

React

biblioteka JavaScript służąca do tworzenia interfejsów użytkownika

tutorial version 1.11

frontend JavaScript framework

tool for building UI components

React

Instead of manipulating the browser's DOM directly, React creates a virtual DOM in memory, where it does all the necessary manipulating, before making the changes in the browser DOM

<https://react.dev/>

JavaScript library created by Facebook

creates a VIRTUAL DOM in memory

https://www.w3schools.com/REACT/react_intro.asp

React tutorials

[LEARN REACT](#) >

Quick Start

Welcome to the React documentation! This page will give you an introduction to the 80% of React concepts that you will use on a daily basis.

You will learn

- How to create and nest components
- How to add markup and styles
- How to display data
- How to render conditions and lists
- How to respond to events and update the screen
- How to share data between components

[LEARN REACT](#) > [QUICK START](#) >

Tutorial: Tic-Tac-Toe

You will build a small tic-tac-toe game during this tutorial. This tutorial does not assume any existing React knowledge. The techniques you'll learn in the tutorial are fundamental to building any React app, and fully understanding it will give you a deep understanding of React.

Note

This tutorial is designed for people who prefer to learn by doing and want to quickly try making something tangible. If you prefer learning each concept step by step, start with Describing the UI.

<https://react.dev/learn>

<https://react.dev/learn/tutorial-tic-tac-toe>

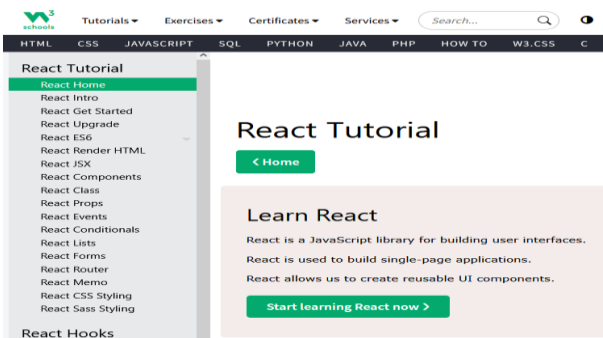
Thinking in React

React can change how you think about the designs you look at and the apps you build. When you build a user interface with React, you will first break it apart into pieces called *components*. Then, you will describe the different visual states for each of your components. Finally, you will connect your components together so that the data flows through them. In this tutorial, we'll guide you through the thought process of building a searchable product data table with React.

Start with the mockup

Imagine that you already have a JSON API and a mockup from a designer.

The JSON API returns some data that looks like this:



<https://react.dev/learn/thinking-in-react>

[w3schools tutorial](https://w3schools.com/react/)

React

React bezpośrednio w pliku HTML (do celów testowych, w produkcji stosuje się specjalne środowisko)

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <script src="https://unpkg.com/react@18/umd/react.development.js" crossorigin></script>
5     <script src="https://unpkg.com/react-dom@18/umd/react-dom.development.js" crossorigin></script>
6     <script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>
7   </head>
8   <body>
9
10    <div id="root"></div>
11
12    <script type="text/babel">
13      function Hello() {
14        return <h1>Hello World React!</h1>;
15      }
16
17      ReactDOM.render(<Hello />, document.getElementById('root'))
18    </script>
19
20  </body>
21 </html>
```

React opiera się na komponentach

Babel konwertuje JSX na JavaScript

tu React wstrzykuje wyrenderowany element Hello

Function Component Hello, czyli definicja elementu Hello (kodu html, który znajduje się w tym elemencie). Nazwa komponentu z DUŻEJ LITERY!

React renderuje element Hello w znaczniku
`<div id="root"></div>`

Hello World React!

React instalacja

Introduction

Getting started

- What is Angular?
- Try it
- Setup

Understanding Angular

Developer guides

Best practices

Angular tools

Tutorials

Updates and releases

Reference

Documentation contributors guide

Install the Angular CLI

You can use the Angular CLI to create projects, generate application and library code, and perform a variety of ongoing development tasks such as testing, bundling, and deployment.

To install the Angular CLI, open a terminal window and run the following command:

```
npm install -g @angular/cli
```

On Windows client computers, the execution of PowerShell scripts is disabled by default. To allow the execution of PowerShell scripts, which is needed for npm global binaries, you must set the following [execution policy](#):

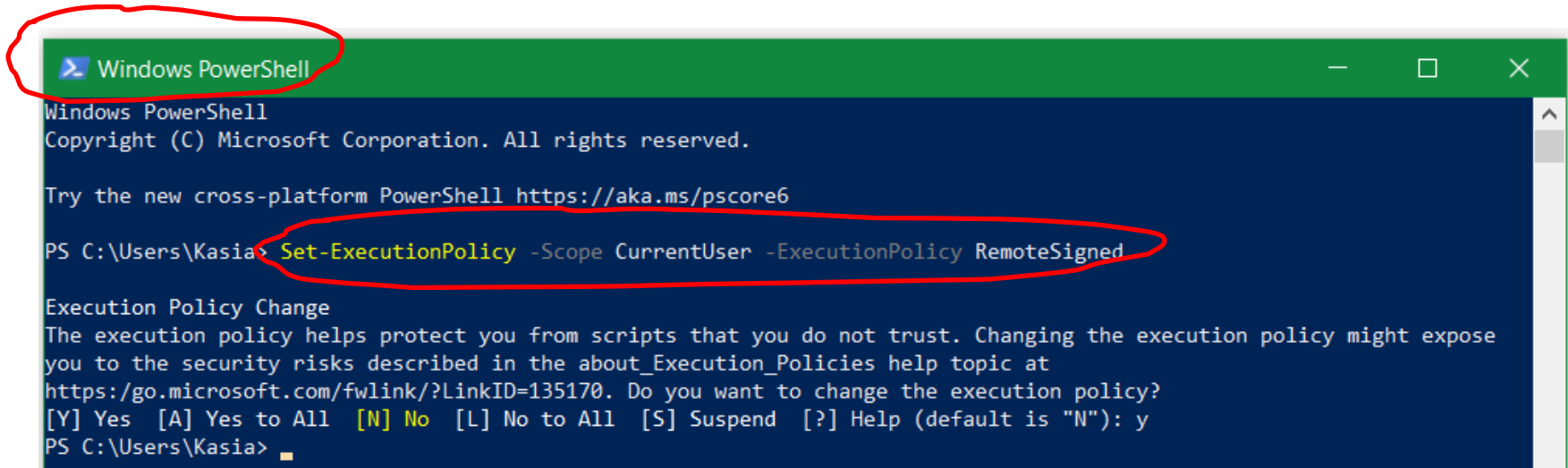
```
Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy RemoteSigned
```

Carefully read the message displayed after executing the command and follow the instructions. Make sure you understand the implications of setting an execution policy.

<https://angular.io/guide/setup-local>

w systemie Windows należy włączyć możliwość wykonywania skryptów w PowerShellu

React instalacja



A screenshot of a Windows PowerShell terminal window. The title bar is green and says "Windows PowerShell". The terminal has a dark blue background with white text. The text in the terminal includes: "Windows PowerShell", "Copyright (C) Microsoft Corporation. All rights reserved.", "Try the new cross-platform PowerShell <https://aka.ms/powershell>", and a command prompt "PS C:\Users\Kasia>". The command entered is `Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy RemoteSigned`, which is circled in red. Below the command, there is a message about execution policy change and a prompt to confirm the change. The user has responded with 'y'.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/powershell

PS C:\Users\Kasia> Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy RemoteSigned

Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose
you to the security risks described in the about_Execution_Policies help topic at
https://go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution policy?
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): y
PS C:\Users\Kasia>
```

komendę wykonujemy w PowerShellu, a nie w cmd

React

instalacja środowiska w Visual Studio Code

instalacja
środowiska
Node.js



instalacja
środowiska
React w
terminalu

```
npm install -g create-react-app
```

utworzenie
projektu
helloworld
w terminalu

```
npx create-react-app helloworld
```

```
Success! Created helloworld at D:\React\programy\HelloWorld2\helloworld
Inside that directory, you can run several commands:

npm start
  Starts the development server.

npm run build
  Bundles the app into static files for production.

npm test
  Starts the test runner.

npm run eject
  Removes this tool and copies build dependencies, configuration files
  and scripts into the app directory. If you do this, you can't go back!

We suggest that you begin by typing:

cd helloworld
npm start

Happy hacking!
```

błędy przy tworzeniu nowego projektu

```
Initialized a git repository.

Installing template dependencies using npm...
npm error code ERESOLVE
npm error ERESOLVE unable to resolve dependency tree
npm error
npm error While resolving: mikolaj@0.1.0
npm error Found: react@19.0.0
npm error node_modules/react
npm error   react@"^19.0.0" from the root project
npm error
npm error Could not resolve dependency:
npm error peer react@"^18.0.0" from @testing-library/react@13.4.0
npm error node_modules/@testing-library/react
npm error   peer react@"^18.0.0" from @testing-library/react@13.4.0
```

<https://github.com/facebook/react/issues/31699>

```
npm install -g yarn
yarn create react-app my-app
cd my-app
yarn start
```



