

Extracted from:

# Build Reactive Websites with RxJS

## Master Observables and Wrangle Events

This PDF file contains pages extracted from *Build Reactive Websites with RxJS*, published by the Pragmatic Bookshelf. For more information or to purchase a paperback or PDF copy, please visit <http://www.pragprog.com>.

Note: This extract contains some colored text (particularly in code listing). This is available only in online versions of the books. The printed versions are black and white. Pagination might vary between the online and printed versions; the content is otherwise identical.

Copyright © 2018 The Pragmatic Programmers, LLC.

All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

The Pragmatic Bookshelf

Raleigh, North Carolina

The  
Pragmatic  
Programmers

# Build Reactive Websites with RxJS

Master Observables and Wrangle Events



Randall Koutnik  
*edited by Brian MacDonald*

# Build Reactive Websites with RxJS

Master Observables and Wrangle Events

Randall Koutnik

The Pragmatic Bookshelf

Raleigh, North Carolina



Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and The Pragmatic Programmers, LLC was aware of a trademark claim, the designations have been printed in initial capital letters or in all capitals. The Pragmatic Starter Kit, The Pragmatic Programmer, Pragmatic Programming, Pragmatic Bookshelf, PragProg and the linking *g* device are trademarks of The Pragmatic Programmers, LLC.

Every precaution was taken in the preparation of this book. However, the publisher assumes no responsibility for errors or omissions, or for damages that may result from the use of information (including program listings) contained herein.

Our Pragmatic books, screencasts, and audio books can help you and your team create better software and have more fun. Visit us at <https://pragprog.com>.

The team that produced this book includes:

Publisher: Andy Hunt

VP of Operations: Janet Furlow

Managing Editor: Susan Conant

Development Editor: Brian MacDonald

Copy Editor: Paula Robertson

Indexing: Potomac Indexing, LLC

Layout: Gilson Graphics

For sales, volume licensing, and support, please contact [support@pragprog.com](mailto:support@pragprog.com).

For international rights, please contact [rights@pragprog.com](mailto:rights@pragprog.com).

Copyright © 2018 The Pragmatic Programmers, LLC.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

ISBN-13: 978-1-68050-295-4

Book version: P1.0—December 2018

# Introduction

---

Hello! If you've picked up this book, you've probably spent some time developing websites. Whether you're an old hand at slinging JavaScript or a newcomer to the frontend world, this book has something for you.

## The Goal of This Book

The list of requirements for frontend work keeps increasing. You now need to build websites that load quickly on shaky connections, render perfectly on mobile devices, and respond with lightning speed to user input. All of these tasks require dealing with a high number of events from disparate sources, be it your CRM, late-breaking news, or just a chat room. The topic of this book, observables, is a new way of thinking about managing these events, even when they may occur sometime in the future. Observables are a neat concept, but what's important is that you can keep things straight in your head, allowing you to build bigger, faster, and less-buggy applications for your users.

It's important to ask, with such a big claim as “simplifying frontend development,” what exactly is simplified? While RxJS (short for “Reactive eXtensions to JavaScript”) brings simplicity to many areas, this book focuses on two areas that can have you reaching for the aspirin time and time again:

## Asynchronous Calls and Control Flow

JavaScript's async-first design has been both a blessing and a curse. While the event loop allows us to fire off AJAX calls with ease, keeping track of them all can be quite the chore. A single AJAX request can be modeled as a promise, but more than one suddenly means there's a cacophony of items to manually track (and even cancel) as the user progresses through our app. One of the most notorious examples, the typeahead, will be covered in [Advanced Async](#). You'll learn how to delegate both the calls and control flow to RxJS, allowing you to focus on the rest of your application.

## State Management

On the other hand, managing an application's state has been the bane of programmers since RAM was invented, leading to the oft-quoted advice to “turn it off and on again,” resetting the computer's state. JavaScript makes this worse by defaulting to a global, mutable state. In recent years, the JavaScript community has started to build some impressive solutions to this problem.

RxJS compartmentalizes your eventing flows, encapsulating each action in a single function. Building on top of this, RxJS also provides many helper operators that keep an internal state, allowing you to outsource your state worries to the library. In addition to these operators, you'll learn about `ngrx`, a state management library built on top of RxJS in [Advanced Angular](#). In [Reactive Game Development](#), you'll build out your own state system that's specific to the HTML5 Canvas API.



