



Table of Contents

| | |
|--|-------|
| Introduction | 1.1 |
| 1. What Is React | 1.2 |
| 2. React Semantics/Terminology | 1.3 |
| 3. React & Babel Basic Setup | 1.4 |
| 3.1 Using react.js & react-dom.js | 1.4.1 |
| 3.2 Using JSX via Babel | 1.4.2 |
| 3.3 Using ES6 & ES* with React | 1.4.3 |
| 3.4 Writing React With JSFiddle | 1.4.4 |
| 4. React Nodes | 1.5 |
| 4.1 What Is a React Node? | 1.5.1 |
| 4.2 Creating React Nodes | 1.5.2 |
| 4.3 Rendering to DOM | 1.5.3 |
| 4.4 Defining Node Attributes/Props | 1.5.4 |
| 4.5 Inlining CSS on Element Nodes | 1.5.5 |
| 4.6 Using Built-in Element Factories | 1.5.6 |
| 4.7 Defining React Node Events | 1.5.7 |
| 5. JavaScript Syntax Extension (a.k.a., JSX) | 1.6 |
| 5.1 What Is a JSX? | 1.6.1 |
| 5.2 Creating React Nodes With JSX | 1.6.2 |
| 5.3 Rendering JSX to DOM | 1.6.3 |
| 5.4 Using JS Expressions in JSX | 1.6.4 |
| 5.5 Using JS Comments in JSX | 1.6.5 |
| 5.6 Using Inline CSS in JSX | 1.6.6 |
| 5.7 Defining JSX Attributes/Props | 1.6.7 |
| 5.8 Defining Events in JSX | 1.6.8 |
| 6. Basic React Components | 1.7 |
| 6.1 What Is a React Component? | 1.7.1 |
| 6.2 Creating Components | 1.7.2 |
| 6.3 Return One Starting Node/Component | 1.7.3 |
| 6.4 Referring to a Component Instance | 1.7.4 |

| | |
|--|--------|
| 6.5 Defining Events on Components | 1.7.5 |
| 6.6 Composing Components | 1.7.6 |
| 6.7 Grokking Component Lifecycle's | 1.7.7 |
| 6.8 Accessing Children Components/Nodes | 1.7.8 |
| 6.9 Using ref Attribute | 1.7.9 |
| 6.10 Re-rendering A Component | 1.7.10 |
| 7. React Component Props | 1.8 |
| 7.1 What Are Component Props? | 1.8.1 |
| 7.2 Sending Component Props | 1.8.2 |
| 7.3 Getting Component Props | 1.8.3 |
| 7.4 Setting Default Component Props | 1.8.4 |
| 7.5 Component Props More Than Strings | 1.8.5 |
| 7.6 Validating Component Props | 1.8.6 |
| 8. React Component State | 1.9 |
| 8.1 What Is Component State? | 1.9.1 |
| 8.2 Working with Component State | 1.9.2 |
| 8.3 State vs. Props | 1.9.3 |
| 8.4 Creating Stateless Function Components | 1.9.4 |

React Enlightenment

Written by [Cody Lindley](#) sponsored by — [Frontend Masters](#)

Learn React in the terse cookbook style found with previous Enlightenment titles (i.e., [jQuery Enlightenment](#), [JavaScript Enlightenment](#), [DOM Enlightenment](#))

Read Online At:

- reactenlightenment.com

download a .pdf, .epub, or .mobi file from:

- <https://www.gitbook.com/book/frontendmasters/react-enlightenment/details>

contribute content, suggestions, and fixes on github:

- <https://github.com/FrontendMasters/react-enlightenment>
-



This work is licensed under a [Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License](#).

What is React?

React is a JavaScript tool that makes it easy to reason about, construct, and maintain stateless and stateful user interfaces. It provides the means to declaratively define and divide a UI into UI components (a.k.a., React components) using HTML-like nodes called React nodes. React nodes eventually get transformed into a format for UI rendering (e.g., HTML/DOM, canvas, svg, etc.).

I could ramble on trying to express in words what React is, but I think it best to just show you. What follows is a whirlwind tour of React and React components from thirty thousand feet. Don't try and figure out all the details yet as I describe React in this section. The entire book is meant to unwrap the details showcased in the following overview. Just follow along grabbing a hold of the big concepts for now.

Using React to create UI components similar to a `<select>`

Below is an HTML `<select>` element that encapsulates child HTML `<option>` elements. Hopefully the creation and functionality of an HTML `<select>` is already familiar.

[source code](#)

When a browser parses the above tree of elements it will produce a UI containing a textual list of items that can be selected. Click on the "Result" tab in the above JSFiddle, to see what the browser produces.

The browser, [the DOM](#), and the [shadow DOM](#) are working together behind the scenes to turn the `<select>` HTML into a UI component. Note that the `<select>` component allows the user to make a selection thus storing the state of that selection (i.e., click on "Volvo", and you have selected it instead of "Mercedes").

Using React you can create a custom `<select>` by using React nodes to make a React component that eventually will result in HTML elements in an HTML DOM.

Let's create our own `<select>`-like UI component using React.

Defining a React component

[Click here to download full PDF material](#)