Reliable JavaScript. How to Code Safely in the World's Most Dangerous Language

Description:
Test first, then code, for more robust large-scale JavaScript applications

Experienced developers still struggle with the failure of large-scale JavaScript applications that collapse under their own weight. Reliable JavaScript provides the solution in the form of a test-driven workflow that produces robust applications that stand the test of time. This book uses unit tests to explain, demonstrate, and employ a wide variety of JavaScript patterns and architectures. As you follow along, you will become an expert in the test-driven development of JavaScript that is suitable for commercial software. If you are ready to bring a high level of software engineering to your JavaScript development, this book is for you.

Reliable JavaScript:
- Shows you how to write code that's verified correct the first time, and has the test coverage to remain correct throughout subsequent maintenance
- Surveys the current landscape of JavaScript frameworks for testing, dependency injection, and aspect-oriented programming
- Explains testability characteristics and how to construct objects to be testable
- Demonstrates the implementation and use of common design patterns and advanced JavaScript features and architectures
- Covers DOM access testing, including excessive updates, excessive access in loops, and XHR/JSONP requests
- Provides all of the book's code both in the text and on the companion website

Wrox Professional guides are planned and written by working programmers to meet the real-world needs of programmers, developers, and IT professionals. Focused and relevant, they address the issues technology professionals face every day. They provide examples, practical solutions, and expert education in new technologies, all designed to help programmers do a better job.

Contents:
INTRODUCTION xxi

PART I: LAYING A SOLID FOUNDATION

CHAPTER 1: PRACTICING SKILLFUL SOFTWARE 3
Writing Code That Starts Correct 4
Mastering the Features of JavaScript 4
Case Study: D3.js 5
JavaScript Is Single-Threaded 15
Avoiding JavaScript Pitfalls in Larger Systems 16
Scripts Are Not Modules 16
Nested Functions Control Scope 16
Coding by Contract 17
Applying the Principles of Software Engineering 18
The SOLID Principles 18
Making Your Code More Reliable with AOP 60
Case Study: Building the Aop.js Module 60
Other AOP Libraries 72
AspectJ 72
AopJs jQuery Plugin 73
YUI s Do Class 73
Conclusion 73
Using a Code-Checking Tool 73
Making Your Code More Reliable with Linting Tools 74
Introducing JSHint 76
Using JSHint 76
If You Don’t Run It, Bugs Will Come 79
Alternatives to JSHint 79
JSLint 79
ESLint 79
Strict Mode 80
Summary 80
CHAPTER 3: CONSTRUCTING RELIABLE OBJECTS 81
Using Primitives 81
Using Object Literals 83
Using the Module Pattern 84
Creating Modules Jat JWill 84
Creating Immediate JExecution Modules 85
Creating Reliable Modules 86
Using Object Prototypes and Prototypal Inheritance 87
The Default Object Prototype 87
Prototypal Inheritance 88
Prototype Chains 88
Creating Objects with New 89
The new Object Creation Pattern 89
Potential for Bad Things to Happen 90
Creating an Aspect to Enforce a Contract on a Returned (Created) Object

Summary

CHAPTER 17: ENSURING CORRECT ARGUMENT TYPES

Introduction

Understanding the Opportunities and Risks Posed by JavaScript’s Type–Free Parameters

Extending the ContractRegistry to Check Arguments

Scoping Out the Task

Determining Whether Every Variable in a Set Fulfill Its Contract

Asserting That Every Variable in a Set Fulfill Its Contract

Packaging Argument–Checking in an Aspect

Supporting Contract Libraries

Putting It All Together

Creating the Contracts Modules

Creating the Application’s ContractRegistry

Bypassing Contracts for Production

Comparing the Aspect–Oriented Solution to a Static Solution

Considering the Advantages of TypeScript

Considering the Advantages of Aspects

Summary

CHAPTER 18: ENSURING CORRECT USE OF CALL, APPLY, AND BIND

Exploring How this Is Bound

Default Binding

Default Binding and strict Mode

Implicit Binding

new Binding

Explicit Binding

Creating and Testing Code That Uses call, apply, and bind

Using call and apply

Creating an Array.prototype.forEach Polyfill Using Test–Driven Development

Using bind

Summary
Summary 410

PART IV: SPECIAL SUBJECTS IN TESTING

CHAPTER 22: TESTING ADVANCED PROGRAM ARCHITECTURES 383

Ensuring Reliable Use of the Observer Pattern 384
Examining the Observer Pattern 384
Enhancing the Reliability of the Observer Pattern 391
Ensuring Reliable Use of the Mediator Pattern 395
Examining the Mediator Pattern 396
Enhancing the Reliability of Mediator-Based Code 397
Developing a Colleague 398
Testing a Colleague 399
Segregating the Mediator’s Interfaces 402
Deciding Where to Put the Contracts 403
Ensuring the Colleague Gets a Mediator with the Expected Interface 404
Developing a Mediator 406
Testing the Mediator 408
Summary 410

