript is an incredibly powerful and expressive object-oriented language that puts many design decisions right into your hands. In The Principles of Object-Oriented JavaScript, Nicholas C. Zakas thoroughly explores JavaScript's object-oriented nature, revealing the language's unique implementation of inheritance and other key characteristics. You'll learn: –The difference between primitive and reference values –What makes JavaScript functions so unique –The various ways to create objects –How to define your own constructors –How to work with and understand prototypes –Inheritance patterns for types and objects The Principles of Object-Oriented JavaScript will leave even experienced developers with a deeper understanding of JavaScript. Unlock the secrets behind how objects work in JavaScript so you can write clearer, more flexible, and more efficient code.

Maintainable JavaScript "O'Reilly Media, Inc."

Get the most out of JavaScript for building web applications through a series of patterns, techniques, and case studies for clean coding Key Features Write maintainable JS code using internal abstraction, well-written tests, and well-documented code Understand the agents of clean coding like SOLID principles, OOP, and functional programming Explore solutions to tackle common JavaScript challenges in building UIs, managing APIs, and writing states Book Description Building robust apps starts with creating clean code. In this book, you'll explore techniques for doing this by learning everything from the basics of JavaScript through to the practices of clean code. You'll write functional, intuitive, and maintainable code while also understanding how your code affects the end user and the wider community. The book starts with popular clean-coding principles such as SOLID, and the Law of Demeter (LoD), along with highlighting the enemies of writing clean code such as cargo culting and over-management. You'll then delve into JavaScript, understanding the more complex aspects of the language. Next, you'll create meaningful abstractions using design patterns, such as the Class Pattern and the Revealing Module Pattern. You'll explore real-world challenges such as DOM reconciliation, state management, dependency management, and security, both within browser and server environments. Later, you'll cover tooling and testing methodologies and the importance of documenting code. Finally, the book will focus on advocacy and good communication for improving code cleanliness within teams or workplaces, along with covering a case study for clean coding. By the end of this book, you'll be well-versed with JavaScript and have learned how to create clean abstractions, test them, and communicate about them via documentation. What you will learn Understand the true purpose of code and the problems it solves for your end-users and colleagues Discover the tenets and enemies of clean code considering the effects of cultural and syntactic conventions Use modern JavaScript syntax and design patterns to craft intuitive abstractions Maintain code quality within your team via wise adoption of tooling and advocating best practices Learn the modern ecosystem of JavaScript and its challenges like DOM

reconciliation and state management Express the behavior of your code both within tests and via various forms of documentation Who this book is for This book is for anyone who writes JavaScript, professionally or otherwise. As this book does not relate specifically to any particular framework or environment, no prior experience of any JavaScript web framework is required. Some knowledge of programming is assumed to understand the concepts covered in the book more effectively.

New features of ECMAScript 2015, 2016, and beyond John Wiley & Sons

Douglas Crockford starts by looking at the fundamentals: names, numbers, booleans, characters, and bottom values. JavaScript's number type is shown to be faulty and limiting, but then Crockford shows how to repair those problems. He then moves on to data structures and functions, exploring the underlying mechanisms and then uses higher order functions to achieve class-free object oriented programming. The book also looks at eventual programming, testing, and purity, all the while looking at the requirements of The Next Language. Most of our languages are deeply rooted in the paradigm that produced FORTRAN. Crockford attacks those roots, liberating us to consider the next paradigm.He also presents a strawman language and develops a complete transpiler to implement it. The book is deep, dense, full of code, and has moments when it is intentionally funny.

Why Smart Engineers Write Bad Code Pearson Education

Summary Get Programming with JavaScript Next introduces the modern age of JavaScript programming with ES6 and ES7 without dragging you through confusing jargon and abstract examples you'll never use. In just 34 quick-fire sessions, you'll quickly be coding with the latest features and functions of ES6 and ES7! Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Great code is readable, bug-free, and maintainable. Modern JavaScript, aka JavaScript Next, makes it much, much easier to write great applications. New features introduced in ES2015 simplify the structure of your JavaScript projects and radically streamline async-oriented tasks like writing reactive applications and microservices. About the Book Get Programming with JavaScript Next introduces you to the new features included in the ES2015-and-later JavaScript releases. You'll learn example by example in 34 short lessons, each designed to drive home a specific skill. The coverage is complete: you'll explore new language syntax, declarations, and data types. You'll structure code with modules, replace callbacks with promises, and use classes instead of constructors. Every time you turn a page, complete an exercise, or study a carefully crafted illustration, you'll be one step closer to JavaScript mastery. What's Inside New features from ES2015 and later Writing asynchronous code Creating custom iterables Troubleshooting modules and classes About the Reader Written for web developers comfortable with standard JavaScript 5 features and coding style. About the Author J.D. Isaacks is a seasoned developer, a JavaScript instructor, and an open source maintainer. Table of Contents Lesson 1 - ECMAScript specification and the proposal process Lesson 2 - Transpiling with Babel Lesson 3 - Bundling modules with Browserify UNIT 1 - VARIABLES AND

