O'REILLY®

"Kyle's way of critically thinking about every tiny bit of the language will creep into your mindset and general workflow."
—**SHANE HUDSON**, freelance frontend website developer

KYLE SIMPSON

SCOPE & CLOSURES

YOU DON'T KNOW JS

YOU DON'T KNOW JS

SCOPE & CLOSURES

No matter how much experience you have with JavaScript, odds are you don't fully understand the language. This concise yet in-depth guide takes you inside scope and closures, two core concepts you need to know to become a more efficient and effective JavaScript programmer. You'll learn how and why they work, and how an understanding of closures can be a powerful part of your development skill set.

Like other books in the ***You Don't Know JS*** series, ***Scope and Closures*** dives into trickier parts of the language that many JavaScript programmers simply avoid. Armed with this knowledge, you can achieve true JavaScript mastery.

- **Learn about scope, a set of rules to help JavaScript engines locate variables in your code**
- **Go deeper into nested scope, a series of containers for variables and functions**
- **Explore function- and block-based scope, "hoisting," and the patterns and benefits of scope-based hiding**
- **Discover how to use closures for synchronous and asynchronous tasks, including the creation of JavaScript libraries**

KYLE SIMPSON is an Open Web Evangelist who's passionate about all things JavaScript. He's an author, workshop trainer, tech speaker, and OSS contributor/leader.

JAVASCRIPT

US $17.99 **CAN $20.99**
ISBN: 978-1-449-33558-8

9 781449 335588 51799

O'REILLY®
oreilly.com
YouDontKnowJS.com

Scope and Closures

Kyle Simpson

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

O'REILLY®

Scope and Closures

by Kyle Simpson

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

Editors: Simon St. Laurent and Brian MacDonald
Production Editor: Melanie Yarbrough
Proofreader: Linley Dolby
Cover Designer: Karen Montgomery
Interior Designer: David Futato
Illustrator: Rebecca Demarest

March 2014: First Edition

Revision History for the First Edition:

2014-03-06: First release

2015-05-01: Second release

See *http://oreilly.com/catalog/errata.csp?isbn=9781449335588* for release details.

ISBN: 978-1-449-33558-8

[LSI]

Table of Contents

Foreword

When I was a young child, I would often enjoy taking things apart and putting them back together again—old mobile phones, hi-fi stereos, and anything else I could get my hands on. I was too young to really use these devices, but whenever one broke, I would instantly ask if I could figure out how it worked.

I remember once looking at a circuit board for an old radio. It had this weird long tube with copper wire wrapped around it. I couldn't work out its purpose, but I immediately went into research mode. What does it do? Why is it in a radio? It doesn't look like the other parts of the circuit board, why? Why does it have copper wrapped around it? What happens if I remove the copper?! Now I know it was a loop antenna, made by wrapping copper wire around a ferrite rod, which are often used in transistor radios.

Did you ever become addicted to figuring out all of the answers to every *why* question? Most children do. In fact it is probably my favorite thing about children—their desire to learn.

Unfortunately, now I'm considered a *professional* and spend my days making things. When I was young, I loved the idea of one day making the things that I took apart. Of course, most things I make now are with JavaScript and not ferrite rods…but close enough! However, despite once loving the idea of making things, I now find myself longing for the desire to figure things out. Sure, I often figure out the best way to solve a problem or fix a bug, but I rarely take the time to question my tools.

And that is exactly why I am so excited about this "You Don't Know JS" series of books. Because it's right. I don't know JS. I use JavaScript day in, day out and have done for many years, but do I really understand it? No. Sure, I understand a lot of it and I often read the specs and the mailing lists, but no, I don't understand as much as my inner six-year-old wishes I did.

Scope and Closures is a brilliant start to the series. It is very well targeted at people like me (and hopefully you, too). It doesn't teach JavaScript as if you've never used it, but it does make you realize how little about the inner workings you probably know. It is also coming out at the perfect time: ES6 is finally settling down and implementation across browsers is going well. If you've not yet made time for learning the new features (such as `let` and `const`), this book will be a great introduction.

So I hope that you enjoy this book, but moreso, that Kyle's way of critically thinking about how every tiny bit of the language works will creep into your mindset and general workflow. Instead of just using the antenna, figure out how and why it works.

—Shane Hudson
www.shanehudson.net

Preface

I'm sure you noticed, but "JS" in the book series title is not an abbreviation for words used to curse about JavaScript, though cursing at the language's quirks is something we can probably all identify with!

From the earliest days of the Web, JavaScript has been a foundational technology that drives interactive experience around the content we consume. While flickering mouse trails and annoying pop-up prompts may be where JavaScript started, nearly two decades later, the technology and capability of JavaScript has grown many orders of magnitude, and few doubt its importance at the heart of the world's most widely available software platform: the Web.

But as a language, it has perpetually been a target for a great deal of criticism, owing partly to its heritage but even more to its design philosophy. Even the name evokes, as Brendan Eich once put it, "dumb kid brother" status next to its more mature older brother, Java. But the name is merely an accident of politics and marketing. The two languages are vastly different in many important ways. "JavaScript" is as related to "Java" as "Carnival" is to "Car."

Because JavaScript borrows concepts and syntax idioms from several languages, including proud C-style procedural roots as well as subtle, less obvious Scheme/Lisp-style functional roots, it is exceedingly approachable to a broad audience of developers, even those with just little to no programming experience. The "Hello World" of JavaScript is so simple that the language is inviting and easy to get comfortable with in early exposure.

While JavaScript is perhaps one of the easiest languages to get up and running with, its eccentricities make solid mastery of the language a vastly less common occurrence than in many other languages. Where it takes a pretty in-depth knowledge of a language like C or C++ to write a full-scale program, full-scale production JavaScript can, and often does, barely scratch the surface of what the language can do.

Sophisticated concepts that are deeply rooted into the language tend instead to surface themselves in *seemingly* simplistic ways, such as passing around functions as callbacks, which encourages the JavaScript developer to just use the language as-is and not worry too much about what's going on under the hood.

It is simultaneously a simple, easy-to-use language that has broad appeal and a complex and nuanced collection of language mechanics that without careful study will elude *true understanding* even for the most seasoned of JavaScript developers.

Therein lies the paradox of JavaScript, the Achilles' heel of the language, the challenge we are presently addressing. Because JavaScript *can* be used without understanding, the understanding of the language is often never attained.

Mission

If at every point that you encounter a surprise or frustration in JavaScript, your response is to add it to the blacklist, as some are accustomed to doing, you soon will be relegated to a hollow shell of the richness of JavaScript.

While this subset has been famously dubbed "The Good Parts," I would implore you, dear reader, to instead consider it the "The Easy Parts," "The Safe Parts," or even "The Incomplete Parts."

This "You Don't Know JavaScript" book series offers a contrary challenge: learn and deeply understand *all* of JavaScript, even and especially "The Tough Parts."

Here, we address head on the tendency of JS developers to learn "just enough" to get by, without ever forcing themselves to learn exactly how and why the language behaves the way it does. Furthermore, we eschew the common advice to *retreat* when the road gets rough.

I am not content, nor should you be, at stopping once something *just works*, and not really knowing *why*. I gently challenge you to journey down that bumpy "road less traveled" and embrace all that JavaScript is and can do. With that knowledge, no technique, no framework, no