alternatywny sposób utworzenia projektu my-app
oraz uruchomienia serwera

React

instalacja środowiska w Visual Studio Code

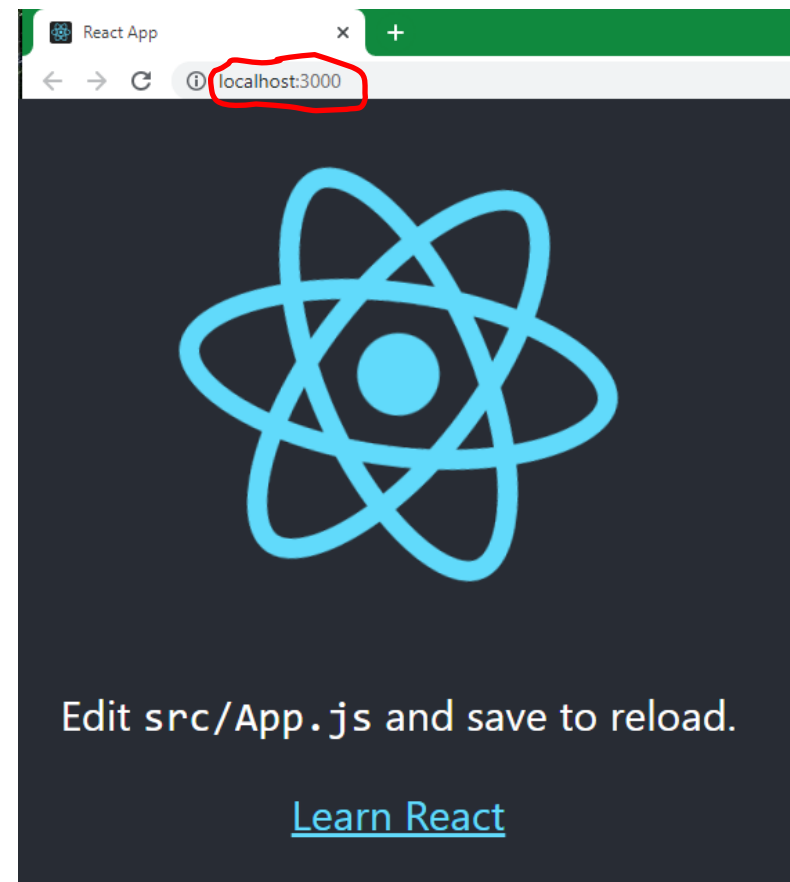
start serwera React

cd helloworld
npm start

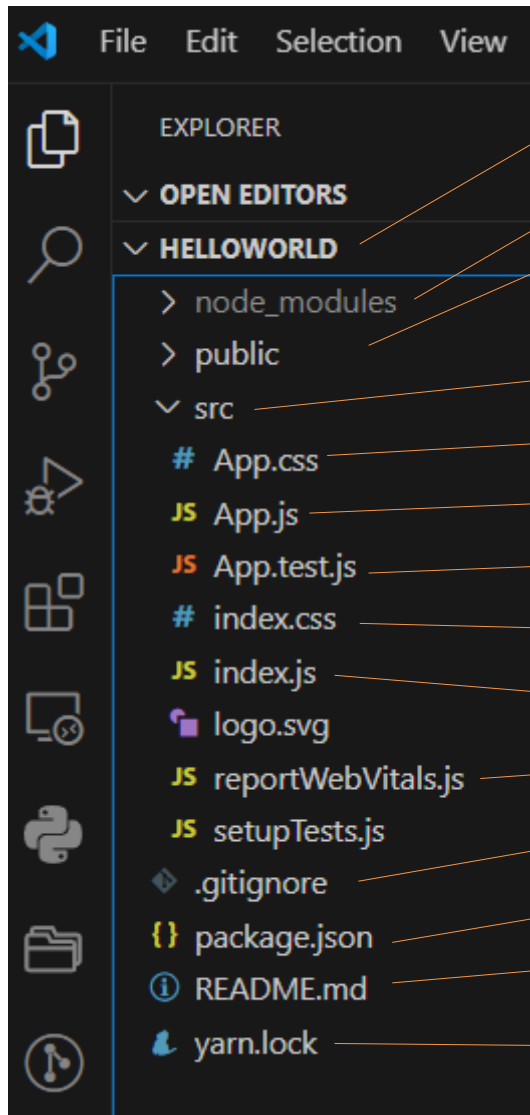
przejdźcie do katalogu projektu

serwer React nasłuchuje na porcie 3000

```
PS D:\React\programy\HelloWorld2> cd .\helloworld\  
PS D:\React\programy\HelloWorld2\helloworld> npm start  
Compiled successfully!  
  
You can now view helloworld in the browser.  
  
Local:      http://localhost:3000  
On Your Network: http://172.25.80.1:3000  
  
Note that the development build is not optimized.  
To create a production build, use npm run build.  
  
webpack compiled successfully  
█
```



React pliki projektu



nazwa projektu

folder z zainstalowanymi bibliotekami, modułami

folder z plikami statycznymi (ikony, logo itp.) oraz plikiem wejściowym aplikacji index.html

folder z kodem źródłowym aplikacji

css głównego komponentu aplikacji

główny komponent aplikacji (o nazwie App)

plik testowy głównego komponentu aplikacji

css strony aplikacji

plik, w którym w którym React renderuje komponent App

plik służący do zbierania i analizowania metryk wydajności aplikacji w celu jej optymalizacji

plik definiujący, które pliki lub foldery mają być ignorowane przez Git

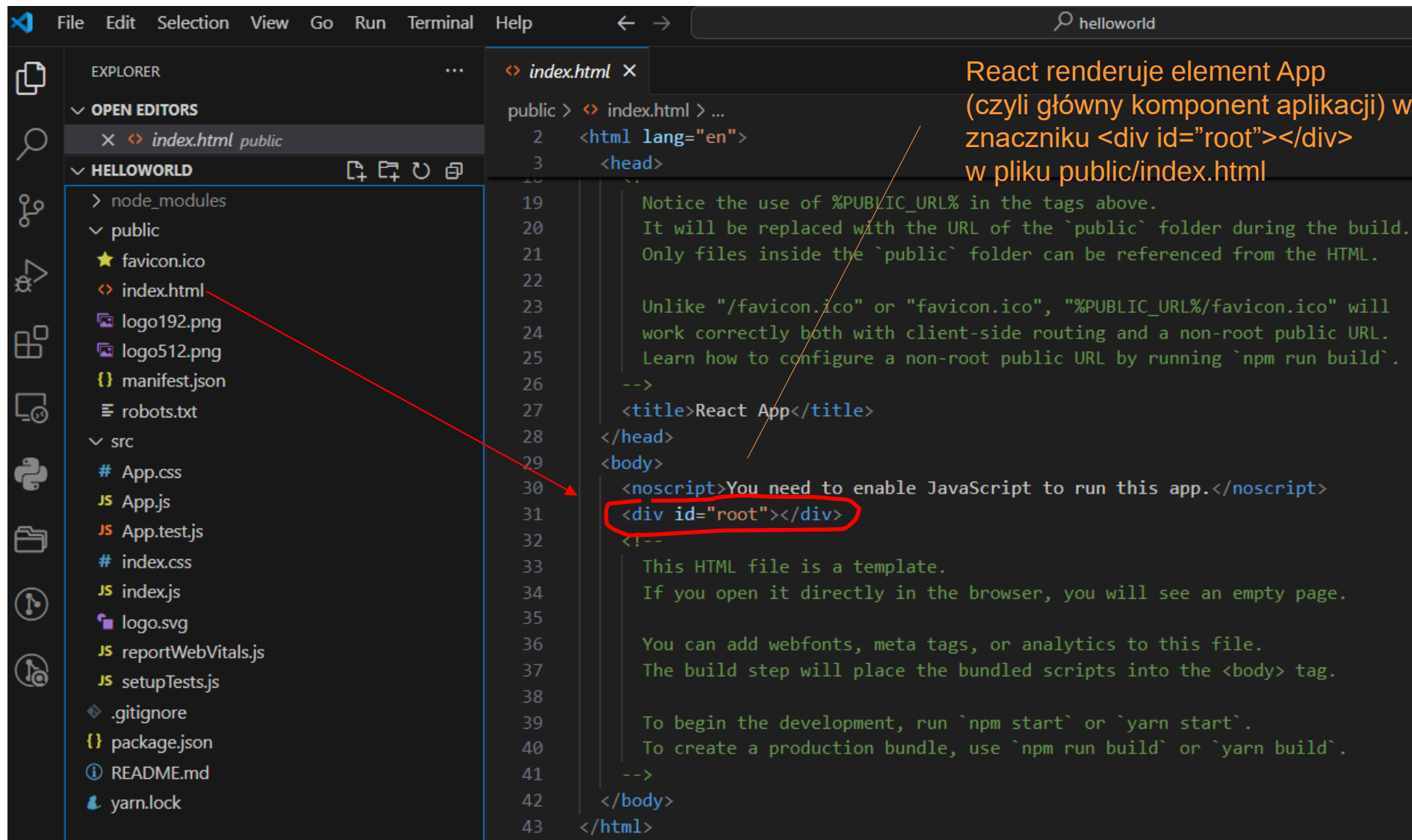
plik konfiguracyjny projektu

plik dokumentacji projektu

(lub package-lock.json) plik, który zapisuje dokładne wersje zależności, aby zapewnić spójność instalacji

React renderowanie

przygotowanie widoku elementu



EXPLORER

OPEN EDITORS

- index.html public

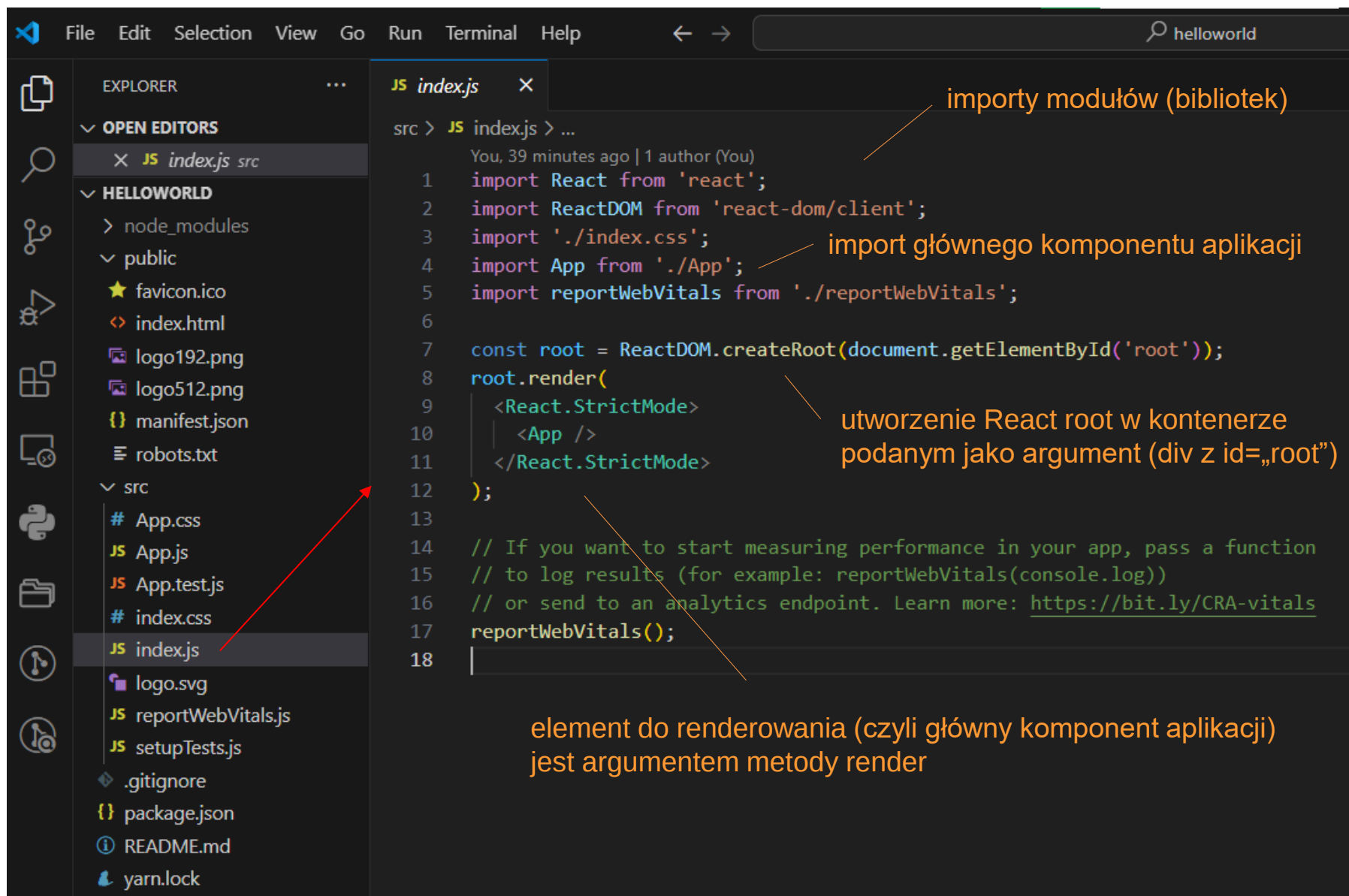
HELLOWORLD

- node_modules
- public
 - favicon.ico
 - index.html
 - logo192.png
 - logo512.png
 - manifest.json
 - robots.txt
- src
 - App.css
 - App.js
 - App.test.js
 - index.css
 - index.js
 - logo.svg
 - reportWebVitals.js
 - setupTests.js
 - .gitignore
 - package.json
 - README.md
 - yarn.lock

```
public > < index.html > ...
2  <html lang="en">
3  <head>
18  ...
19  Notice the use of %PUBLIC_URL% in the tags above.
20  It will be replaced with the URL of the `public` folder during the build.
21  Only files inside the `public` folder can be referenced from the HTML.
22
23  Unlike `"/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
24  work correctly both with client-side routing and a non-root public URL.
25  Learn how to configure a non-root public URL by running `npm run build`.
26  -->
27  <title>React App</title>
28 </head>
29 <body>
30  <noscript>You need to enable JavaScript to run this app.</noscript>
31  <div id="root"></div>
32  <!--
33    This HTML file is a template.
34    If you open it directly in the browser, you will see an empty page.
35
36    You can add webfonts, meta tags, or analytics to this file.
37    The build step will place the bundled scripts into the <body> tag.
38
39    To begin the development, run `npm start` or `yarn start`.
40    To create a production bundle, use `npm run build` or `yarn build`.
41  -->
42 </body>
43 </html>
```

