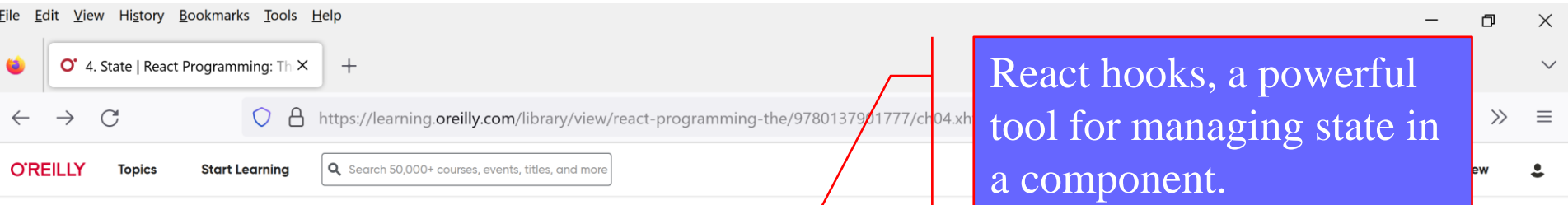


# Using React

**Dr. Atef Bader**

# Ottergram React App



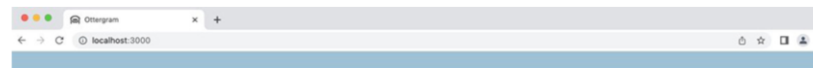
## Chapter 4. State

While it is great that Barry the otter is getting some time to shine, all the otters should have equal opportunities for fame. Your app currently listens for user events and responds by logging the name of the otter that the user clicks. In this chapter, you will finish up Ottergram by adding *state* to keep track of and display the featured otter.

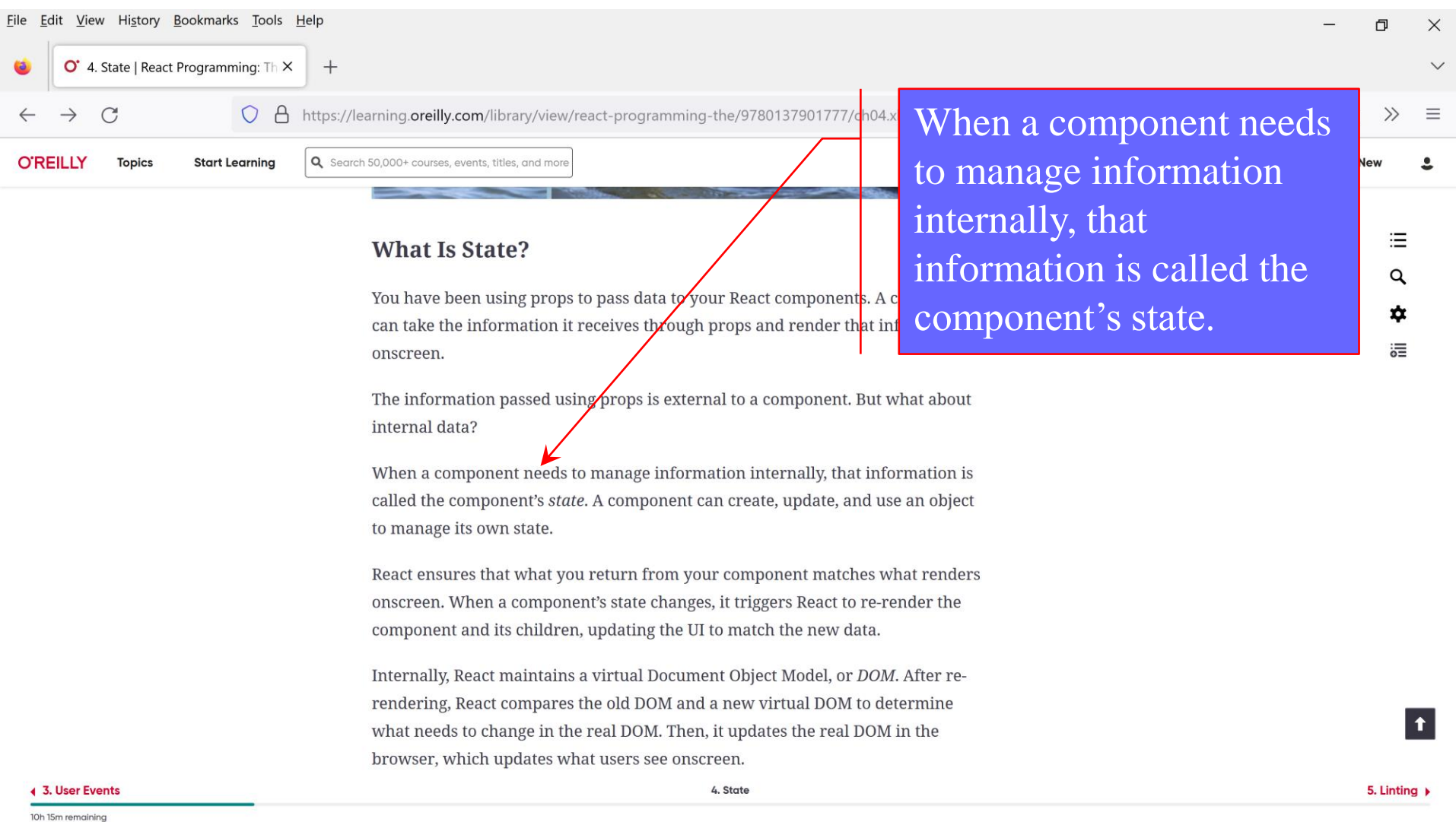
This chapter introduces React *hooks*, a powerful tool for managing state in a component. You will continue to learn about hooks throughout this book, so do not worry about mastering them right away. In this chapter, you will focus on using the `useState` hook to manage a component's state.

