



C o m m u n i t y E x p e r i e n c e D i s t i l l e d

Functional Programming in JavaScript

Unlock the powers of functional programming hidden within
JavaScript to build smarter, cleaner, and more reliable web apps

Dan Mantyla

www.it-ebooks.info

[PACKT] open source*
PUBLISHING

community experience distilled

Functional Programming in JavaScript

Unlock the powers of functional programming hidden within JavaScript to build smarter, cleaner, and more reliable web apps

Dan Mantyla



BIRMINGHAM - MUMBAI

Functional Programming in JavaScript

Copyright © 2015 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author nor Packt Publishing, and its dealers and distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

First published: March 2015

Production reference: 1230315

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham B3 2PB, UK

ISBN 978-1-78439-822-4

www.packtpub.com

Cover Image by Dan Mantyla

Credits

Author

Dan Mantyla

Project Coordinator

Nidhi Joshi

Reviewers

Dom Derrien

Joe Dorocak

Peter Ehrlich

Edward E. Griebel Jr.

Proofreaders

Stephen Copestake

Maria Gould

Paul Hindle

Commissioning Editor

Julian Ursell

Indexer

Tejal Daruwale Soni

Acquisition Editor

Owen Roberts

Production Coordinator

Aparna Bhagat

Content Development Editor

Kirti Patil

Cover Work

Aparna Bhagat

Technical Editor

Abhishek R. Kotian

Copy Editors

Aditya Nair

Aarti Saldanha

Vikrant Phadkey

About the Author

Dan Mantyla works as a web application developer for the University of Kansas. He enjoys contributing to open source web frameworks and wrenching on motorcycles. Dan is currently living in Lawrence, Kansas, USA – the birthplace of Python Django and home to Linux News Media.

Dan has also clicked the cover image, which was taken outside his home in Lawrence, Kansas, USA, where the sunflower fields are in bloom for only one short week in September.

About the Reviewers

Dom Derrien is a full stack web developer who has recently been defining application environments with a focus on high availability and scalability. He's been in the development field for more than 15 years and has worked for big and small companies and as an entrepreneur.

He's currently working for the game company Ubisoft, where he defines the next generation services platform for its successful AAA games. To extend the gamer experience on to the Web and on mobiles, he provides technical means that are transparent, efficient, and highly flexible.

Having developed smart clients before the introduction of XHR, using a frameset tag to keep the context and a hidden frame of size=0 to dynamically exchange data with servers, he had a great pleasure of reviewing this book, which pushes the language to its limits. He hopes that it will help developers improve their programming skills.

I want to thank my wife, Sophie, and our sons, Erwan and Goulven, with whom I enjoy a peaceful life in Montréal, Québec, Canada.

Joe Dorocak, whose Internet moniker is Joe Codeswell, is a very experienced programmer. He enjoys creating readable code that implements project requirements efficiently and in a manner that can be easily understood. He considers writing code akin to writing poetry.

Joe prides himself on the ability to communicate clearly and professionally. He considers his code to be communication, not only with the machine platforms on which it runs, but also with human programmers who might read it in the future.

Joe has worked as an employee as well as in a contractual role for major brands such as IBM, HP, GTE/Sprint, and other top-shelf companies. He is presently consulting on web, mobile, and desktop applications, which are coded primarily, but not exclusively, in Python and JavaScript. For more details about him, please visit <https://www.linkedin.com/in/joedorocak>.

Peter Ehrlich taught himself web programming in 2007, and now works on performance JavaScript and WebGL at Leap Motion, Inc. In his spare time, he enjoys dancing, rock climbing, and taking naps.

Edward E. Griebel Jr. has been developing enterprise software for over 20 years in C, C++, and Java. He has a bachelor of science degree in computer engineering. He is currently a middleware architect at a leading payroll and financial services provider in the U.S., focusing on systems integration and UI and server development.

www.PacktPub.com

Support files, eBooks, discount offers and more

For support files and downloads related to your book, please visit www.PacktPub.com.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at www.PacktPub.com and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at service@packtpub.com for more details.

At www.PacktPub.com, you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.



<https://www2.packtpub.com/books/subscription/packtlib>

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can access, read and search across Packt's entire library of books.

Why Subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print and bookmark content
- On demand and accessible via web browser

Free Access for Packt account holders

If you have an account with Packt at www.PacktPub.com, you can use this to access PacktLib today and view nine entirely free books. Simply use your login credentials for immediate access.