React renderuje element App (czyli główny komponent aplikacji) w znaczniku `<div id="root"></div>` w pliku public/index.html

React renderowanie



The image shows a screenshot of the Visual Studio Code editor interface. The Explorer sidebar on the left displays the project structure, including a `src` directory with files like `App.css`, `App.js`, `App.test.js`, `index.css`, `index.js`, `logo.svg`, `reportWebVitals.js`, and `setupTests.js`. The `index.js` file is selected and open in the main editor. The code in `index.js` is as follows:

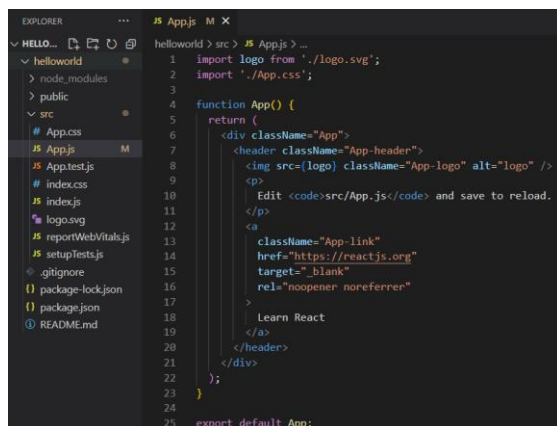
```
src > JS index.js > ...
You, 39 minutes ago | 1 author (You)
1  import React from 'react';
2  import ReactDOM from 'react-dom/client';
3  import './index.css';
4  import App from './App';
5  import reportWebVitals from './reportWebVitals';
6
7  const root = ReactDOM.createRoot(document.getElementById('root'));
8  root.render(
9    <React.StrictMode>
10     <App />
11   </React.StrictMode>
12 );
13
14 // If you want to start measuring performance in your app, pass a function
15 // to log results (for example: reportWebVitals(console.log))
16 // or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
17 reportWebVitals();
18
```

Annotations in orange text with arrows point to specific parts of the code:

- importy modułów (bibliotek)** points to line 1: `import React from 'react';`
- import głównego komponentu aplikacji** points to line 4: `import App from './App';`
- utworzenie React root w kontenerze podanym jako argument (div z id="root")** points to line 7: `const root = ReactDOM.createRoot(document.getElementById('root'));`
- element do renderowania (czyli główny komponent aplikacji) jest argumentem metody render** points to line 10: `<App />`

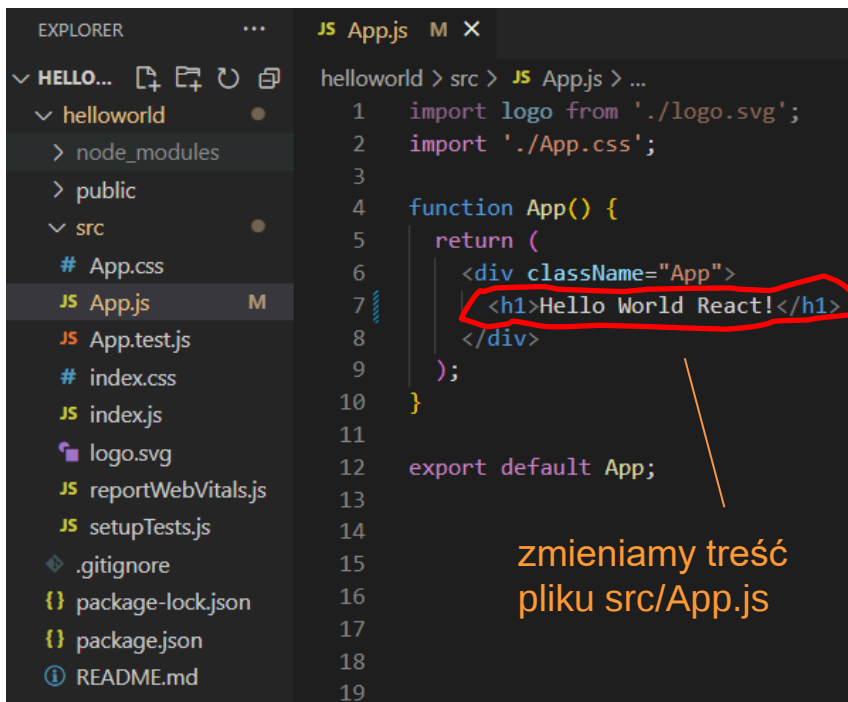
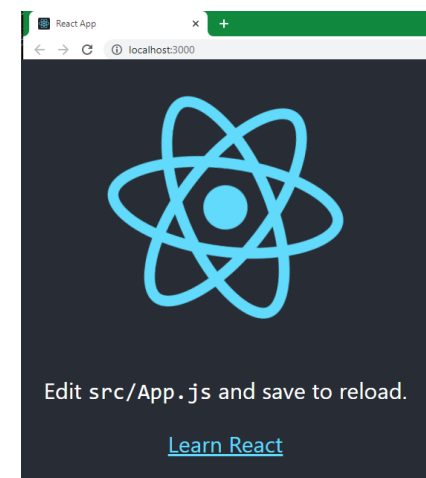
React

zmiana treści pliku App.js



The image shows the VS Code Explorer on the left with the file tree expanded to 'src', where 'App.js' is selected. The main editor shows the content of 'App.js' with the following code:

```
1 import logo from './logo.svg';
2 import './App.css';
3
4 function App() {
5   return (
6     <div className="App">
7       <header className="App-header">
8         <img src={logo} className="App-logo" alt="logo" />
9         <p>
10           Edit <code>src/App.js</code> and save to reload.
11         </p>
12         <a
13           className="App-link"
14           href="https://reactjs.org"
15           target="_blank"
16           rel="noopener noreferrer"
17         >
18           Learn React
19         </a>
20       </header>
21     </div>
22   );
23 }
24
25 export default App;
```

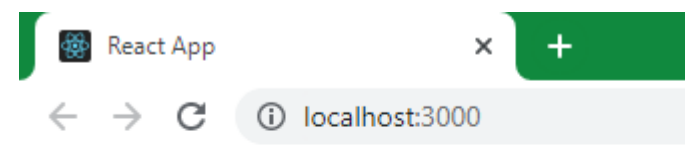


The image shows the VS Code Explorer on the left with the file tree expanded to 'src', where 'App.js' is selected. The main editor shows the content of 'App.js' with the following code:

```
1 import logo from './logo.svg';
2 import './App.css';
3
4 function App() {
5   return (
6     <div className="App">
7       <h1>Hello World React!</h1>
8     </div>
9   );
10 }
11
12 export default App;
```

An orange arrow points from the text 'zmieniamy treść pliku src/App.js' to the line containing the new JSX element.

zmieniamy treść pliku src/App.js



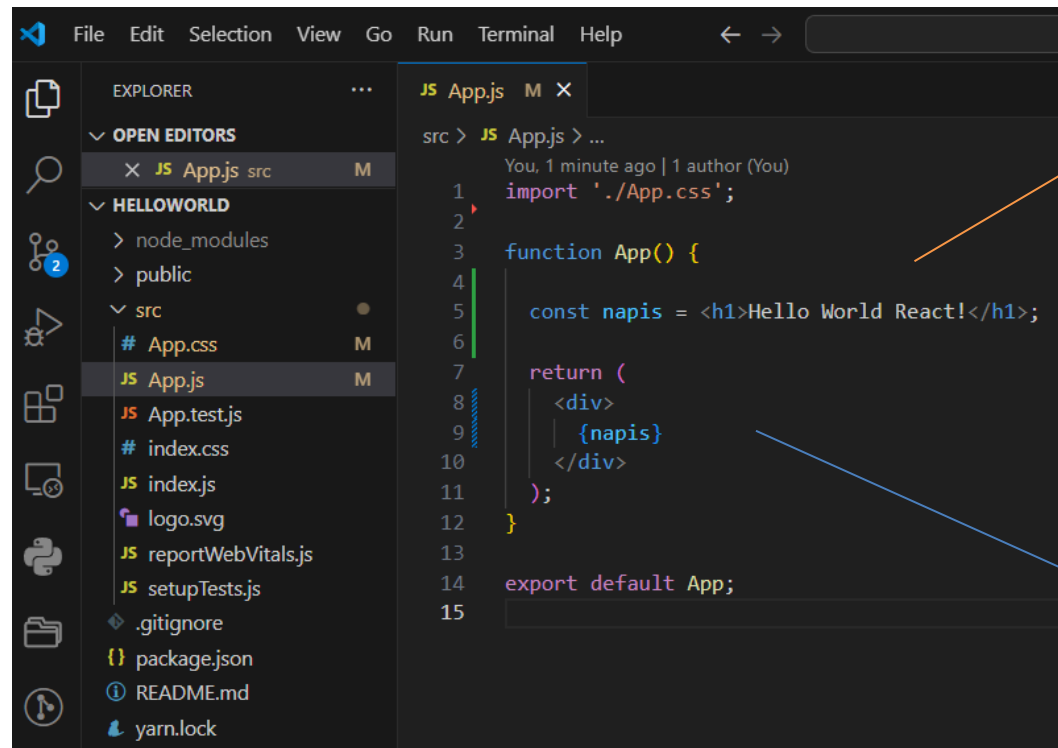
Hello World React!

React JSX

JavaScript XML

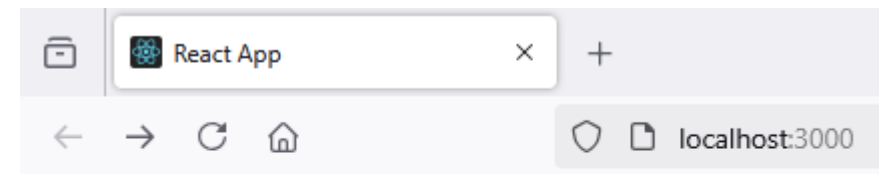
JSX umożliwia pisanie kodu HTML w React – jest to rozszerzona składnia języka JavaScript o możliwość wstawiania znaczników (React konwertuje tagi HTML na elementy Reactowe)

można nie używać składni JSX w React, ale składnia jest wtedy nieco bardziej skomplikowana



```
src > JS App.js > ...
You, 1 minute ago | 1 author (You)
1 import './App.css';
2
3 function App() {
4
5   const napis = <h1>Hello World React!</h1>;
6
7   return (
8     <div>
9       {napis}
10    </div>
11  );
12
13
14 export default App;
15
```

JSX umożliwia przypisanie elementów HTML do zmiennej (bez cudzysłów)



Hello World React!

React JSX

przekazanie zmiennej name do JSX

```
src > JS App.js > App
You, 58 seconds ago | 1 author (You)
1 import './App.css';
2
3 function App() {
4
5   let name = "React";
6   const napis = <h1>Hello World {name}</h1>;
7
8   return (
9     <div>
10      {napis}
11    </div>
12  );
13 }
14
15 export default App;
16
```

React App

localhost:3000

Hello World React!