By the end of this chapter, your Ottergram application will be complete ([Figure 4.1, “Completed Ottergram”](#)).

Figure 4.1. Completed Ottergram



# Ottergram React App



The screenshot shows a web browser window with the address bar displaying <https://learning.oreilly.com/library/view/react-programming-the/9780137901777/ch04.x>. The page title is "4. State | React Programming: Th". The page content is titled "What Is State?" and explains the concept of state in React. A blue callout box with a red border contains the text: "When a component needs to manage information internally, that information is called the component's state." A red arrow points from this box to the sentence: "When a component needs to manage information internally, that information is called the component's *state*." The page also includes a navigation bar at the bottom with links for "3. User Events", "4. State", and "5. Linting".

File Edit View History Bookmarks Tools Help

4. State | React Programming: Th

https://learning.oreilly.com/library/view/react-programming-the/9780137901777/ch04.x

O'REILLY Topics Start Learning Search 50,000+ courses, events, titles, and more

## What Is State?

You have been using props to pass data to your React components. A component can take the information it receives through props and render that information onscreen.

The information passed using props is external to a component. But what about internal data?

When a component needs to manage information internally, that information is called the component's *state*. A component can create, update, and use an object to manage its own state.

React ensures that what you return from your component matches what renders onscreen. When a component's state changes, it triggers React to re-render the component and its children, updating the UI to match the new data.

Internally, React maintains a virtual Document Object Model, or *DOM*. After re-rendering, React compares the old DOM and a new virtual DOM to determine what needs to change in the real DOM. Then, it updates the real DOM in the browser, which updates what users see onscreen.

3. User Events 4. State 5. Linting

10h 15m remaining

# Ottergram React App

4. State | React Programming: The

https://learning.oreilly.com/library/view/react-programming-the/9780137901777/ch04.xhtml

O'REILLY Topics Start Learning Search 50,000+ courses, events, titles, and more

## What Is State?

You have been using props to pass data to your React components. A component can take the information it receives through props and render that information onscreen.