Table of Contents

Preface	v
Chapter 1: The Powers of JavaScript's Functional Side – a Demonstration	1
Introduction	1
The demonstration	2
The application – an e-commerce website	2
Imperative methods	2
Functional programming	4
Summary	7
Chapter 2: Fundamentals of Functional Programming	9
Functional programming languages	9
What makes a language functional?	10
Advantages	11
Cleaner code	11
Modularity	12
Reusability	12
Reduced coupling	12
Mathematically correct	12
Functional programming in a nonfunctional world	14
Is JavaScript a functional programming language?	15
Working with functions	17
Self-invoking functions and closures	18
Higher-order functions	19
Pure functions	20
Anonymous functions	21
Method chains	23
Recursion	24
Divide and conquer	25
Lazy evaluation	26

Table of Contents

The functional programmer's toolkit	27
Callbacks	28
Array.prototype.map()	29
Array.prototype.filter()	30
Array.prototype.reduce()	31
Honorable mentions	32
Array.prototype.forEach	32
Array.prototype.concat	33
Array.prototype.reverse	34
Array.prototype.sort	34
Array.prototype.every and Array.prototype.some	35
Summary	35
Chapter 3: Setting Up the Functional Programming Environment	37
Introduction	37
Functional libraries for JavaScript	38
Underscore.js	38
Fantasy Land	41
Bilby.js	42
Lazy.js	44
Bacon.js	45
Honorable mentions	46
Development and production environments	48
Browsers	48
Server-side JavaScript	49
A functional use case in the server-side environment	49
CLI	50
Using functional libraries with other JavaScript modules	50
Functional languages that compile into JavaScript	51
Summary	52
Chapter 4: Implementing Functional Programming Techniques in JavaScript	53
Partial function application and currying	54
Function manipulation	54
Apply, call, and the this keyword	54
Binding arguments	55
Function factories	56
Partial application	57
Partial application from the left	58
Partial application from the right	59
Currying	60

Function composition	62
Compose	62
Sequence – compose in reverse	63
Compositions versus chains	64
Programming with compose	65
Mostly functional programming	68
Handling events	70
Functional reactive programming	71
Reactivity	72
Putting it all together	73
Summary	75
Chapter 5: Category Theory	77
Category theory	78
Category theory in a nutshell	78
Type safety	80
Object identities	82
Functors	83
Creating functors	84
Arrays and functors	84
Function compositions, revisited	85
Monads	87
Maybes	88
Promises	90
Lenses	92
jQuery is a monad	94
Implementing categories	95
Summary	98
Chapter 6: Advanced Topics and Pitfalls in JavaScript	99
Recursion	100
Tail recursion	100
The Tail-call elimination	101
Trampolining	103
The Y-combinator	106
Memoization	108
Variable scope	109
Scope resolutions	109
Global scope	110
Local scope	110
Object properties	111
Closures	112
Gotchas	113

Table of Contents

Function declarations versus function expressions versus the function constructor	114
Function declarations	114
Function expressions	115
The function constructor	115
Unpredictable behavior	116
Summary	117
Chapter 7: Functional and Object-oriented Programming in JavaScript	119
JavaScript – the multi-paradigm language	120
JavaScript's object-oriented implementation – using prototypes	121
Inheritance	121
JavaScript's prototype chain	122
Inheritance in JavaScript and the Object.create() method	123
Mixing functional and object-oriented programming in JavaScript	125
Functional inheritance	125
Strategy Pattern	126
Mixins	128
Classical mixins	129
Functional mixins	130
Summary	133
Appendix A: Common Functions for Functional Programming in JavaScript	135
Appendix B: Glossary of Terms	143
Index	147

Preface

Functional programming is a style that emphasizes and enables the writing of smarter code, which minimizes complexity and increases modularity. It's a way of writing cleaner code through clever ways of mutating, combining, and using functions. JavaScript provides an excellent medium for this approach. JavaScript, the Internet's scripting language, is actually a functional language at heart. By learning how to expose its true identity as a functional language, we can implement web applications that are powerful, easier to maintain, and more reliable. By doing this, JavaScript's odd quirks and pitfalls will suddenly become clear and the language as a whole will make infinitely more sense. Learning how to use functional programming will make you a better programmer for life.

This book is a guide for both new and experienced JavaScript developers who are interested in learning functional programming. With a focus on the progression of functional programming techniques, styles, and detailed information about JavaScript libraries, this book will help you to write smarter code and become a better programmer.

What this book covers

Chapter 1, The Powers of JavaScript's Functional Side – a Demonstration, sets the pace of the book by creating a small web application with the help of both traditional methods and functional programming. It then compares these two methods to underline the importance of functional programming.

Chapter 2, Fundamentals of Functional Programming, introduces you to the core concepts of functional programming as well as built-in JavaScript functions.

