# Section 1: Let's Dive In!

## How to Get Help

## Join Our Community!

## Course Resources

## Let's Build an App!

## Critical Questions

## Quiz 1: Core Concepts

## A Few More Critical Questions

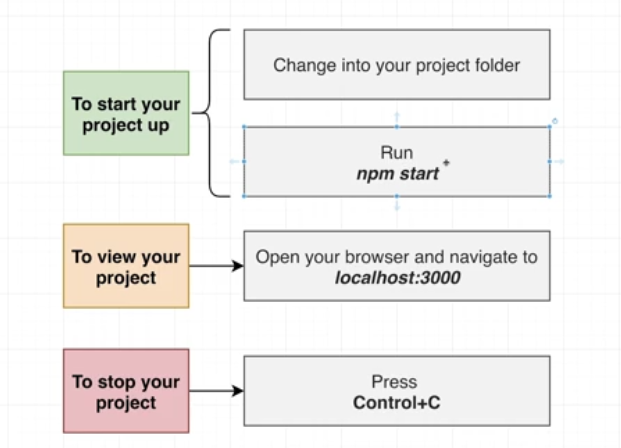
## Node Setup

npx create-react-app <project name>

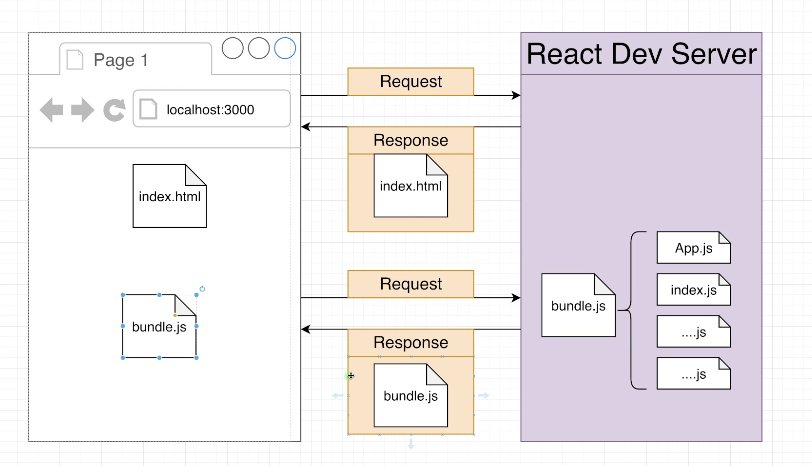
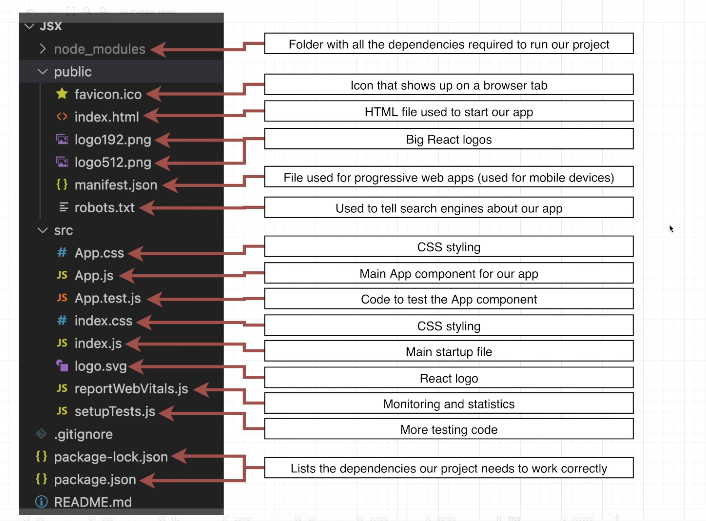
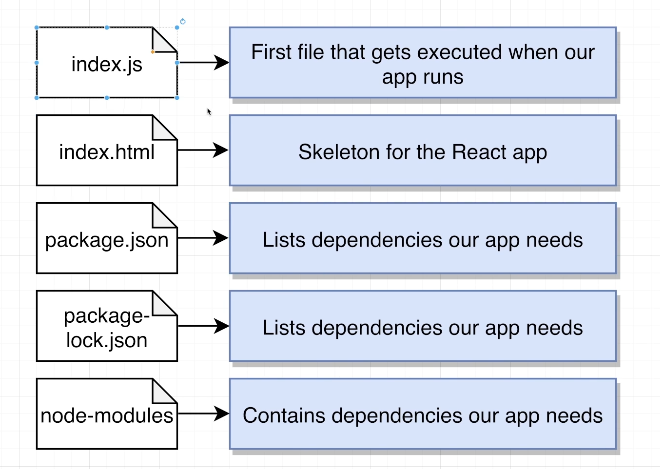
npx create-react-app jsx