The information passed using props is external to a component. But what about internal data?

When a component needs to manage information internally, that information is called the component's *state*. A component can create, update, and use an object to manage its own state.

React ensures that what you return from your component matches what renders onscreen. When a component's state changes, it triggers React to re-render the component and its children, updating the UI to match the new data.

Internally, React maintains a virtual Document Object Model, or *DOM*. After re-rendering, React compares the old DOM and a new virtual DOM to determine what needs to change in the real DOM. Then, it updates the real DOM in the browser, which updates what users see onscreen.

3. User Events 4. State 5. Linting

10h 15m remaining

# Ottergram React App

The screenshot shows a web browser window with the URL <https://learning.oreilly.com/library/view/react-programming-the/9780137901777/ch04.x>. The page title is "4. State | React Programming: Th". The page content is titled "What Is State?" and discusses the concept of state in React. A blue callout box with a red border is overlaid on the right side of the page, containing the text: "Internally, React maintains a virtual Document Object Model, or DOM. After re-rendering, React compares the old DOM and a new virtual DOM to determine what needs to change in the real DOM". A red arrow points from the bottom of the callout box to the text "Internally, React maintains a virtual Document Object Model, or *DOM*." in the article.

File Edit View History Bookmarks Tools Help

4. State | React Programming: Th X

https://learning.oreilly.com/library/view/react-programming-the/9780137901777/ch04.x

O'REILLY Topics Start Learning Search 50,000+ courses, events, titles, and more

## What Is State?

You have been using props to pass data to your React components. A component can take the information it receives through props and render that information onscreen.

The information passed using props is external to a component. But what about internal data?

When a component needs to manage information internally, that information is called the component's *state*. A component can create, update, and use state to manage its own state.

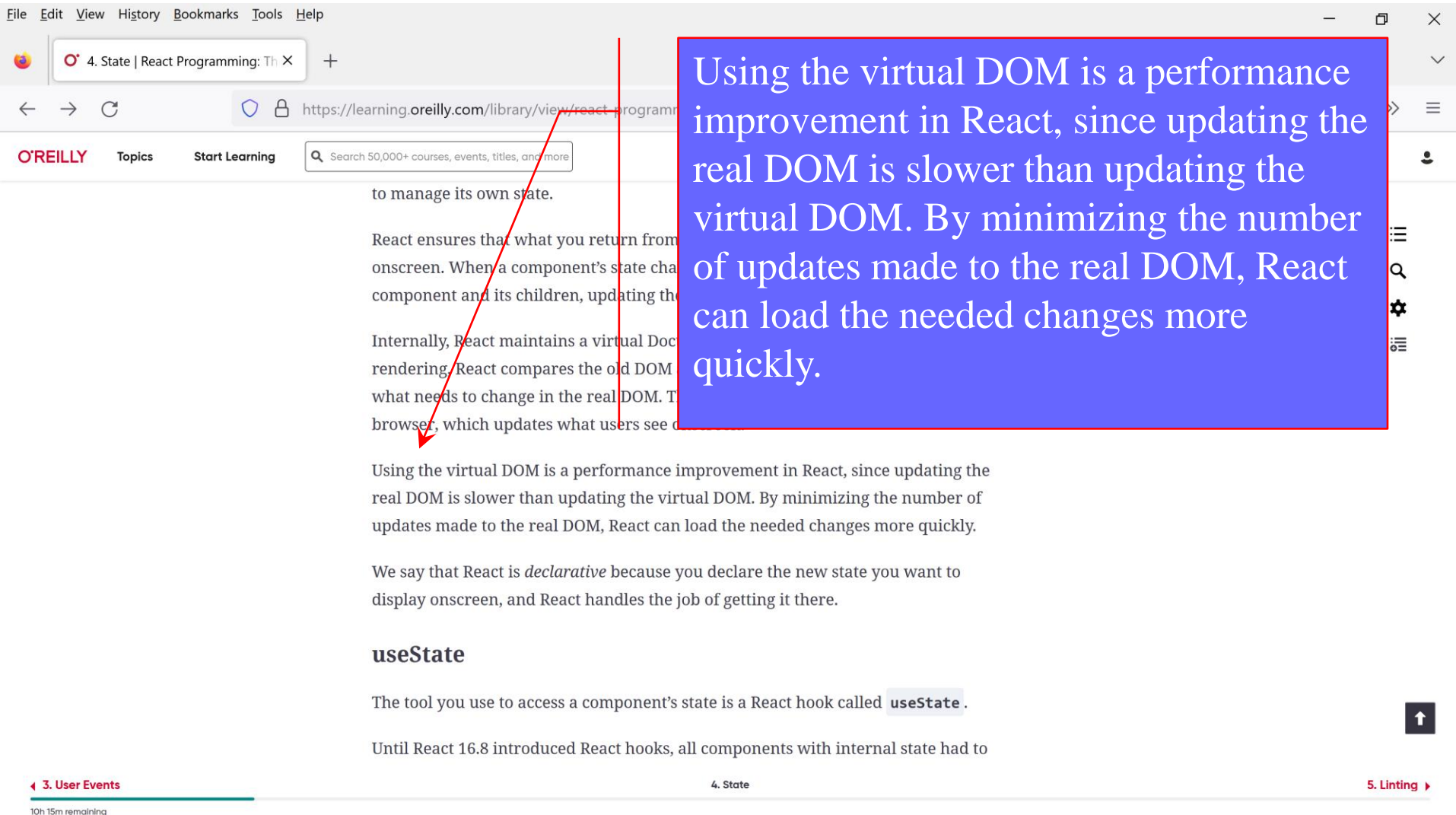
React ensures that what you return from your component matches what renders onscreen. When a component's state changes, it triggers React to re-render the component and its children, updating the UI to match the new data.