CHAPTER 23: ENSURING CONFORMANCE TO STANDARDS 435

Using ESLint 436
Installing ESLint 436
Installing Node and npm 436
Installing ESLint Using npm 439
Running ESLint 439
Executing ESLint on a Single File 442
Executing ESLint on All the JavaScript Files in a Directory 443
Enforcing Coding Standards with ESLint 444
Creating a Custom ESLint Rule 445
Running ESLint with Custom Rules 448
Enforcing Architectural Divisions 449
The Family-Secret Technique 450
The Imprinting Technique 452
The Mission Impossible Technique 454
The Magic Wand Technique 459
Do Not Use the Call Stack Technique 460
Other Techniques 460
Other Architectures 460
Summary 460

PART V: SUMMARY

CHAPTER 24: SUMMARY OF THE PRINCIPLES OF TEST-DRIVEN DEVELOPMENT 465
Recalling Why Test-Driven Development Is Worthwhile 465
Practicing Test-Driven Development 466
Writing Unit-Testable Code 466
Mastering the Mechanics of Test-Driven Development 466
Writing the Test Before the Code 467
Keeping Your Tests DRY 467
Testing Error Conditions First 467
Testing the Simple before the Complex 467
Being Specific 467
Testing Just One Thing 468
Your Test Data Are Just As Important As the Test 468
Using Jasmine Effectively 468
Testing the Patterns in This Book 468
Testing Aspect-Oriented Programming 468
Testing Object Construction 469
Testing Callbacks 469
Testing Promise-Based Code 469
Testing a Partial Function Application 470
Testing Memoization 470
Testing a Singleton 470
Testing a Factory Method 470
Testing a Sandbox 470
Testing the Decorator Pattern 471
Testing the Strategy Pattern 471
Ordering:

Order Online - [http://www.researchandmarkets.com/reports/3148631/](http://www.researchandmarkets.com/reports/3148631/)

Order by Fax - using the form below

Order by Post - print the order form below and send to

Research and Markets,
Guinness Centre,
Taylors Lane,
Dublin 8,
Ireland.
Fax Order Form
To place an order via fax simply print this form, fill in the information below and fax the completed form to 646-607-1907 (from USA) or +353-1-481-1716 (from Rest of World). If you have any questions please visit http://www.researchandmarkets.com/contact/

Order Information
Please verify that the product information is correct.

Product Name: Reliable JavaScript. How to Code Safely in the World’s Most Dangerous Language
Web Address: http://www.researchandmarkets.com/reports/3148631/
Office Code: SC

Product Format
Please select the product format and quantity you require:

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Hard Copy (Paper back)</th>
<th>USD 96 + USD 28 Shipping/Handling</th>
</tr>
</thead>
</table>

* Shipping/Handling is only charged once per order.
* The price quoted above is only valid for 30 days. Please submit your order within that time frame to avail of this price as all prices are subject to change.

Contact Information
Please enter all the information below in **BLOCK CAPITALS**

Title:  Mr  □  Mrs  □  Dr  □  Miss  □  Ms  □  Prof  □
First Name: __________________________  Last Name: __________________________
Email Address: * __________________________
Job Title: __________________________
Organisation: __________________________
Address: __________________________
City: __________________________
Postal / Zip Code: __________________________
Country: __________________________
Phone Number: __________________________
Fax Number: __________________________

* Please refrain from using free email accounts when ordering (e.g. Yahoo, Hotmail, AOL)
Payment Information

Please indicate the payment method you would like to use by selecting the appropriate box.

☐ Pay by credit card: You will receive an email with a link to a secure webpage to enter your credit card details.

☐ Pay by check: Please post the check, accompanied by this form, to:

Research and Markets, Guinness Center, Taylors Lane, Dublin 8, Ireland.

☐ Pay by wire transfer: Please transfer funds to:

<table>
<thead>
<tr>
<th>Description</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account number</td>
<td>833 130 83</td>
</tr>
<tr>
<td>Sort code</td>
<td>98-53-30</td>
</tr>
<tr>
<td>Swift code</td>
<td>ULSBIE2D</td>
</tr>
<tr>
<td>IBAN number</td>
<td>IE78ULSB98533083313083</td>
</tr>
<tr>
<td>Bank Address</td>
<td>Ulster Bank, 27-35 Main Street, Blackrock, Co. Dublin, Ireland.</td>
</tr>
</tbody>
</table>

If you have a Marketing Code please enter it below:

Marketing Code: ____________________________

Please note that by ordering from Research and Markets you are agreeing to our Terms and Conditions at http://www.researchandmarkets.com/info/terms.asp

Please fax this form to:

(646) 607-1907 or (646) 964-6609 - From USA
+353-1-481-1716 or +353-1-653-1571 - From Rest of World