React JSX

VS Code Explorer sidebar showing the file structure:

- EXPLORED
- OPEN EDITORS
 - JS App.js src M
- HELLOWORLD
 - node_modules
 - public
 - src
 - App.css M
 - App.js M
 - App.test.js
 - index.css
 - index.js
 - logo.svg
 - reportWebVitals.js
 - setupTests.js
 - .gitignore
 - package.json
 - README.md
 - yarn.lock

VS Code Editor showing the code for `App.js`:

```
src > JS App.js > ...
You, 1 second ago | 1 author (You)
1 import './App.css';
2
3 function App() {
4   const software = {
5     nazwa: "React",
6     wiek: 12
7   };
8 };
9
10 function sprawdzWiek(software){
11   if (software.wiek > 11){
12     return "Masz już ponad 11 lat!";
13   }else{
14     return "Nie masz jeszcze 12 lat!";
15   }
16 }
17
18 const napis = <h1>Hello World {software.nazwa}! {sprawdzWiek(software)}</h1>;
19
20 return (
21   <div>
22     {napis}
23   </div>
24 );
25 }
26
27 export default App;
```

Annotations in Polish:

- definicja obiektu software (points to `const software = { ... }`)
- definicja funkcji sprawdzWiek (points to `function sprawdzWiek(software){ ... }`)
- wstawienie nazwy software oraz funkcji do zmiennej *napis* (points to `{software.nazwa}` and `{sprawdzWiek(software)}` in the JSX element)

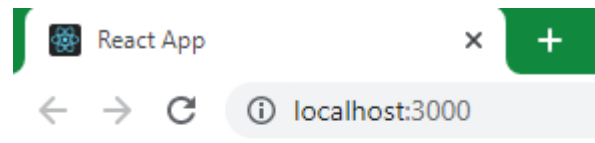
Browser window showing the rendered output:

React App | localhost:3000

Hello World React! Masz już ponad 11 lat!

React JSX

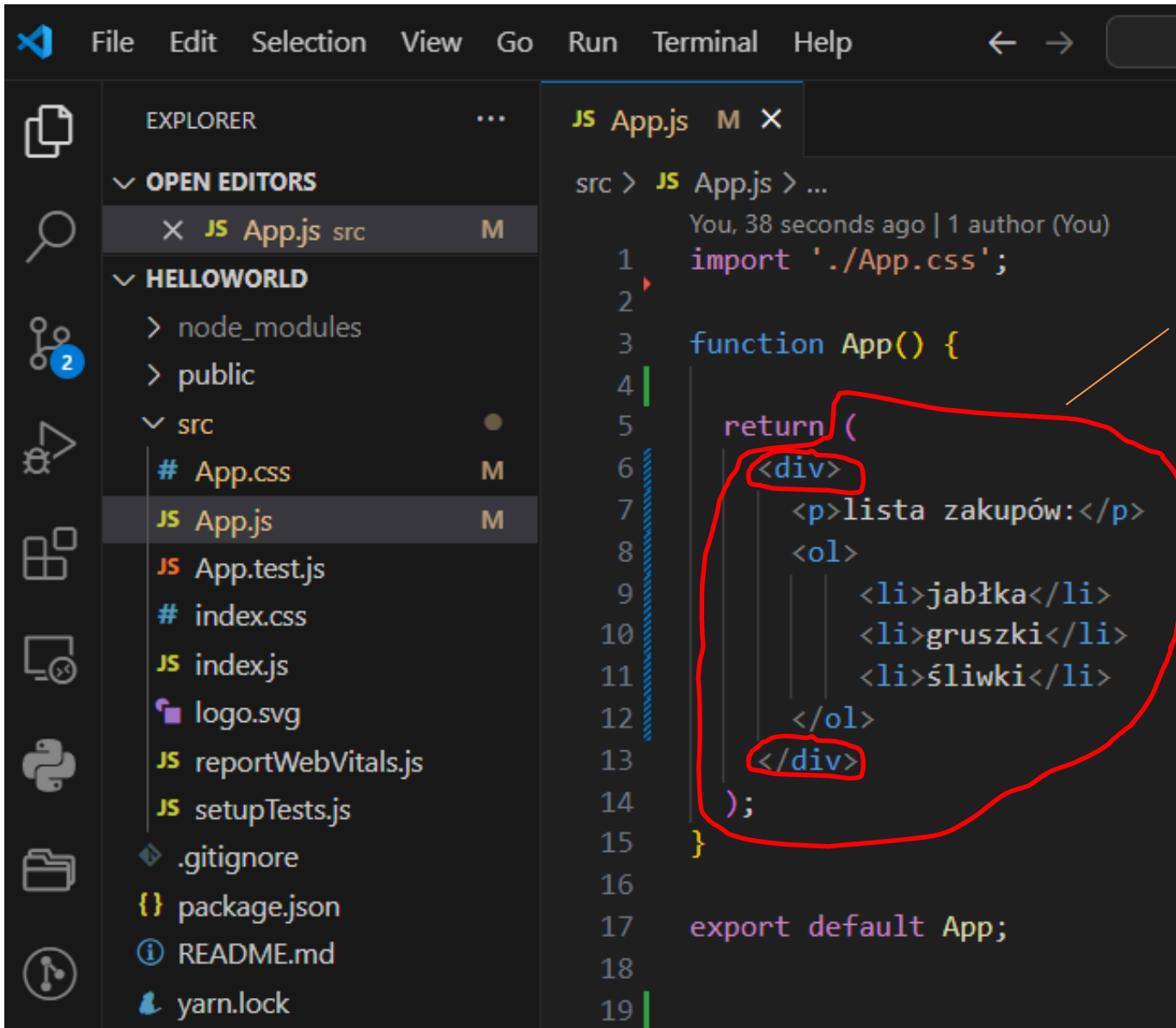
```
src > JS App.js > ...  
You, 1 second ago | 1 author (You)  
1 import './App.css';  
2  
3  
4 function App() {  
5   const a = 15;  
6   const b = 20;  
7  
8   const rownanie = <h1> {a} + {b} = {a + b} </h1>;  
9  
10  return (  
11    <div>  
12      {rownanie}  
13    </div>  
14  );  
15 }  
16  
17 export default App;  
18  
19
```



w JSX można
stosować
wyrażenia

15 + 20 = 35

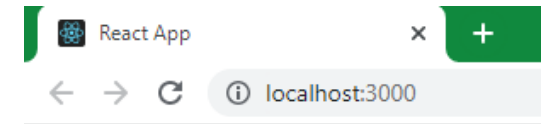
React JSX



```
src > JS App.js > ...
You, 38 seconds ago | 1 author (You)
1 import './App.css';
2
3 function App() {
4
5   return (
6     <div>
7       <p>lista zakupów:</p>
8       <ol>
9         <li>jabłka</li>
10        <li>gruszki</li>
11        <li>śliwki</li>
12      </ol>
13    </div>
14  );
15
16  export default App;
17
18
19
```

kilka elementów
HTML musi się
znaleźć w
pojedynczym
elemente top
level np.
<div></div> (lub
pustym <></>)

top level element
może być pusty



lista zakupów:

1. jabłka
2. gruszki
3. śliwki

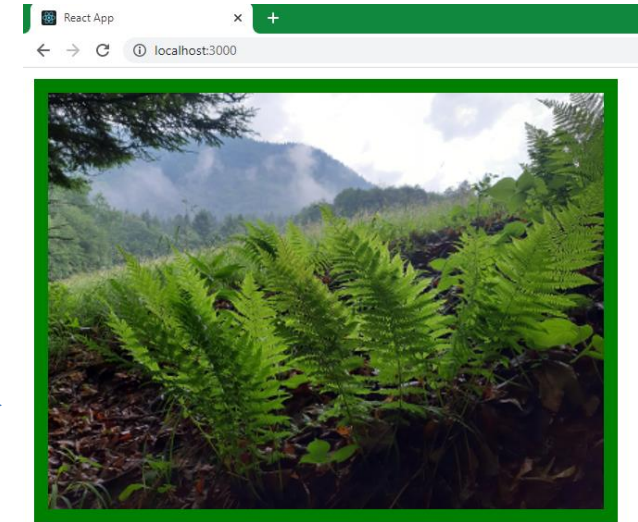
```
function App() {
  return (
    <>
      <p>lista zakupów:</p>
      <ol>
        <li>jabłka</li>
        <li>gruszki</li>
        <li>śliwki</li>
      </ol>
    </>
  );
}
```

React JSX className

```
1 import './App.css';
2 import gory from './pieniny.png';
3
4 function App() {
5
6   return (
7     <div>
8       <img src={gory} className="green" alt="Pieniny" />;
9     </div>
10   );
11 }
12
13 export default App;
```

zamiast
słowa class w
React
stosujemy
className

element bez
tagu
kończącego
musi być
zamknięty



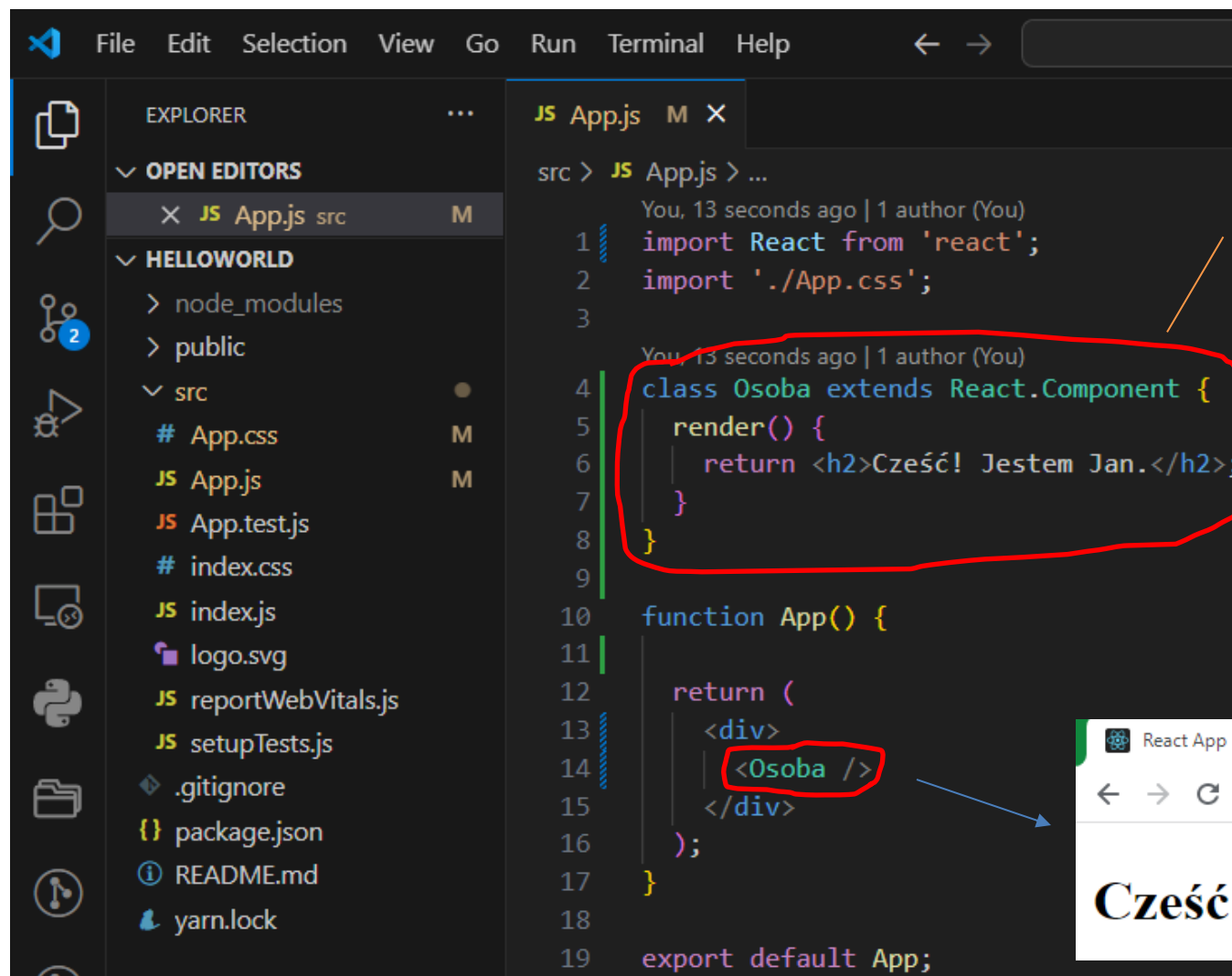
```
1 .green{
2   background-color: green;
3   padding: 10px;
4 }
```