Internally, React maintains a virtual Document Object Model, or *DOM*. After re-rendering, React compares the old DOM and a new virtual DOM to determine what needs to change in the real DOM. Then, it updates the real DOM in the browser, which updates what users see onscreen.

3. User Events 4. State 5. Linting

10h 15m remaining

# Ottergram React App



The screenshot shows a web browser window with the address bar displaying `https://learning.oreilly.com/library/view/react-programming/`. The page content includes a search bar, navigation links, and several paragraphs of text. A blue callout box with a red border is overlaid on the right side of the page, containing text about the virtual DOM. A red arrow points from the callout box to the text in the article.

File Edit View History Bookmarks Tools Help

4. State | React Programming: Th X

← → ↻ 🔒 https://learning.oreilly.com/library/view/react-programming/

O'REILLY Topics Start Learning 🔍 Search 50,000+ courses, events, titles, and more

to manage its own state.

React ensures that what you return from `render` is what you see onscreen. When a component's state changes, React updates the component and its children, updating the DOM.

Internally, React maintains a virtual DOM. When you call `render`, React compares the old DOM with the new one to determine what needs to change in the real DOM. Then, React updates the browser, which updates what users see onscreen.

Using the virtual DOM is a performance improvement in React, since updating the real DOM is slower than updating the virtual DOM. By minimizing the number of updates made to the real DOM, React can load the needed changes more quickly.

We say that React is *declarative* because you declare the new state you want to display onscreen, and React handles the job of getting it there.

## useState

The tool you use to access a component's state is a React hook called `useState`.