Chapter 3, Setting Up the Functional Programming Environment, explores different JavaScript libraries and how they can be optimized for functional programming.

Chapter 4, Implementing Functional Programming Techniques in JavaScript, explains the functional paradigm in JavaScript. It covers several styles of functional programming and demonstrates how they can be employed in different scenarios.

Chapter 5, Category Theory, explains the concept of Category Theory in detail and then implements it in JavaScript.

Chapter 6, Advanced Topics and Pitfalls in JavaScript, highlights various drawbacks you may face while programming in JavaScript, and the various ways to successfully deal with them.

Chapter 7, Functional and Object-oriented Programming in JavaScript, relates both functional and object-oriented programming to JavaScript, and shows you how the two paradigms can complement each other and coexist side by side.

Appendix A, Common Functions for Functional Programming in JavaScript, contains common functions used to perform functional programming in JavaScript.

Appendix B, Glossary of Terms, includes a glossary of terms used throughout the book.

What you need for this book

Only a browser is needed to get you up and running.

Who this book is for

If you are a JavaScript developer interested in learning functional programming, looking for a quantum leap toward mastering the JavaScript language, or just want to become a better programmer in general, then this book is ideal for you. This guide is aimed at programmers involved in developing reactive frontend applications, server-side applications that wrangle with reliability and concurrency, and everything else in between.

Conventions

In this book, you will find a number of text styles that distinguish between different kinds of information. Here are some examples of these styles and an explanation of their meaning.

Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows:
"We can include other contexts through the use of the `include` directive."

A block of code is set as follows:

```
Function.prototype.partialApply = function() {  
    var func = this;  
    args = Array.prototype.slice.call(arguments);  
    return function() {  
        return func.apply(this, args.concat(  
            Array.prototype.slice.call(arguments)  
        ));  
    };  
};
```

When we wish to draw your attention to a particular part of a code block, the relevant lines or items are set in bold:

```
var messages = ['Hi', 'Hello', 'Sup', 'Hey', 'Hola'];  
messages.map(function(s, i){  
    return printSomewhere(s, i*10, i*10);  
}).forEach(document.body.appendChild);
```

New terms and **important words** are shown in bold. Words that you see on the screen, for example, in menus or dialog boxes, appear in the text like this: "Clicking the **Next** button moves you to the next screen."



Warnings or important notes appear in a box like this.



Tips and tricks appear like this.

Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book – what you liked or disliked. Reader feedback is important for us as it helps us develop titles that you will really get the most out of.

To send us general feedback, simply e-mail feedback@packtpub.com, and mention the book's title in the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide at www.packtpub.com/authors.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Downloading the example code

You can download the example code files from your account at <http://www.packtpub.com> for all the Packt Publishing books you have purchased. If you purchased this book elsewhere, you can visit <http://www.packtpub.com/support> and register to have the files e-mailed directly to you.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books – maybe a mistake in the text or the code – we would be grateful if you could report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting <http://www.packtpub.com/submit-errata>, selecting your book, clicking on the **Errata Submission Form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded to our website or added to any list of existing errata under the Errata section of that title.

To view the previously submitted errata, go to <https://www.packtpub.com/books/content/support> and enter the name of the book in the search field. The required information will appear under the **Errata** section.

Piracy

Piracy of copyrighted material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works in any form on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at copyright@packtpub.com with a link to the suspected pirated material.

We appreciate your help in protecting our authors and our ability to bring you valuable content.

Questions

If you have a problem with any aspect of this book, you can contact us at questions@packtpub.com, and we will do our best to address the problem.

1

The Powers of JavaScript's Functional Side – a Demonstration

Introduction

For decades, functional programming has been the darling of computer science aficionados, prized for its mathematical purity and puzzling nature that kept it hidden in dusty computer labs occupied by data scientists and PhD hopefuls. But now, it is going through a resurgence, thanks to modern languages such as **Python**, **Julia**, **Ruby**, **Clojure** and – last but not least – **JavaScript**.

JavaScript, you say? The web's scripting language? Yes!

JavaScript has proven to be an important technology that isn't going away for quite a while. This is largely due to the fact that it is capable of being reborn and extended with new frameworks and libraries, such as **backbone.js**, **jQuery**, **Dojo**, **underscore.js**, and many more. *This is directly related to JavaScript's true identity as a functional programming language.* An understanding of functional programming with JavaScript will be welcome and useful for a long time for programmers of any skill level.

Why so? Functional programming is very powerful, robust, and elegant. It is useful and efficient on large data structures. It can be very advantageous to use JavaScript – a client-side scripting language, as a functional means to manipulate the DOM, sort API responses or perform other tasks on increasingly complex websites.