App.css ze stylem klasy green

React Class Component

Komponenty zwracają elementy HTML

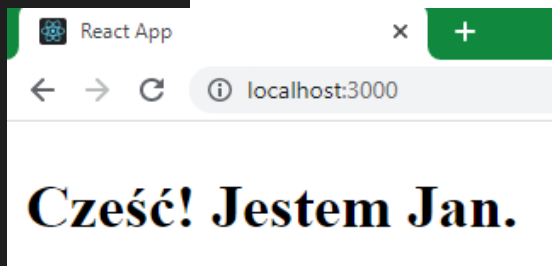
komponent klasowy Osoba



```
src > JS App.js > ...
You, 13 seconds ago | 1 author (You)
1 import React from 'react';
2 import './App.css';
3
4 class Osoba extends React.Component {
5   render() {
6     return <h2>Cześć! Jestem Jan.</h2>;
7   }
8 }
9
10 function App() {
11
12   return (
13     <div>
14       <Osoba />
15     </div>
16   );
17 }
18
19 export default App;
```

klasa Osoba (nazwa z dużej litery!) dziedziczy z klasy React.Component

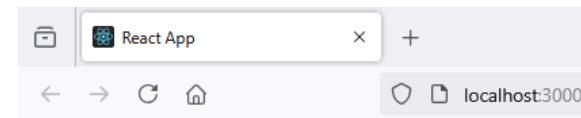
metoda render zwraca kod HTML



React

Function Component

komponent funkcyjny Osoba z
właściwością *imie*



Cześć! Jestem Jan.

Cześć! Jestem Bożena.

Cześć! Jestem Jarek.

Cześć! Jestem Marek.

właściwość (atrybut)
komponentu z
przypisanym imieniem

właściwość komponentu to po
angielsku *properties*, stąd
potoczna nazwa **props**

React

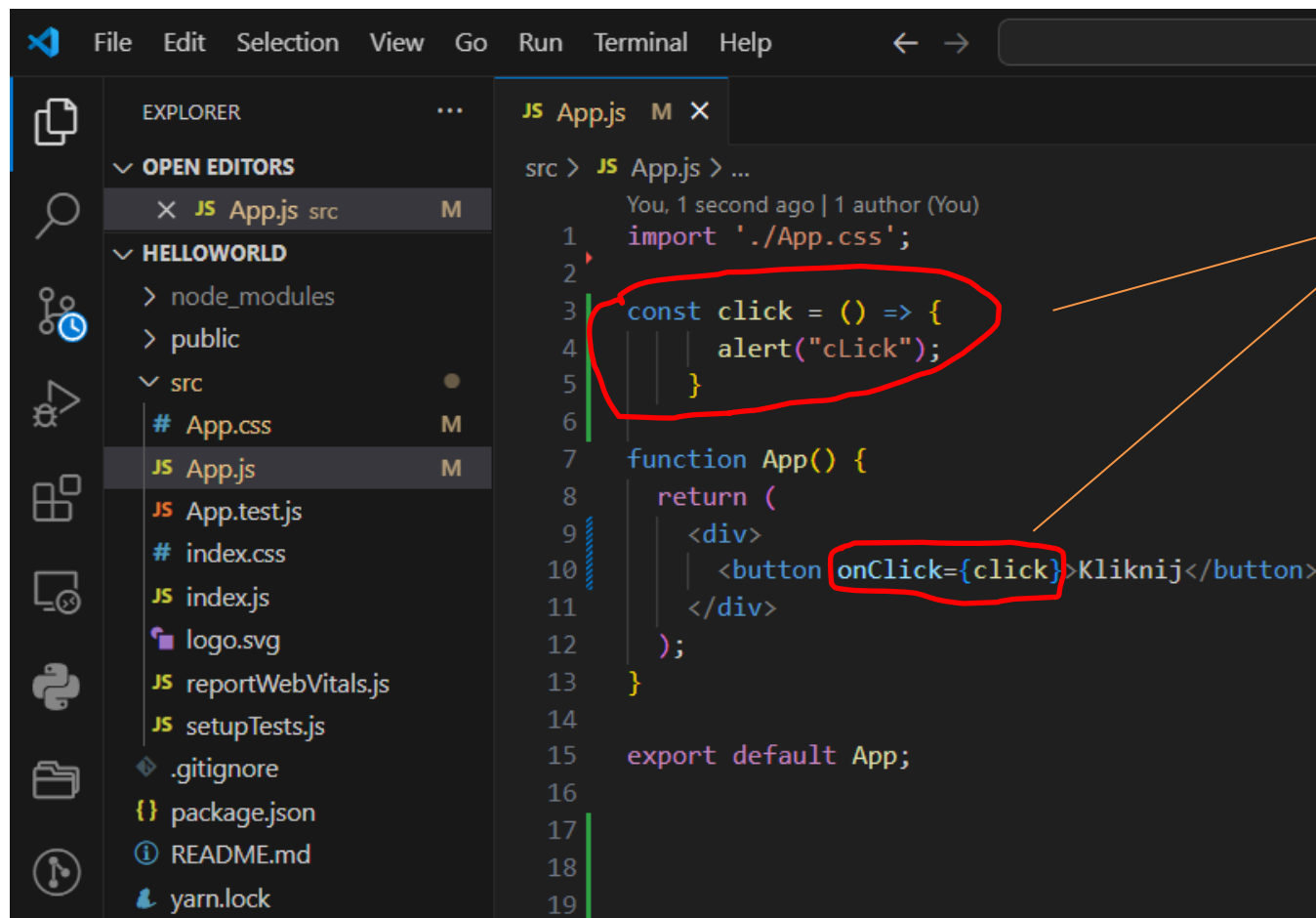
Function Component

komponent funkcyjny *Osoba*
zagnieżdżony w komponencie
funkcyjnym *Firma*

W naszej firmie pracuje:

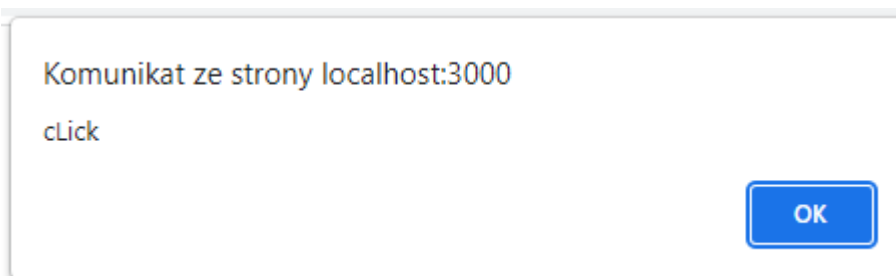
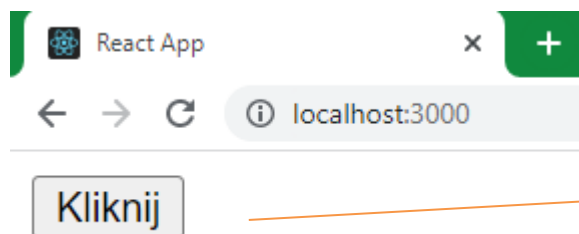
Władysław, w wieku 35 lat.

React events

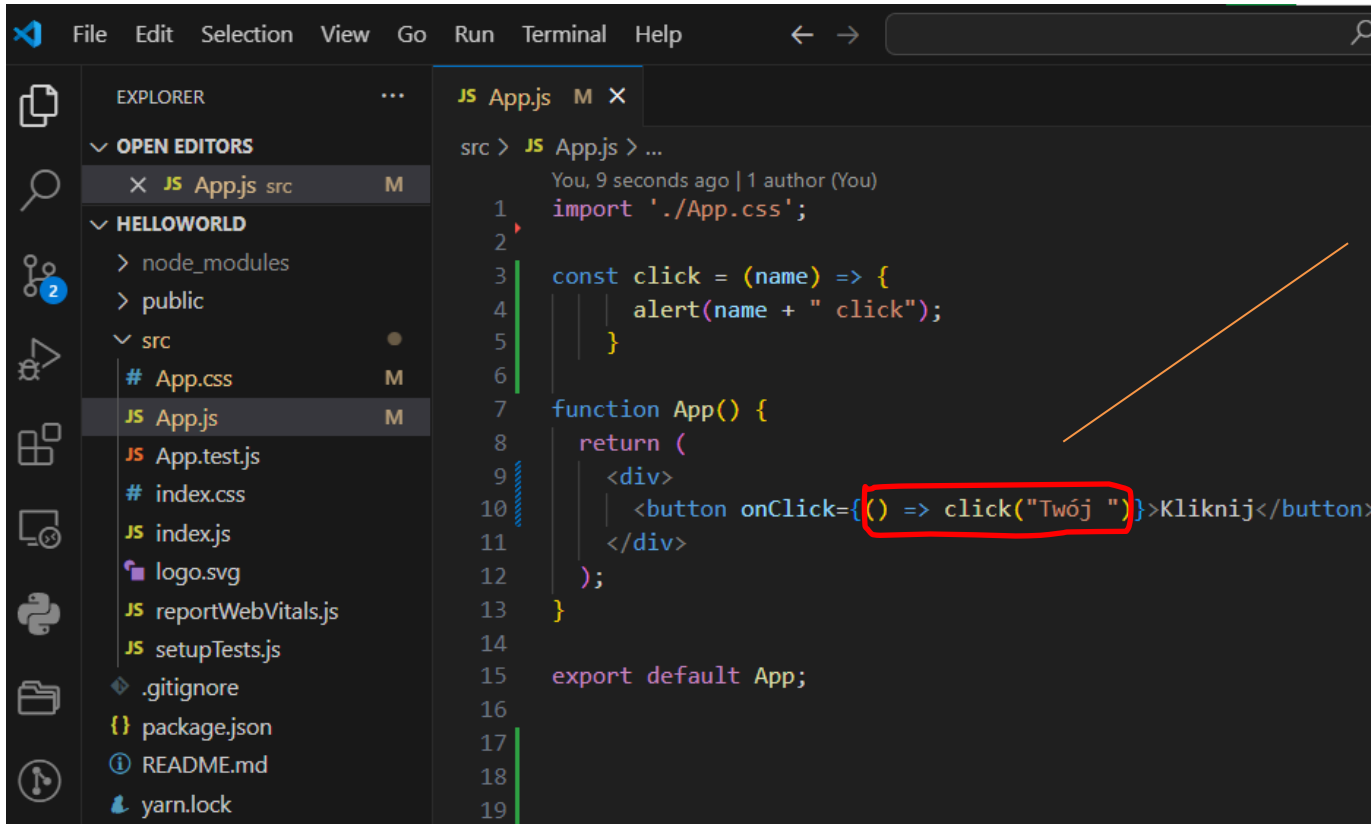


```
src > JS App.js > ...
You, 1 second ago | 1 author (You)
1  import './App.css';
2
3  const click = () => {
4    alert("cLick");
5  }
6
7  function App() {
8    return (
9      <div>
10       <button onClick={click}>Kliknij</button>
11     </div>
12   );
13 }
14
15 export default App;
16
17
18
19
```

