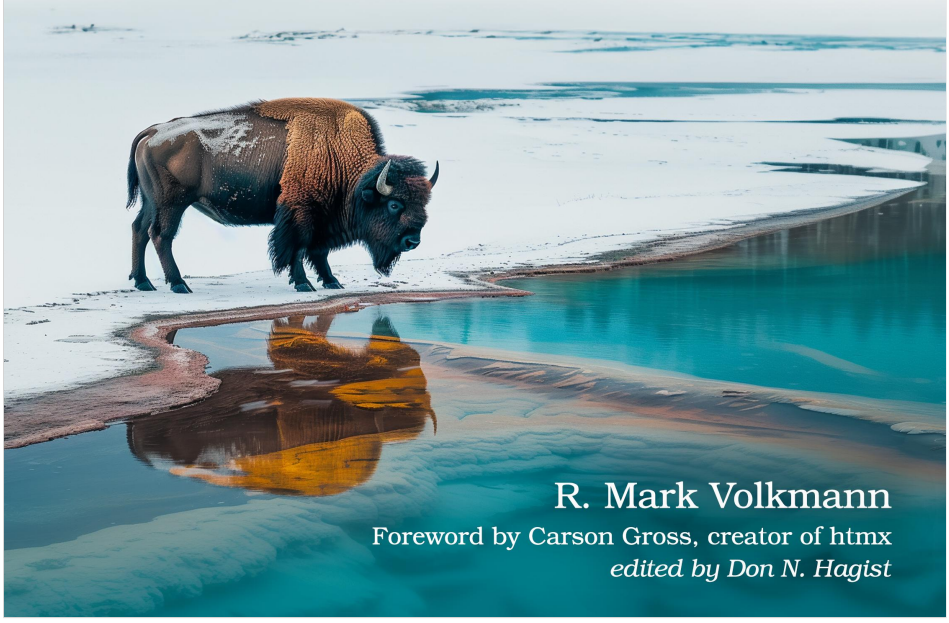
The Pragmatic Programmers

Pragmatic exPress

Server-Driven Web Apps with htmx

Any Language, Less Code, Simpler Code



R. Mark Volkmann

Foreword by Carson Gross, creator of htmx

edited by Don N. Hagist

This extract shows the online version of this title, and may contain features (such as hyperlinks and colors) that are not available in the print version.

For more information, or to purchase a paperback or ebook copy, please visit https://www.pragprog.com.

Preface

Modern web development has become overly complicated. Popular frameworks have somewhat steep learning curves and often perform more work than necessary to achieve a desired result.

I have firsthand experience with many web development approaches including vanilla JavaScript, jQuery, AngularJS, Angular, React, Vue, and Svelte. For me, each of these provided improvements over what came before. But these were incremental improvements.

I find htmx to be very different from these frameworks and libraries. It's a breath of fresh air that I'm excited to share with you! Let's discover how htmx simplifies web development, resulting in applications that are easier to understand and require less code.

Modern web frameworks for implementing single-page applications (SPAs) frequently encourage the following steps:

- The browser downloads somewhat large amounts of JavaScript code.
- User interaction triggers sending an HTTP request to a server endpoint.
- The endpoint queries a database.
- Data from the database is converted to JSON.
- The endpoint returns a JSON response.
- JavaScript running in the browser parses the JSON into a JavaScript object.
- The framework generates HTML from the JavaScript object and inserts it into the DOM.

HyperText Markup eXtensions (*htmx*) is a client-side JavaScript library that simplifies this process.

Glossary of Web Application Terms

If you've forgotten some of these acronyms, here's a quick reminder:

- *HTTP*—Hypertext Transfer Protocol, the protocol used to send requests from a web browser to a server
- *JSON*—JavaScript Object Notation, a data format based on JavaScript objects
- *DOM*—Document Object Model, a tree of JavaScript objects that represent the structure of a document in a format such as HTML

With htmx, endpoints convert data to HTML (or plain text) rather than JSON, and that is returned to the browser. JavaScript in the browser no longer needs to parse JSON and generate HTML from it. It merely needs to insert the HTML into the DOM. A full-page refresh isn't necessary.

The htmx library is quite small—less than 17KB minified and compressed. Pages load faster due to downloading less JavaScript code than when using typical SPA frameworks. You can see these improvements with app metrics such as First Contentful Paint and Time to Interactive. Htmx applications also provide faster server interactions because the time spent generating and parsing JSON is eliminated.

The fact that htmx endpoints generate HTML means that htmx moves a large portion of web development from the client to the server.

Htmx keeps most of the application state on the server. State that's only of concern to the user interface, such as hiding and showing content, can remain on the client. But the client-only state is typically a small portion of the overall state.

Required Knowledge

Now that you understand some of the benefits of using htmx, let's discuss what you need to know to use it.

It's useful to have some knowledge of the following:

- A code editor such as VS Code or Vim
- HTML for specifying what will be rendered in the browser

- CSS for styling what is rendered
- A programming language for implementing HTTP endpoints
- HTTP basics such as verbs, requests, and responses
- Command-line basics such as changing the working directory and starting a local server

If you're not already a full-stack developer, using htmx will provide motivation to move in that direction. Front-end web developers will become comfortable with implementing server endpoints. Back-end developers will become comfortable with HTML and CSS.