**Joe asks:**

### Why RxJS over Other Observable Libraries?

RxJS is just one of many different JavaScript libraries that implement the observable pattern. RxJS is one of the oldest, and as of this writing, has had six major releases. It's also one of the most popular observable tools, which means that there will be plenty of StackOverflow questions, blog posts, and yes, books about it, if you run into trouble. RxJS goes for the “batteries included” philosophy, so any conceivable method or pattern you need, RxJS will have it. It also includes tooling for you to create your own observables as needed. RxJS is built on TypeScript, allowing editors like Visual Studio Code to take advantage of type hinting to point out bugs as soon as possible. Finally, it's the backbone for the popular Angular framework, a topic we'll cover later on (though RxJS is by no means exclusive to Angular).

Other JavaScript libraries that implement the observable pattern include `mobx`, `BaconJS`, and `Kefir`. These either focus on a specific use case for observables (like `mobx` and state management) or prefer a limited feature set to focus on performance (though RxJS is very fast, especially the latest version).

## But I Already Use (Insert Framework Here)!

Don't worry! RxJS is not intended to replace your entire frontend toolset. Rather, it's been designed to integrate with whatever JavaScript framework you use, be it a tangled mess of jQuery and twine, or the latest framework that was just released this week. The Angular framework in particular has decided to integrate with RxJS out-of-the-box, and this book covers some of



those scenarios, but the knowledge contained in this book is by no means specific to Angular.

## How to Read This Book

This book intends to default to the practical. Everyone learns best by doing, and so each chapter is a series of projects for you to build, each one illustrating a new concept for you to learn. The projects start out small (the first one's just a stopwatch) but grow in complexity. Before you know it, you'll be building entire websites backed by the impressive power of RxJS. This is not a book to be read from cover to cover in one sitting; rather it's a companion, meant to be cracked open on a second monitor as you type away, building out one of the many examples.

If you're a complete newcomer to the world of observables, I'd recommend starting at the beginning of this book and working your way through. If you've subscribed to an observable or two, you may want to skip ahead to [Managing Asynchronous Events](#), though the interim chapters might be a good refresher.

This book is split into three sections, each building on concepts introduced in the previous sections.

### Section 1: Vanilla RxJS

This section starts off with an introduction to observables and how to create them in RxJS. It focuses on teaching the concepts behind RxJS, so that you're more familiar with what's going on. You'll learn the concepts and best practices behind observables while tackling ever-harder examples. This section is framework-agnostic, so it will be equally useful to you regardless of your preferred toolset.

### Section 2: RxJS in Angular

As incredible as RxJS is, it works best as a glue that holds the rest of the application together. An application built entirely of glue is not a good application. In this section, you'll dive into RxJS' use in Angular, building entire websites. You'll see how RxJS integrates into Angular to simplify tasks both complicated and repetitive. You'll also move beyond pure coding to see how RxJS use can enable us to build much more effective user experience. This section ends with a performance profiling task, where you'll use RxJS to update a page only when it's absolutely needed.

## Section 3: Reactive Game Development

In this final section, you'll construct your own game based on the Canvas API. Games have an incredible amount of things going on at any given moment, so RxJS is a perfect fit for organizing all of the hectic events that might be triggering. This is also the most advanced section, where you'll build your own reusable observable operator alongside a from-scratch state management tool.

## Setting Up the Environment

To use this book, you'll need the following (older versions of node/npm may work, but they haven't been tested):

- NodeJS version 10.1.0 or greater
- npm version 6.1.0 or greater
- The code found on the page for this book at The Pragmatic Bookshelf site<sup>1</sup>

Download the zipped code from this book's page and unzip it. You'll find four folders—three corresponding to one of the three sections in this book and an asset server.

- The vanilla folder contains the relevant code and scaffolding for the first five chapters that cover the RxJS library in depth.
- ng2 covers the next three chapters, which are an overview for using RxJS within the Angular framework.
- canvas contains the code and other related files to build your first RxJS-based game.

The asset-server folder contains a local server you'll use to serve assets throughout the book. Once you've unzipped the code, run `npm install` in the asset-server folder to ensure all dependencies are installed. When everything is installed, run `npm start` to start the server, which will be listening at <http://localhost:3000>. You'll use this server throughout the book.

Section-specific instructions for using the code provided is at the beginning of each section.