przypisanie funkcji *click* do zdarzenia buttona *onClick*

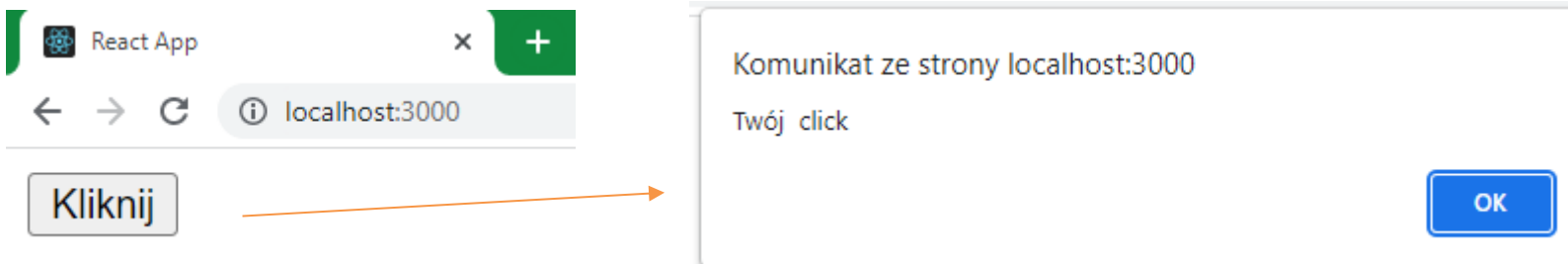


React events

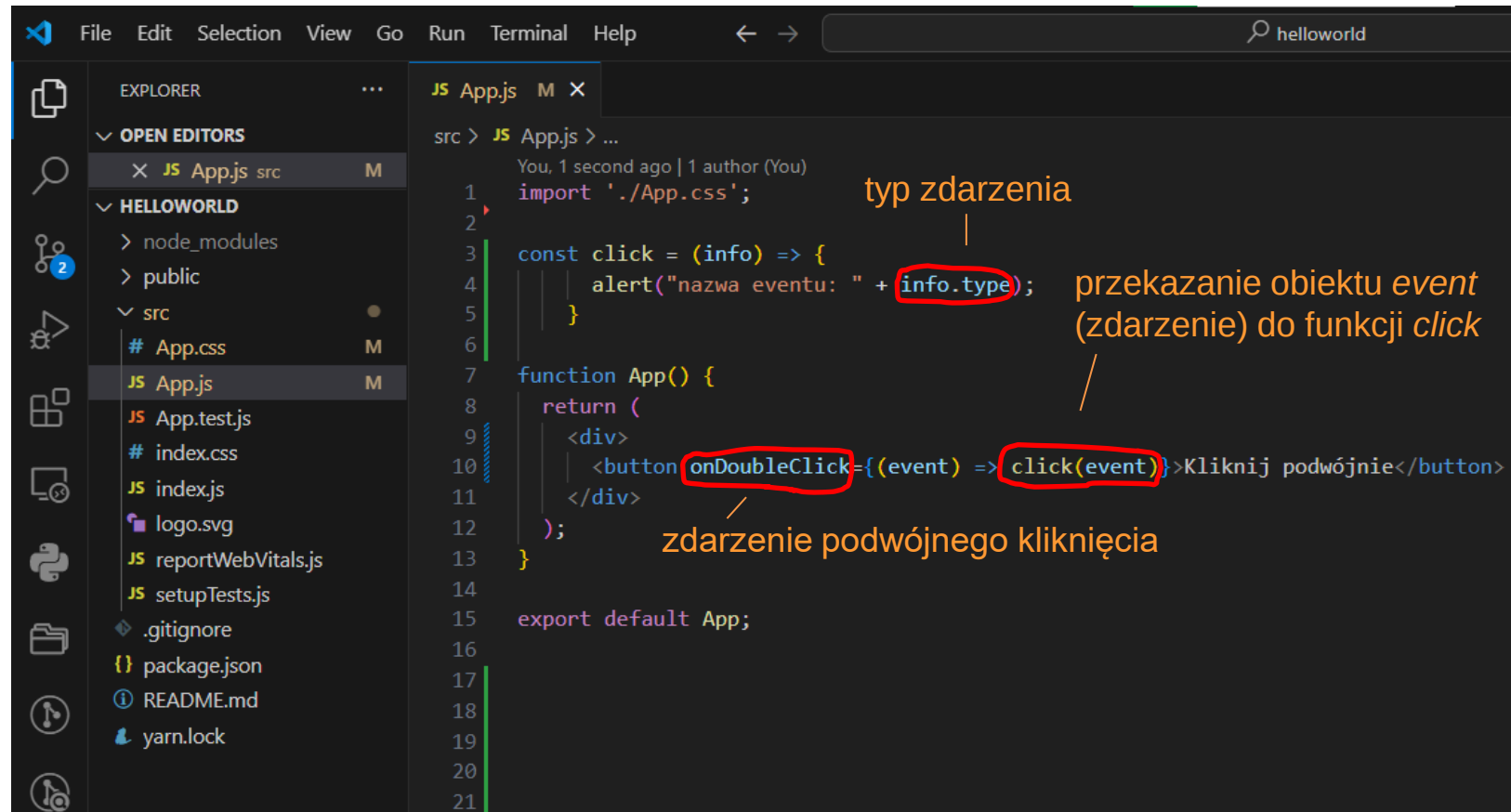


```
src > JS App.js > ...
You, 9 seconds ago | 1 author (You)
1 import './App.css';
2
3 const click = (name) => {
4   alert(name + " click");
5 }
6
7 function App() {
8   return (
9     <div>
10      <button onClick={() => click("Twój ")}>Kliknij</button>
11    </div>
12  );
13 }
14
15 export default App;
16
17
18
19
```

aby przekazać argument do funkcji *click* trzeba użyć funkcji strzałkowej (array function)



React events

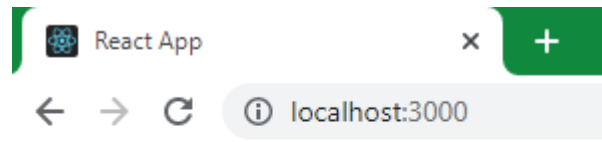


The screenshot shows the VS Code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with a `src` directory containing `App.css`, `App.js`, `App.test.js`, `index.css`, `index.js`, `logo.svg`, `reportWebVitals.js`, `setupTests.js`, `.gitignore`, `package.json`, `README.md`, and `yarn.lock`. The code editor shows the `App.js` file with the following code:

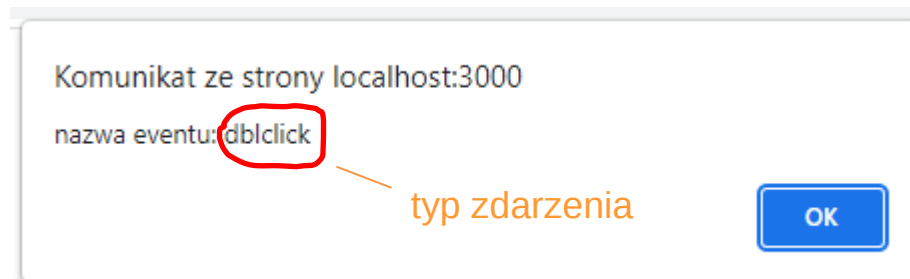
```
1 import './App.css';
2
3 const click = (info) => {
4   alert("nazwa eventu: " + info.type);
5 }
6
7 function App() {
8   return (
9     <div>
10       <button onClick={event => click(event)}>Kliknij podwójnie</button>
11     </div>
12   );
13 }
14
15 export default App;
```

Annotations in Polish explain the code:

- `typ zdarzenia` (event type) points to `info.type` in the `click` function.
- `przekazanie obiektu event (zdarzenie) do funkcji click` (passing the event object to the `click` function) points to `click(event)` in the `onClick` prop.
- `zdarzenie podwójnego kliknięcia` (double-click event) points to `onDoubleClick` in the `onClick` prop.



Kliknij podwójnie



typ zdarzenia

OK

useState Hook

useState is a React Hook that lets you add a state variable to your component



useState

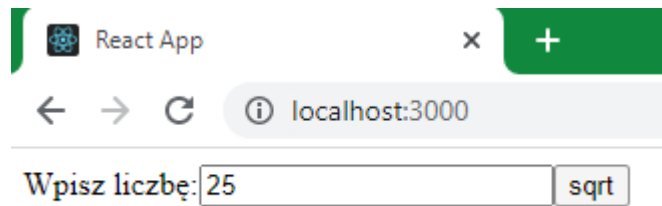
State generally refers to data or properties that need to be tracking in an application.

<https://react.dev/reference/react/useState>

https://www.w3schools.com/REACT/react_usestate.asp

React form

komponent funkcyjny
obsługujący formularz ma swój
stan, w którym przechowuje
dane (wartość z pola input)



React App

localhost:3000

Wpisz liczbę: 25 sqrt

Komunikat ze strony localhost:3000
Pierwiastek kwadratowy z 25 to 5

OK

React form

```
1 import './App.css';
2 import { useState } from 'react';
3
4 function App() {
5   const [number, setNumber] = useState("");
6
7   const handleSubmit = (event) => {
8     event.preventDefault();
9     let a = Math.sqrt(number);
10    alert(`Pierwiastek kwadratowy z ${number} to ${a}`);
11  }
12
13  return (
14    <form onSubmit={handleSubmit}>
15      <label>Wpisz liczbę:
16        <input
17          type="text"
18          value={number}
19          onChange={(e) => setNumber(e.target.value)}
20        />
21      </label>
22      <input type="submit" value="sqrt" />
23    </form>
24  )
25 }
26
27 export default App;
```

import useState Hook

funkcja zmieniająca
stan komponentu

stan komponentu
(pole number)

wartość początkowa
stanu komponentu

formularz

zdarzenie *onChange*
powoduje zapisanie
wartości pola *input* w
stanie komponentu
(polu number)

React form

Ankieta

imię

nazwisko

pleć

mężczyzna ☐ kobieta ☐

zainteresowania

sport ☐ film ☐ elektronika ☐ informatyka ☐

szkoła

Podstawowa

▼

krótko o sobie

///

wyświetl dane

resetuj ankietę

napisz ankietę

wpisane dane
wyświetl w dowolny
sposób

useEffect Hook

useEffect is a React Hook that lets you synchronize a component with an external system.



useEffect



Effects are an escape hatch from the React paradigm. They let you “step outside” of React and synchronize your components with some external system like a non-React widget, network, or the browser DOM

<https://react.dev/reference/react/useEffect>

useEffect Hook

useEffect(setup, dependencies?)



setup: The function with your Effect's logic. Your setup function may also optionally return a cleanup function. When your component is added to the DOM, React will run your setup function. After every re-render with changed dependencies, React will first run the cleanup function (if you provided it) with the old values, and then run your setup function with the new values. After your component is removed from the DOM, React will run your cleanup function.



optional dependencies: The list of all reactive values referenced inside of the setup code. Reactive values include props, state, and all the variables and functions declared directly inside your component body.

useEffect Hook

1. Running the Effect:

- The function passed to `useEffect` runs after every render by default.
- By specifying a second argument, an array of dependencies, you can control when the effect runs.
 - `[]`: the effect runs only once after the initial render (similar to `componentDidMount`).
 - `[dependency]`: the effect runs whenever the specified dependency changes.
 - No array: the effect runs after every render.

2. Cleanup:

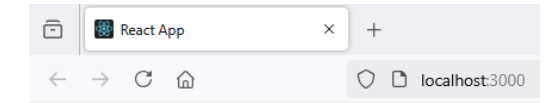
- If your effect returns a function, that function will be called when the component unmounts or before the effect runs next. This is used to clean up subscriptions, timers, etc.