cd jsx

npm start



## What is Create React App?

# Section 2: Creating Content with JSX

## Showing Basic Content

// 1) Import the React and ReactDOM libraries

import React from 'react';

import ReactDOM from 'react-dom/client';

// 2) Get a reference to the div with ID root

const el = document.getElementById('root');

// 3) Tell React to take control of that element

const root = ReactDOM.createRoot(el);

// 4) Create a component

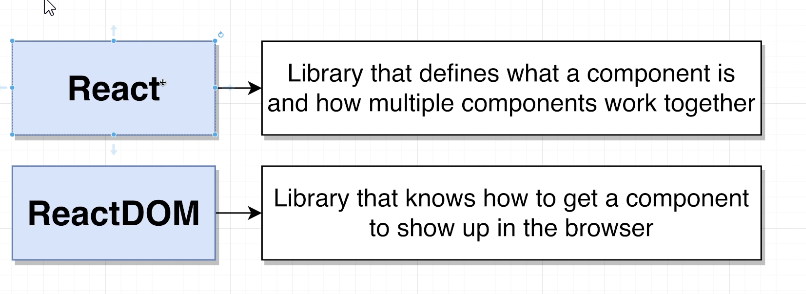
function App() {

  return <h1>Hi there!</h1>;

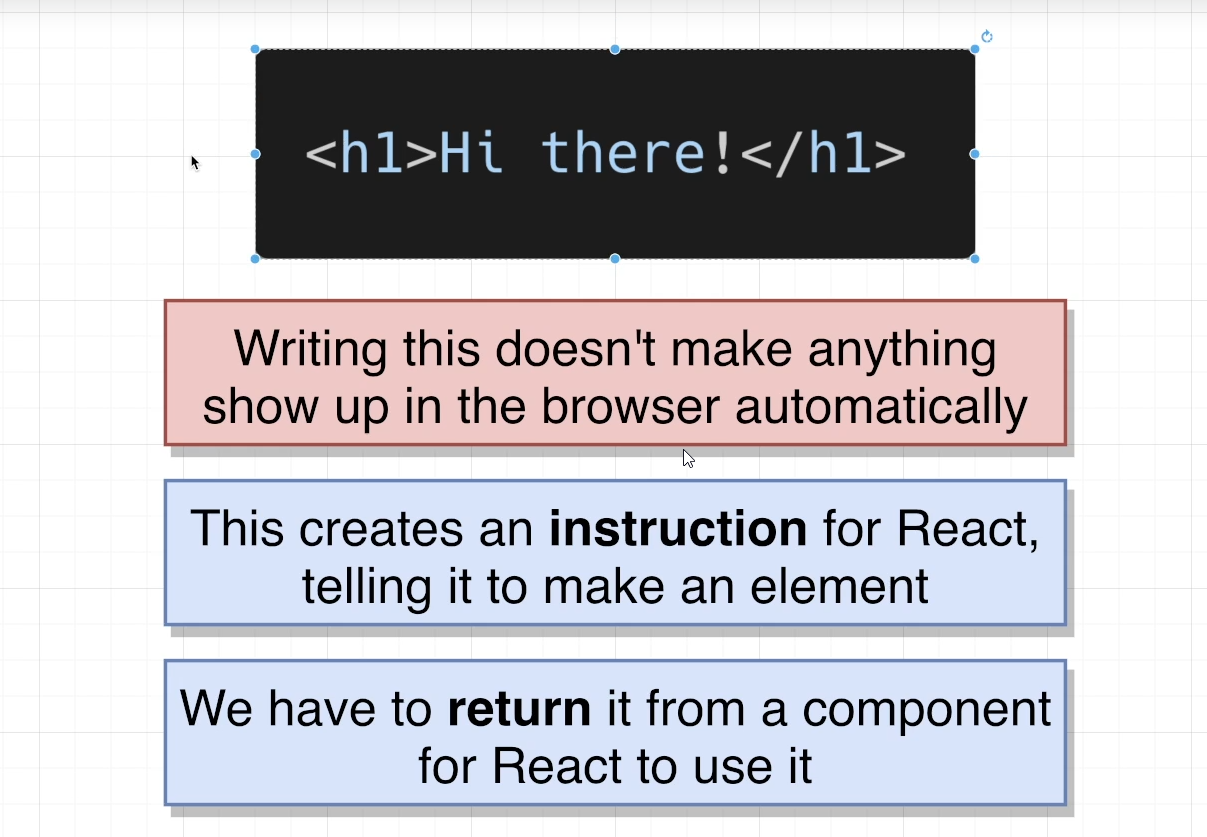
}

// 5) Show the component on the screen

root.render(<App />);

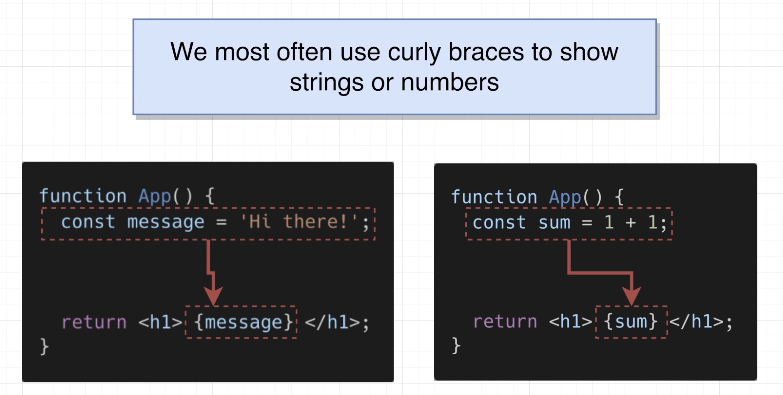
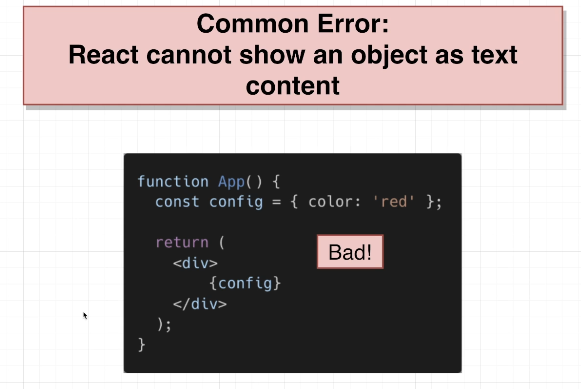


## What is JSX?



## Printing JavaScript Variables in JSX

1. function App() {
2. let message = 'Bye there';
3. if(Math.random() > 0.5) {
4. message = 'Hello there!';
5. }
6. return <h1>{message}</h1>;
7. }

## Shorthand JS Expressions

1. function App() {
2. const date = new Date();
3. const time = date.toLocaleTimeString();
4. return <h1>{time}</h1>;
5. }

function App() {

  return <h1>{new Date().toLocaleTimeString()}</h1>;

}

## Quiz 2: Showing Javascript Values in JSX

## 14. Exercise Overview

## Coding Exercise 1: Quick Practice with JSX

import React from 'react';

function App() {

const name = 'Mehmet';

return (

<div>

My name is:

<h1>{name}</h1>

</div>

);

}

export default App;