Until React 16.8 introduced React hooks, all components with internal state had to

3. User Events 4. State 5. Linting

10h 15m remaining

# Ottergram React App

App.js - 04-state - Visual Studio Code

EXPLORER

- 04-STATE
  - node\_modules
  - ottergram
  - public
    - favicon.ico
    - index.html
    - logo192.png
    - logo512.png
    - manifest.json
    - robots.txt
  - src
    - components
      - Header.js
      - Post.js
      - SelectedItem.js
    - otters
    - App.css
    - App.js**
    - App.test.js
    - index.css
    - index.js
    - logo.svg
    - reportWebVitals.js
    - setupTests.js
    - .gitignore
    - package-lock.json
    - package.json
    - README.md

JS App.js

```
src > JS App.js > App > setSelectedPostName
9 import Lesley from './otters/otter4.jpg';
10 import Barbara from './otters/otter5.jpg';
11
12 const ottersArray = [
13   { image: Barry, name: 'Barry', id: 1 },
14   { image: Robin, name: 'Robin', id: 2 },
15   { image: Maurice, name: 'Maurice', id: 3 },
16   { image: Lesley, name: 'Lesley', id: 4 },
17   { image: Barbara, name: 'Barbara', id: 5 },
18 ];
19
20 function App() {
21   const [selectedPostName, setSelectedPostName] = useState('Barry');
22   const selectedPost = ottersArray.find(otter => otter.name === selectedPostName);
23
24   return (
25     <div>
26       <Header />
27       <div className='app-content'>
28         <ul className='post-list'>
29           {ottersArray.map((post) => (
30             <Post
31               key={post.id}
32               image={post.image}
33               name={post.name}
34               setSelectedPostName={setSelectedPostName}
35             />
36           ))}
37         </ul>
38         <SelectedItem
39           image={selectedPost.image}
40           name={selectedPost.name}
41         />
42       </div>
43     </div>
44   );
45 }
46
47 export default App;
```

useState  
Hook function in  
App.js

Ln 21, Col 47 (19 selected) Spaces: 2 UTF-8 LF {} JavaScript

# Ottergram React App

Event Handler in Post.js

```
1 function Post({ image, name, setSelectedPostName }) {
2   return (
3     <li className='post-component'>
4       <button onClick={() => setSelectedPostName(name)}>
5         <img src={image} alt={name}/>
6         <p className="post-name">{name}</p>
7       </button>
8     </li>
9   )
10 }
11
12 export default Post;
13
```

Ln 4, Col 53 (19 selected) Spaces: 2 UTF-8 LF {} JavaScript



# Ottergram React App

```
Windows PowerShell
> react-scripts start

Browserslist: caniuse-lite is outdated. Please run:
  npx update-browserslist-db@latest
  Why you should do it regularly: https://github.com/browserslist/update-db#readme
(node:12664) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: 'onAfterSetupMiddleware' option is deprecated. Please use t
(Use 'node --trace-deprecation ...' to show where the warning was created)
(node:12664) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: 'onBeforeSetupMiddleware' option is deprecated. Please use
Starting the development server...
Compiled successfully!

You can now view ottergram in the browser.

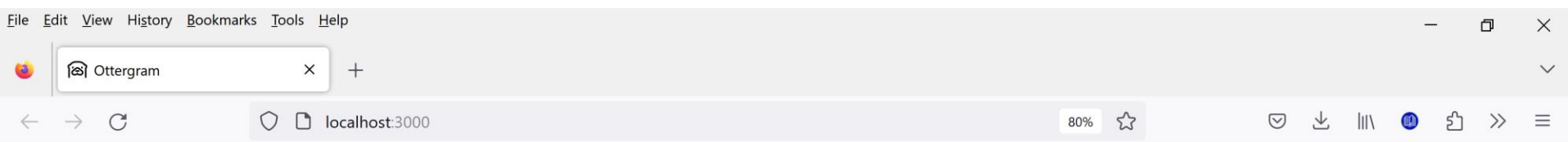
Local:      http://localhost:3000
On Your Network: http://192.168.0.222:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

npm start

# Ottergram React App



## OTTERGRAM



Robin

# References

- Klingman, L. 2023. React Programming: The Big Nerd Ranch Guide. Indianapolis, IN: Pearson Technology Group. [ISBN: 978-0137901692]
-