3. Multiple Effects:

- You can have multiple `useEffect` hooks in a single component, which allows you to separate different concerns.

prognoza pogody

```
src > JS App.js > ...
You, 49 seconds ago | 1 author (You)
1  import React, { useState, useEffect } from 'react';
2
3  function App() {
4
5      const [data, setData] = useState(null);
6      const [loading, setLoading] = useState(true);
7
8      useEffect(() => {
9          const fetchData = async () => {
10              const response = await fetch('https://api.open-meteo.com/v1/forecast?latitude=51.747936&longitude=19.44281');
11              const result = await response.json();
12              setData(result);
13              setLoading(false);
14          };
15
16          fetchData();
17
18          // Optional cleanup
19          return () => {
20              console.log("Cleanup if necessary");
21          };
22      }, []); // Empty array means it runs once at mount
23
24      if (loading) return <div>Loading...</div>;
25
26      return (
27          <div>
28              <h1>Prognoza pogody</h1>
29              <pre>{JSON.stringify(data, null, 2)}</pre>
30          </div>
31      );
32  }
33
```



Prognoza pogody

```
{
  "latitude": 51.747936,
  "longitude": 19.44281,
  "generationtime_ms": 0.02574920654296875,
  "utc_offset_seconds": 3600,
  "timezone": "Europe/Berlin",
  "timezone_abbreviation": "GMT+1",
  "elevation": 208,
  "hourly_units": {
    "time": "iso8601",
    "temperature_2m": "°C"
  },
  "hourly": {
    "time": [
      "2025-02-25T00:00",
      "2025-02-25T01:00",
      "2025-02-25T02:00",
      "2025-02-25T03:00",
      "2025-02-25T04:00",
      "2025-02-25T05:00",
      "2025-02-25T06:00",
      "2025-02-25T07:00",
      "2025-02-25T08:00",
      "2025-02-25T09:00",
      "2025-02-25T10:00",
      "2025-02-25T11:00",
      "2025-02-25T12:00",
      "2025-02-25T13:00",
      "2025-02-25T14:00",
      "2025-02-25T15:00",
      "2025-02-25T16:00",
      "2025-02-25T17:00",
      "2025-02-25T18:00",
      "2025-02-25T19:00",
      "2025-02-25T20:00",
      "2025-02-25T21:00",
      "2025-02-25T22:00",
      "2025-02-25T23:00"
    ],
    "temperature_2m": [
      3.9,
      3.8,
      3.6,
      2.4,
      1.2,
      0.7,
      0.8,
      1.1,
      1.4,
      2.2,
      3.3,
      4.6,
      6.1,
      7.5,
      8.4,
      8.9,
      9.1,
      8.9,
      7.9,
      7.3,
      6.7,
      6.7,
      6.5,
      5.9
    ]
  }
}
```

https://www.google.pl/maps/place/Stefana+Żeromskiego+115,+90-001+Łódź/@51.7553166,19.4471638,9z/data=!4m6!3m5!1s0x471a35280bdd169d:0xfb8e92f27cb3f0e5!8m2!3d51.7557629!4d19.4482762

Stefana Żeromskiego 115

Restauracje Hotele Atrakcje Transport publiczny Parkingi Apteki Bankomaty

Zapisane Ostatnie Czajplinek & Łódź

Stefana Żeromskiego 115
Budynek

Wyznacz trasę Zapisz W pobliżu Wyślij na telefon Udostępnij

Stefana Żeromskiego 115, 90-001 Łódź

Zaproponuj zmianę dotyczącą: Stefana Żeromskiego 115

Dodaj brakujące miejsce

Dodaj swoją firmę

Zdjęcia

Katalog

Wyszukaj miejsca

Usługi 2 Inne 15

Haft Tęcza

Towarzystwo Przyjaciół Łodzi

Centrum Kształcenia Zawodowego i...

Branżowa Szkoła II Stopnia po BS1st...

Parking

Stefana Żeromskiego

Franciszka Żwirki

Kpt. Francisz

Fontanna w Parku Poniatońskiego

Google

Dane mapy ©2025 Google Polska Warunki Prywatność Prześlij opinię o produkcie 20 m

open-meteo.com

prognoza pogody

Open-Meteo Home Features Pricing API Docs

Weather Forecast API
Seamless integration of high-resolution weather models with up to 16 days forecast

szerokość i długość geograficzna CKZiU

Weather Forecast

Location and Time

Location: **Coordinates** **List**

Latitude: 51.7551091 Longitude: 19.4465375 Timezone: Europe/Berlin

Search +

Time: **Forecast Length** **Time Interval**

Forecast days: 1 day Past days: 0 (default)

By default, we provide forecasts for 7 days, but you can access forecasts for up to 16 days. If you're interested in past weather data, you can use the **Past Days** feature to access archived forecasts.

Hourly Weather Variables

☒ Temperature (2 m) ☐ Weather code ☐ Wind Speed (10 m) ☐ Soil Temperature (0 cm)

Download XLSX Download CSV

<https://open-meteo.com/en/docs>

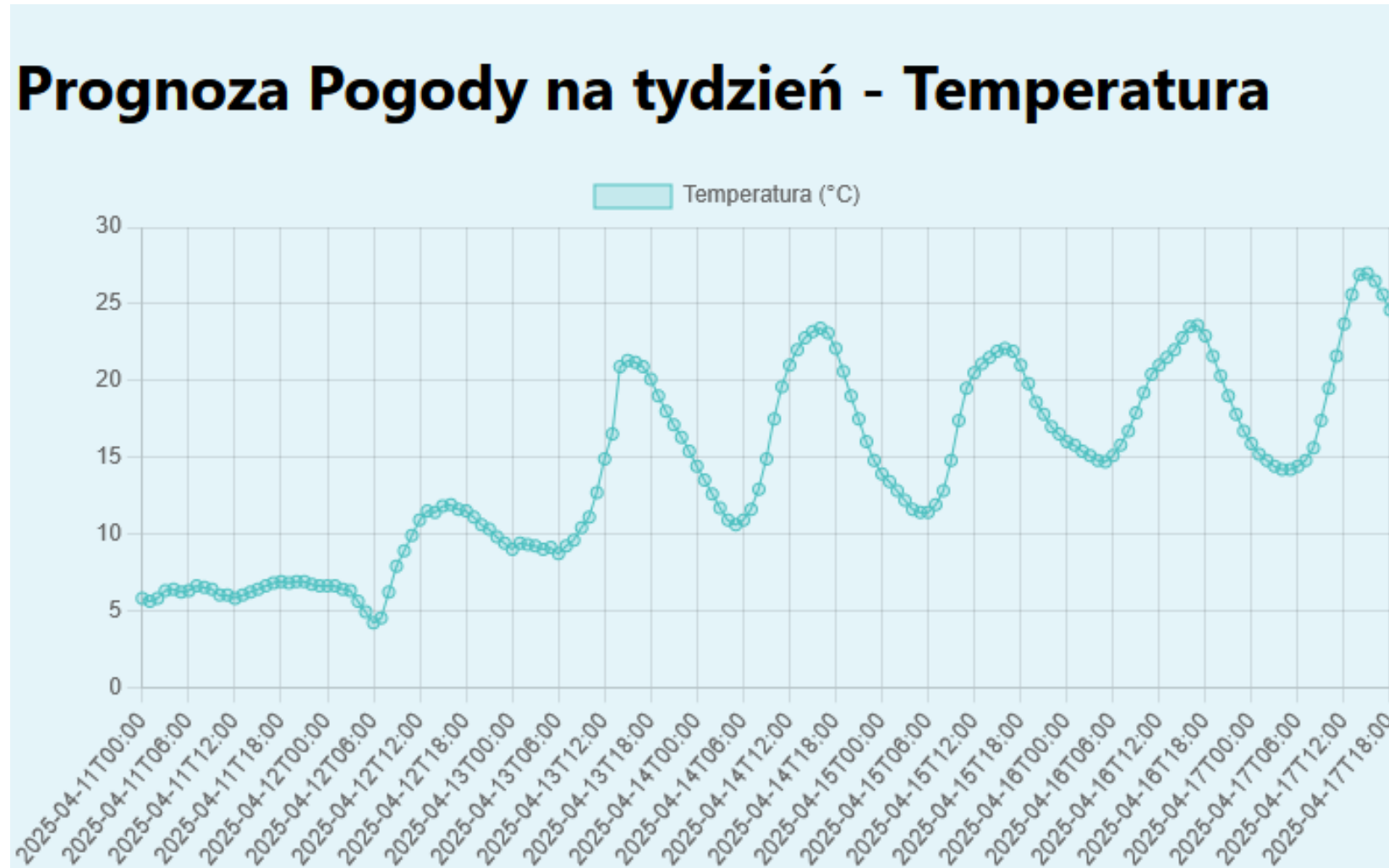
API URL

API URL ([Open in new tab](#) or copy this URL into your application)

https://api.open-meteo.com/v1/forecast?latitude=51.7551091&longitude=19.4465375&hourly=temperature_2m&timezone=Europe%2FBerlin&forecast_days=1

prognoza pogody

korzystając z [bibliotek graficznych w js](#) zaprezentuj
przesłane dane z open-meteo.com w postaci wykresu



useRef Hook

The useRef Hook allows you to persist values between renders.



useRef

It can be used to store a mutable value that does not cause a re-render when updated.

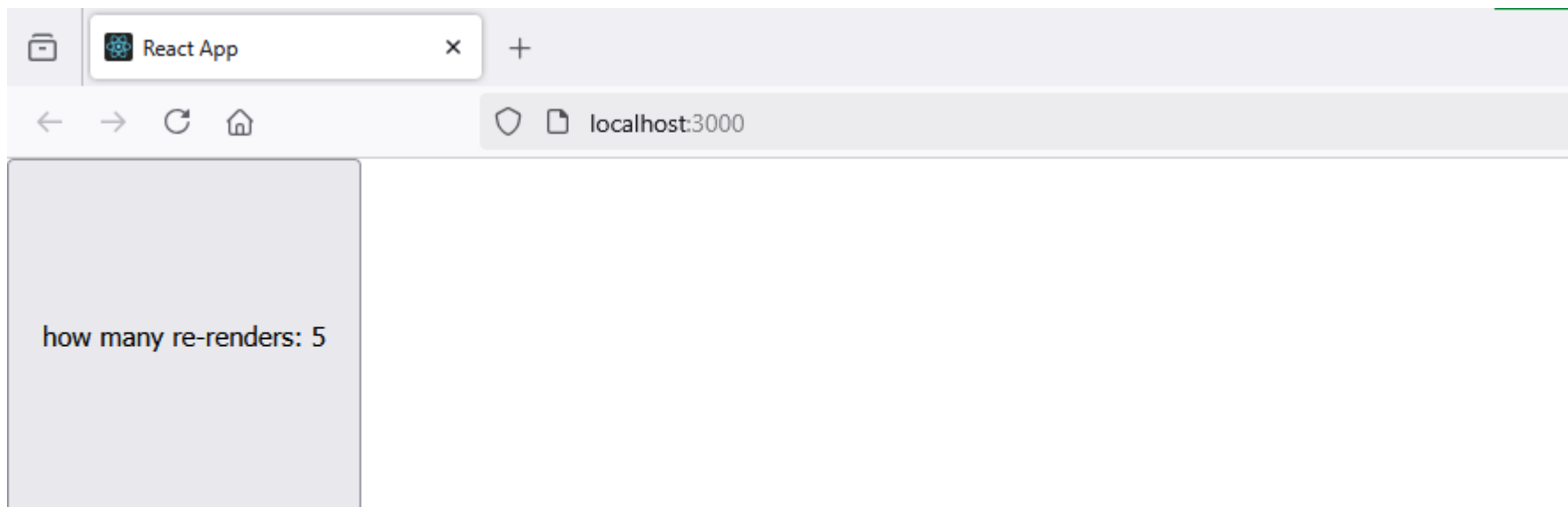
It can be used to access a DOM element directly.