## Typescript

The code in this book, as well as all the examples, use TypeScript, a language that is a superset of JavaScript that adds optional types. No TypeScript

---

1. [https://pragprog.com/titles/rxjs/source\\_code](https://pragprog.com/titles/rxjs/source_code)



knowledge is needed to start this book. Everything you need to know will be explained along the way.

TypeScript can't be run directly, so a compile step is needed. The code in this book is configured to use SystemJS to compile your code behind the scenes, so you don't need to worry about it. You may want to install language support into your editor of choice, though this is not required.

## Why TypeScript?

In vanilla JavaScript, the following is perfectly valid syntax:

```
function canRegister(age) {
  return age > 13;
}

let user = {
  name: 'Jose',
  age: 23
};

canRegister(user);
```

This code prevents poor Jose from using our site, despite the fact that he's well over the minimum required age. The bug here is a *type mismatch*. The function `canRegister` was expecting a number, but was passed an object. JavaScript helpfully tries to make up for our crass mistake, sees an object used against `>`—an operator that expects a number, and converts that object to a number: `NaN`. What if we could discover innocuous bugs like this without even leaving our editor (and more importantly, before our users do)?

A superset of JavaScript, TypeScript adds optional types into JavaScript. We can add an optional type notation to `canRegister` that explicitly notes that `canRegister` is looking for a numerical type.

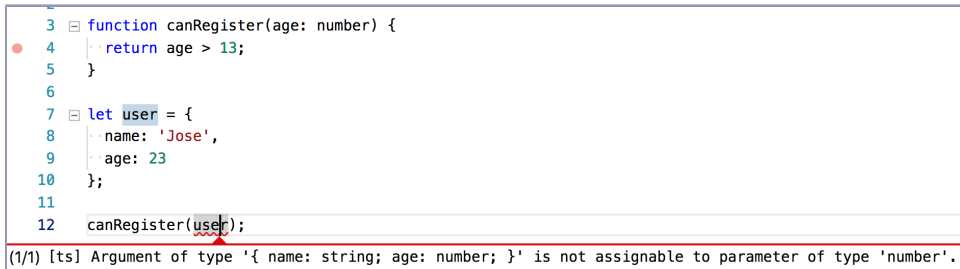
```
function canRegister(age: number) {
  return age > 13;
}

let user = {
  name: 'Jose',
  age: 23
};

canRegister(user);
```

In any editor with a TypeScript plugin, this will immediately reveal the bug. TypeScript knows the type of `user` is “an object with a `name` property set to a string, and an `age` property set to a number.” If we try to pass that through to a function that just wants a number, the type mismatch is revealed. While

it's possible to write Angular code in regular JavaScript, TypeScript is strongly recommended, and what we'll use here.



```

3 function canRegister(age: number) {
4   return age > 13;
5 }
6
7 let user = {
8   name: 'Jose',
9   age: 23
10 };
11
12 canRegister(user);

```

(1/1) [ts] Argument of type '{ name: string; age: number; }' is not assignable to parameter of type 'number'.

You don't need to know anything about TypeScript to read this book. All of the compilation will be handled for you as part of the code included in this book.

## Experimenting in the Sandbox

You may wish to travel off the beaten path and experiment with RxJS as you learn about various aspects of RxJS. The `sandbox.ts` file has been set up for this purpose (with a link to the corresponding HTML file from the project homepage). All of RxJS is available, so feel free to play around with whatever you like. To view the sandbox in your browser, start the server as mentioned above and browse to <http://localhost:8081/sandbox/sandbox.html>.

## Online Resources

Visit this book's online home,<sup>2</sup> where you'll find errata and any source code you need.

## Acknowledgments

Thanks to Kevin W. Beam, Alex Bezek, Nathan Brenner, Nick Capito, Chad Dumler-Montplaisir, Kevin Gisi, Steve Hill, Daivid Morgan, Peter Perlepes, and Stephen Wolff for reviewing early copies of this book. Thanks to Pascal Precht and Jurgen Van de Moere for all their writing and blogging about Angular and observables that taught me so much. Special thanks also goes to Rob Wormald for selflessly helping me when I was struggling to first learn RxJS. I'd also like to thank my editor, Brian MacDonald, for putting up with the many beginner mistakes I made during the development of this book.

2. <https://pragprog.com/book/rkrxs/build-reactive-websites-with-rxjs>