You Don't Know JS

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 2 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 which 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 which 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 popular buzzword acronym of the week, will be beyond your understanding.

These books each take on specific core parts of the language which are most commonly misunderstood or under-understood, and dive very deep and exhaustively into them. You should come away from reading with a firm confidence in your understanding, not just of the theoretical, but the practical "what you need to know" bits.

The JavaScript you know *right now* is probably *parts* handed down to you by others who've been burned by incomplete understanding. *That* JavaScript is but a shadow of the true language. You don't *really* know JavaScript, *yet*, but if you dig into this series, you *will*. Read on, my friends. JavaScript awaits you.

Summary

JavaScript is awesome. It's easy to learn partially, and much harder to learn completely (or even *sufficiently*). When developers encounter confusion, they usually blame the language instead of their lack of understanding. These books aim to fix that, inspiring a strong appreciation for the language you can now, and *should*, deeply *know*.

Note: Many of the examples in this book assume modern (and future-reaching) JavaScript engine environments, such as ES6. Some code may not work as described if run in older (pre-ES6) engines.