STRINGS Lesson 4 - Declaring variables with let Lesson 5 - Declaring constants with const Lesson 6 - New string methods Lesson 7 - Template literals Lesson 8 - Capstone: Building a domain-specific language UNIT 2 - OBJECTS AND ARRAYS Lesson 9 - New array methods Lesson 10 - Object.assign Lesson 11 - Destructuring Lesson 12 - New object literal syntax Lesson 13 - Symbol-a new primitive Lesson 14 - Capstone: Simulating a lock and key UNIT 3 - FUNCTIONS Lesson 15 - Default parameters and rest Lesson 16 - Destructuring parameters Lesson 17 - Arrow functions Lesson 18 - Generator functions Lesson 19 - Capstone: The prisoner's dilemma UNIT 4 - MODULES Lesson 20 - Creating modules Lesson 21 - Using modules Lesson 22 - Capstone: Hangman game UNIT 5 - ITERABLES Lesson 23 - Iterables Lesson 24 - Sets Lesson 25 - Maps Lesson 26 - Capstone: Blackjack UNIT 6 - CLASSES Lesson 27 - Classes Lesson 28 - Extending classes Lesson 29 - Capstone: Comets UNIT 7 - WORKING ASYNCHRONOUSLY Lesson 30 - Promises Lesson 31 - Advanced promises Lesson 32 - Async functions Lesson 33 - Observables Lesson 34 - Capstone: Canvas image gallery Appendix - Exercise answers

Mastering the JavaTM Persistence API "O'Reilly Media, Inc."

With Learning JavaScript Design Patterns, you'll learn how to write beautiful, structured, and maintainable JavaScript by applying classical and modern design patterns to the language. If you want to keep your code efficient, more manageable, and up-to-date with the latest best practices, this book is for you. Explore many popular design patterns, including Modules, Observers, Facades, and Mediators. Learn how modern architectural patterns—such as MVC, MVP, and MVVM—are useful from the perspective of a modern web application developer. This book also walks experienced JavaScript developers through modern module formats, how to namespace code effectively, and other essential topics. Learn the structure of design patterns and how they are written Understand different pattern categories, including creational, structural, and behavioral Walk through more than 20 classical and modern design patterns in JavaScript Use several options for writing modular code—including the Module pattern, Asyncronous Module Definition (AMD), and CommonJS Discover design patterns implemented in the jQuery library Learn popular design patterns for writing maintainable jQuery plug-ins "This book should be in every JavaScript developer's hands. It's the go-to book on JavaScript patterns that will be read and referenced many times in the future."—Andrée Hansson, Lead Front-End Developer, presis!

The Principles of Object-Oriented JavaScript SitePoint

Get more out of your legacy systems: more performance, functionality, reliability, and manageability Is your code easy to change? Can you get nearly instantaneous feedback when you do change it? Do you understand it? If the answer to any of these questions is no, you have legacy code, and it is draining time and money away from your development efforts. In this book, Michael Feathers offers start-to-finish strategies for working more effectively with large, untested legacy code bases. This book draws on material Michael created for his renowned Object Mentor seminars: techniques Michael has used in mentoring to help hundreds of developers, technical managers, and

testers bring their legacy systems under control. The topics covered include Understanding the mechanics of software change: adding features, fixing bugs, improving design, optimizing performance Getting legacy code into a test harness Writing tests that protect you against introducing new problems Techniques that can be used with any language or platform—with examples in Java, C++, C, and C# Accurately identifying where code changes need to be made Coping with legacy systems that aren't object-oriented Handling applications that don't seem to have any structure This book also includes a catalog of twenty-four dependency-breaking techniques that help you work with program elements in isolation and make safer changes.

Learning JavaScript Design Patterns

Maintainable JavaScriptWriting Readable Code

What's the best approach for developing an application with JavaScript? This book helps you answer that question with numerous JavaScript coding patterns and best practices. If you're an experienced developer looking to solve problems related to objects, functions, inheritance, and other language-specific categories, the abstractions and code templates in this guide are ideal—whether you're using JavaScript to write a client-side, server-side, or desktop application. Written by JavaScript expert Stoyan Stefanov—Senior Yahoo! Technical and architect of YSlow 2.0, the web page performance optimization tool—JavaScript Patterns includes practical advice for implementing each pattern discussed, along with several hands-on examples. You'll also learn about anti-patterns: common programming approaches that cause more problems than they solve. Explore useful habits for writing high-quality JavaScript code, such as avoiding globals, using single var declarations, and more Learn why literal notation patterns are simpler alternatives to constructor functions Discover different ways to define a function in JavaScript Create objects that go beyond the basic patterns of using object literals and constructor functions Learn the options available for code reuse and inheritance in JavaScript Study sample JavaScript approaches to common design patterns such as Singleton, Factory, Decorator, and more Examine patterns that apply specifically to the client-side browser environment