https://www.w3schools.com/react/react_useRef.asp

<https://react.dev/reference/react/useRef>

useRef - zapisywanie wartości pomiędzy renderowaniami

liczbę renderowań
liczymy
korzystając z
useRef



Po kliknięciu zmienia się liczba losowa (zmienna stanu) 0.41628540431840333, co powoduje re-render komponentu

liczba renderowań

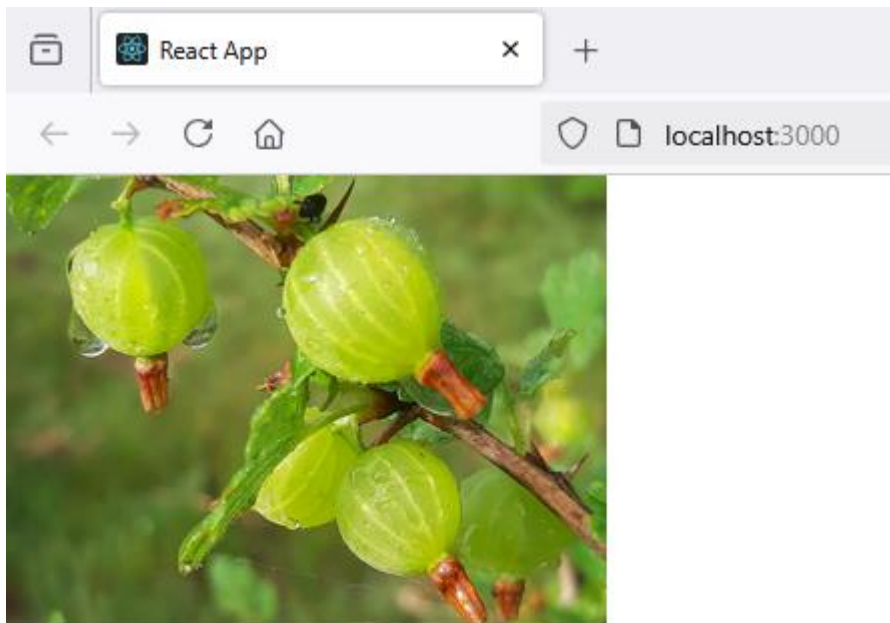
The screenshot shows a VS Code editor with a project named 'useeffecthookcounter'. The file explorer on the left shows the project structure, including 'src' and 'App.js'. The main editor displays the code for 'App.js'.

```
1 import './App.css';
2 import {useEffect, useState, useRef} from 'react'
3
4 function App() {
5   const [randomValue, setRandomValue] = useState(0);
6   const count = useRef(0); // liczbę renderowań liczymy korzystając z useRef
7
8   useEffect(() => {
9     count.current = count.current + 1;
10  }); // uruchamia się za każdym razem gdy jest re-render (gdy zmieni się zmienna stanu randomValue)
11
12  return (
13    <>
14      { /* po kliknięciu zmienia się wartość zmiennej stanu randomValue */ }
15      <button id="counter-button" onClick={(e) => setRandomValue(Math.random())}>
16        how many re-renders: {count.current}
17      </button>
18      <h2>Po kliknięciu zmienia się liczba losowa (zmienna stanu) {randomValue},<br />
19      co powoduje re-render komponentu</h2>
20    </>
21  );
22 }
23
24
25 export default App;
```

Annotations in orange text explain the code:

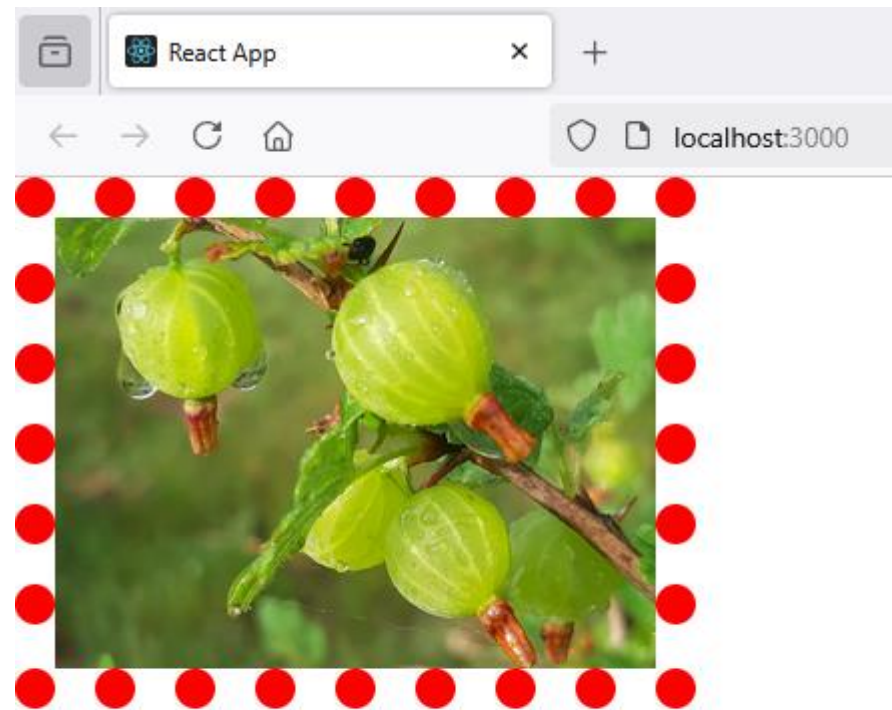
- użycie useRef do zainicjalizowania zmiennej count, przeznaczonej na liczbę renderowań komponentu (points to line 6)
- inkrementacja count po każdym prerenderowaniu komponentu (points to line 9)
- po kliknięciu na przycisk zmieniamy zmienną stanu randomValue, co powoduje prerenderowanie komponentu (points to line 15)

useRef – przypisanie referencji do elementu DOM



dodaj ramkę

zdjęcie ma atrybut
ref dzięki czemu
możemy mu
przypisać styl



dodaj ramkę

klikając na napis
dodajemy do
zdjęcia ramkę

ramka

The image shows a VS Code editor with a project named 'USEREFHOOKHTMLDOMPICTURE'. The file explorer on the left shows the project structure, including a 'src' directory with files like 'agrest.jpg', 'App.css', 'App.js', 'App.test.js', 'index.css', 'index.js', 'logo.svg', 'reportWebVitals.js', and 'setupTests.js'. The code editor on the right shows the content of 'App.js'.

```
1 import agrest from './agrest.jpg';
2 import './App.css';
3 import {useRef} from 'react'
4
5 function App() {
6
7   const pictureRef = useRef(null) //referencja do DOM element <img>
8
9   function handleClick() {
10     pictureRef.current.style.border= '20px dotted red';
11   }
12
13   return (
14     <div className="App">
15       <header className="App-header">
16         <img src={agrest} ref={pictureRef} width="300px" alt="agrest" />
17       </header>
18       <h1 onClick={handleClick}>dodaj ramkę</h1>
19     </div>
20   );
21 }
22
23 export default App;
```

Annotations in orange text explain the code:

- deklaracja referencji do elementu DOM (points to line 7)
- zmiana stylu elementu DOM (points to line 10)
- przypisanie referencji do elementu DOM (points to line 16)

formularz

Menu

1. ziemniaki, schabowy, kapusta
2. pomidorowa
3. rosół

podaj numer potrawy:



 Inspektor  **Konsola**  Debugger  Sieć

 Filtruj zawartość

wybrana potrawa: ziemniaki, schabowy, kapusta

>>

formularz

```
1 import { useRef } from 'react';
2
3 function App() {
4   // tablica z menu
5   const menu = ['ziemniaki', 'schabowy', 'kapusta', 'pomidorowa', 'rosół'];
6
7   // referencje do input
8   const nrPotrawyRef = useRef("");
9
10  let wybranaPotrawa = "";
11  let nrPotrawy = 0;
12
13  const handleSubmit = (event) => {
14    event.preventDefault();
15    nrPotrawy = nrPotrawyRef.current.value;
16    wybranaPotrawa = menu[nrPotrawy - 1];
17    console.log("wybrana potrawa: " + wybranaPotrawa);
18  }
19
20  return (
21    <div>
22      <h2>Menu</h2>
23      <ol>
24        {menu.map((nazwa) => <li> {nazwa}</li> )}
25      </ol>
26      <form onSubmit={handleSubmit}>
27        <div>
28          <label htmlFor='nrPotrawy'>podaj numer potrawy:</label><br />
29          <input
30            id="nrPotrawy"
31            type="text"
32            ref={nrPotrawyRef}
33          />
34        </div>
35        <input type="submit" value="Zamów" />
36      </form>
37    </div>
38  );
39 }
40
41 export default App;
```

deklaracja referencji do
elementu <input>

pobranie wartości z pola
tekstowego formularza po
naciśnięciu przycisku
„Zamów”

przypisanie referencji do
elementu <input>

clicker

napisz clickera,
który będzie
zmieniał prędkość
kręcenia się logo

czas animacji skracamy o 0.1 sekundy po każdym kliknięciu

9.7000000000000001s

WTF!!!

!?

Change spin

clicker



useContext Hook

React Context is a way to manage state globally.



A diagram consisting of two thin blue lines. One line starts from the text 'React Context is a way to manage state globally.' and points diagonally down and to the right towards the word 'useContext'. The second line starts from the word 'useContext' and points diagonally down and to the left towards the text 'It can be used together with the useState Hook to share state between deeply nested components more easily than with useState alone.'

useContext

It can be used together with the useState Hook to share state between deeply nested components more easily than with useState alone.

https://www.w3schools.com/REACT/react_usecontext.asp

<https://react.dev/reference/react/useContext>

useContext Hook

po zapoznaniu się z przykładem
na w3schools: [useContext](#)
napisz dowolną aplikację
korzystającą z useContext Hook

porównanie różnych frameworków do budowy aplikacji webowych



TodoMVC

Helping you **select** an MV* framework

[Download](#)[View on GitHub](#)[Blog](#)



Introduction

Developers have a number of choices today when it comes to selecting a JavaScript framework or UI library for building scalable web apps.

React / Next.js, Vue / Nuxt, Angular... the list of solutions continues to grow, but just how do you decide on which to use in a sea of so many options?

To help you understand the options, we created TodoMVC - a project which has offered the same Todo applications implemented in popular JavaScript frameworks for the last decade.

TodoMVC is useful for comparing syntax and solutions, is officially used in cross-browser benchmarks (e.g. [Speedometer](#)) and aims to stay up to date as trends change over time.

[Follow](#)[Post](#)

Examples

JavaScriptCompile-to-JSLabs

These are examples written in pure JavaScript.

React New	React Redux New	Vue.js New
Preact New	Backbone.js New	Angular New
Ember.js New	Lit New	KnockoutJS
Dojo	Knockback.js	CanJS
Polymer	Mithril	Marionette.js

Compare these to a non-framework implementation

JavaScript ES5 New	JavaScript ES6 New	jQuery New
---------------------------	---------------------------	-------------------

<https://todomvc.com/>

How to deploy a React App on Apache web server

Firstly, in your react project go to your **package.json** file and adjust this line of code to match your destination domain address + folder:

```
"homepage": "https://yourwebsite.com/your_folder_name/",
```

Secondly, go to **terminal** in your react project and type:

```
npm run build
```



Now, **take all files** from that newly created **build** folder and upload them into **your_folder_name**, with filezilla in subfolder like this:

```
public_html/your_folder_name
```

Check in the browser!

