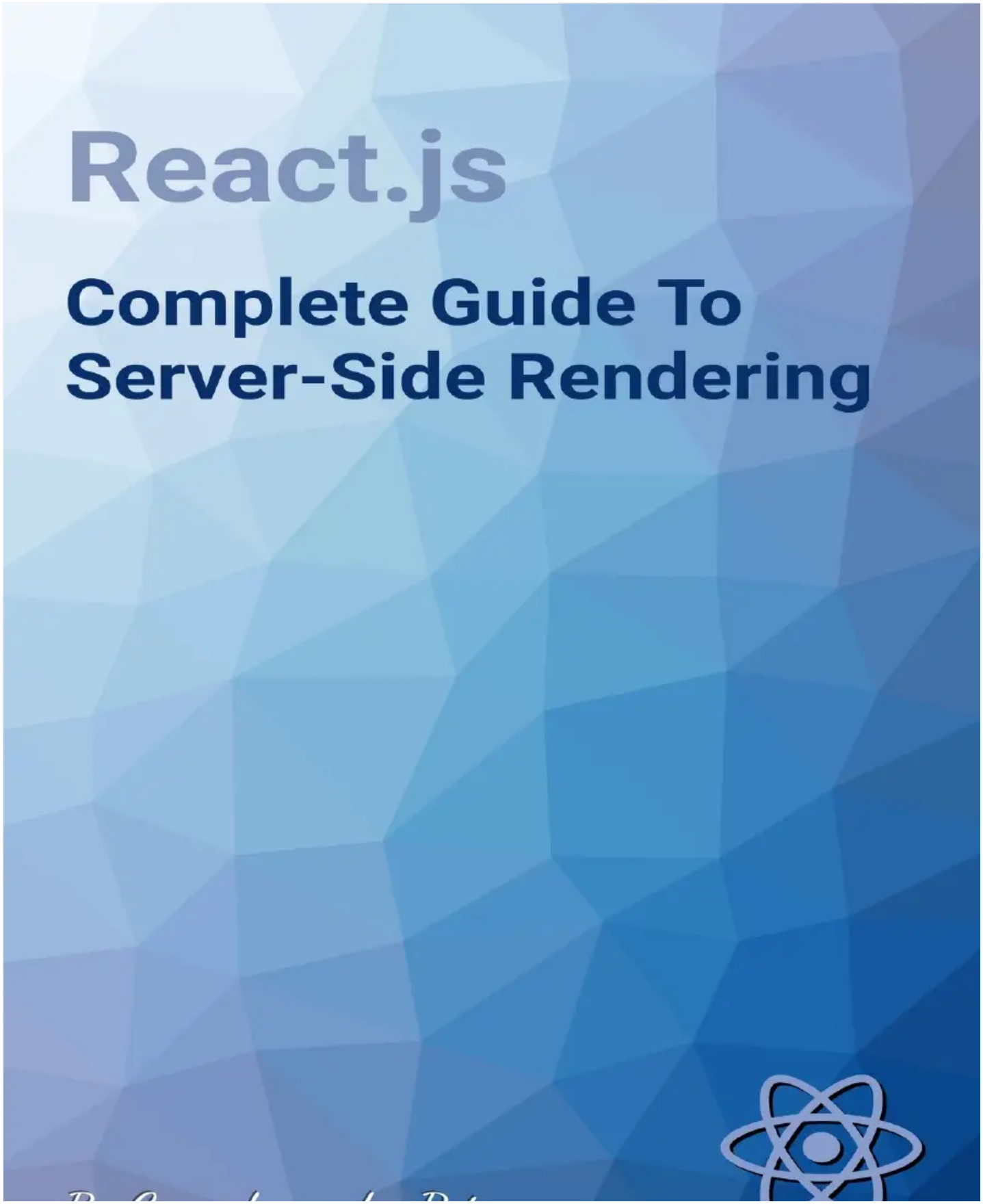
React.js
Complete Guide To
Server-Side Rendering

React.js

Complete Guide To Server-Side Rendering

Gerard van der Put, May 2020

Introduction

Many React developers find the term **server-side rendering** a scary one. It sounds like a concept from another universe. "Rendering a front-end React application on a server? But what happened to rendering it in the browser? Like I always do?"

That must be something for wizards only.

And if you finally dare to take the step to start searching the internet for information about this topic you will be pushed away even further: new terminology, difficult tutorials and as always countless opinions that could not differ more from each other.

Throughout many years I have collected all the small bits and pieces that are actually useful and compressed them into one single guide that will not only provide you with solutions; it will also explain them thoroughly so you will get a **solid foundation of knowledge** about this topic. Knowledge that you can bring with you when you start developing and maintaining your own server-side rendered React applications.

We will keep it minimalistic and as simple as possible.

Before we start to write code let's answer an important question first.

What Is Server-Side Rendering?

In a regular React application (without server-side rendering) we write our JavaScript and bundle this together in one or multiple files. Most commonly we then serve our users an HTML file with an empty body which - once loaded - will execute the JavaScript from our bundle. This React code will build and manipulate the DOM while the user is interacting with our application. We describe this scenario as **client-side rendering** since the rendering happens in the users browser. It happens on their machine.

Notice how we mentioned that "we serve the users an HTML file with an

empty body". This is the crucial part. When we talk about an application with server-side rendering we will not serve an empty HTML file. Instead, we will let our server serve an HTML file **with a pre-rendered DOM**. After that has happened everything will continue as normal. The JavaScript bundle will be loaded and executed. And React "will take over" and make our application interactive.

Benefits

Faster initial server response

When our users visit a traditional React application with client-side rendering they have to wait for the following events to complete *before they see the rendered DOM elements on the screen* in their browser, in chronological order:

1. Browser request to get the index file for the application
2. Server response with initial index HTML file (with empty body)
3. Browser request to get the JavaScript bundle
4. Server response with JavaScript bundle
5. Browser executes JavaScript code
6. JavaScript code (React) updates the DOM

Let's be fair: in most cases this happens all rather quick. But each of the six theoretical steps (*in a real-world example there would be more steps - think about loading additional assets such as CSS, images and fonts*) take a bit of time and it all adds up. You've probably experienced long initial loading times yourself when opening a website and I bet it frustrated you.

> *When initial loading times are too long users might "bounce". They don't bother to wait and leave your application before they've seen it.*

If you have a large (enterprise-) React application with a large JavaScript bundle the risk for this happening increases significantly. If your users have a rather slow internet connection on top of that you might struggle with

Share this document

Document • 47 pages

RESTful Day 1 PDF

enrique

No ratings yet

Document • 31 pages

React Handbook

Dwarakanath Reddy

No ratings yet

Document • 31 pages

Deploy To Heroku

lelana

No ratings yet

Document • 13 pages

Case Study

Sahil Sarwar

No ratings yet

Document • 31 pages

React Handbook

Noor

No ratings yet

Document • 89 pages

The Complete Beginner's Guide To React: by Kristen Dyrr

Raj Kumar

No ratings yet

Document • 34 pages

React Beginner Handbook

loridan

No ratings yet

Document • 38 pages

IP - Chapter No 6-React JS-SH 2022-Prepared by Reshma Koli

Velmurgan Santhanam

No ratings yet

Document • 31 pages

React Handbook

Document • 19 pages

Reactjs - Interlace Your Interface Introduction To Reactjs: What Is Exciting Here?

Mahesh VP

No ratings yet

Document • 37 pages

Report

BHAVYA JYOTI SINGH

No ratings yet

Document • 26 pages

React

Shyam Santhosh

No ratings yet

Show more