Building Maintainable Software, Java Edition Pragmatic Bookshelf

How often do you hear people say things like this? "Our JavaScript is a mess, but we’re thinking about using [framework of the month]." Like it or not, JavaScript is not going away. No matter what framework or ”compiles-to-js” language or library you use, bugs and performance concerns will always be an issue if the underlying quality of your JavaScript is poor. Rewrites, including porting to the framework of the month, are terribly expensive and unpredictable. The bugs won’t magically go away, and can happily reproduce themselves in a new context. To complicate things further, features will get dropped, at least temporarily. The other popular method of fixing your JS is playing “JavaScript Jenga,” where each developer slowly and carefully takes their best guess at how the out-of-control system can be altered to allow for new features, hoping that this doesn’t bring the whole stack of blocks down. This book provides clear guidance on how best to avoid these pathological approaches to writing JavaScript:

Recognize you have a problem with your JavaScript quality. Forgive the code you have now, and the developers who made it. Learn repeatable, memorable, and time-saving refactoring techniques. Apply these techniques as you work, fixing things along the way. Internalize these techniques, and avoid writing as much problematic code to begin with. Bad code doesn't have to stay that way. And making it better doesn't have to be intimidating or unreasonably expensive. Develop reliable, maintainable, and robust JavaScript Pragmatic Bookshelf Summary Functional Programming in JavaScript teaches JavaScript developers functional techniques that will improve extensibility, modularity, reusability, testability, and performance. Through concrete examples and jargon-free explanations, this book teaches you how to apply functional programming to real-life development tasks Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology In complex web applications, the low-level details of your JavaScript code can obscure the workings of the system as a whole. As a coding style, functional programming (FP) promotes loosely coupled relationships among the components of your application, making the big picture easier to design, communicate, and maintain. About the Book Functional Programming in JavaScript teaches you techniques to improve your web applications - their extensibility, modularity, reusability, and testability, as well as their performance. This easy-to-read book uses concrete examples and clear explanations to show you how to use functional programming in real life. If you're new to functional programming, you'll appreciate this guide's many insightful comparisons to imperative or object-oriented programming that help you understand functional design. By the end, you'll think about application design in a fresh new way, and you may even grow to appreciate monads! What's Inside High-value FP techniques for real-world uses Using FP where it makes the most sense Separating the logic of your system from implementation details FP-style error handling, testing, and debugging All code samples use JavaScript ES6 (ES 2015) About the Reader Written for developers with a solid grasp of JavaScript fundamentals and web application design. About the Author Luis Atencio is a software engineer and architect building enterprise applications in Java, PHP, and JavaScript. Table of Contents PART 1 THINK FUNCTIONALLY Becoming functional Higher-order JavaScript PART 2 GET FUNCTIONAL Few data structures, many operations Toward modular, reusable code Design patterns against complexity PART 3 ENHANCING YOUR FUNCTIONAL SKILLS Bulletproofing your code Functional optimizations Managing asynchronous events and data

Programming JavaScript Applications
Simon and Schuster
ECMAScript 6 represents the biggest update to the core of JavaScript in the history of the language. In Understanding ECMAScript 6, expert developer Nicholas C. Zakas provides a complete guide to the object types, syntax, and other exciting changes that ECMAScript 6 brings to JavaScript. Every chapter is packed with example code that works in any JavaScript environment so you'll be able to see new features in action. You'll learn: –How ECMAScript 6 class syntax relates to more familiar JavaScript concepts –What makes iterators and generators useful –How

arrow functions differ from regular functions –Ways to store data with sets, maps, and more –The power of inheritance –How to improve asynchronous programming with promises –How modules change the way you organize code Whether you're a web developer or a Node.js developer, you'll find Understanding ECMAScript 6 indispensable on your journey from ECMAScript 5 to ECMAScript 6.

Working Effectively with Legacy Code No Starch Press

"Writing readable code"--Cover.

Reflections on the Craft of Programming Packt Publishing Ltd

Have you ever felt frustrated working with someone else's code? Difficult-to-maintain source code is a big problem in software development today, leading to costly delays and defects. Be part of the solution. With this practical book, you'll learn 10 easy-to-follow guidelines for delivering Java software that's easy to maintain and adapt. These guidelines have been derived from analyzing hundreds of real-world systems. Written by consultants from the Software Improvement Group (SIG), this book provides clear and concise explanations, with advice for turning the guidelines into practice. Examples for this edition are written in Java, while our companion C# book provides workable examples in that language. Write short units of code: limit the length of methods and constructors Write simple units of code: limit the number of branch points per method Write code once, rather than risk copying buggy code Keep unit interfaces small by extracting parameters into objects Separate concerns to avoid building large classes Couple architecture components loosely Balance the number and size of top-level components in your code Keep your codebase as small as possible Automate tests for your codebase Write clean code, avoiding "code smells" that indicate deeper problems

Related with Maintainable Javascript Writing Readable Code Ebook:
© Maintainable Javascript Writing Readable Code Ebook How Do Phospholipids Interact In An Aqueous Solution
© Maintainable Javascript Writing Readable Code Ebook How Are Economic Decisions Made In Traditional Economies
© Maintainable Javascript Writing Readable Code Ebook How Can Understanding Economics Make You A Better Citizen