## 15. Exercise Solution

## 16. Typical Component Layouts

function App() {

  const name = 'Samantha';

  const age = 23;

  return (

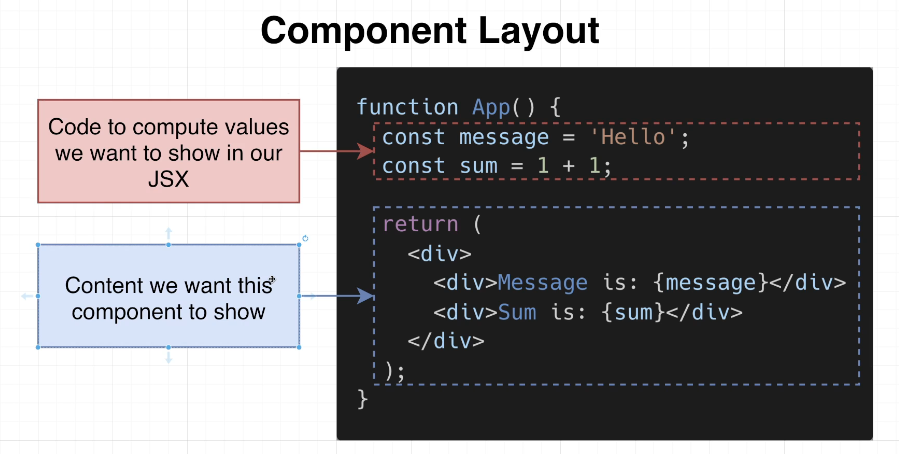
    <h1>

      Hi, my name is {name} and my age is {age}

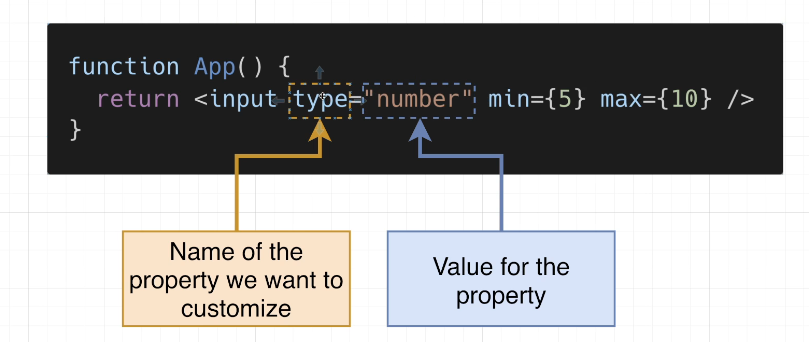
    </h1>

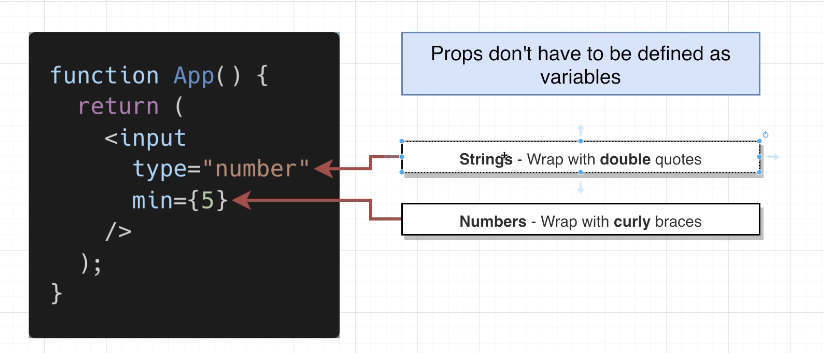
  );

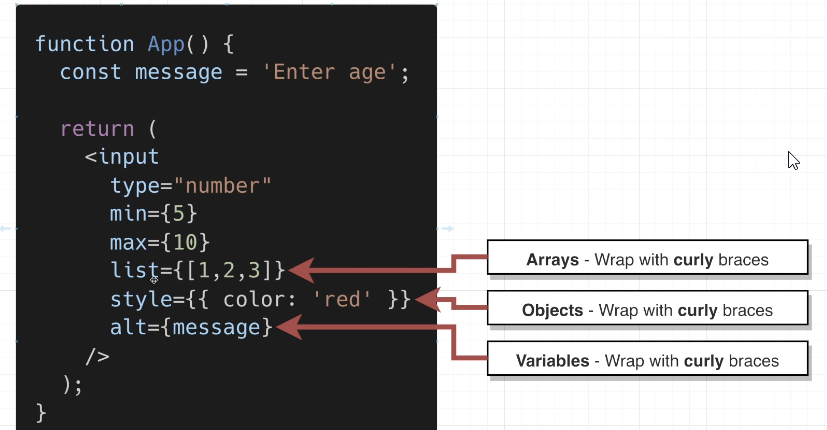
}

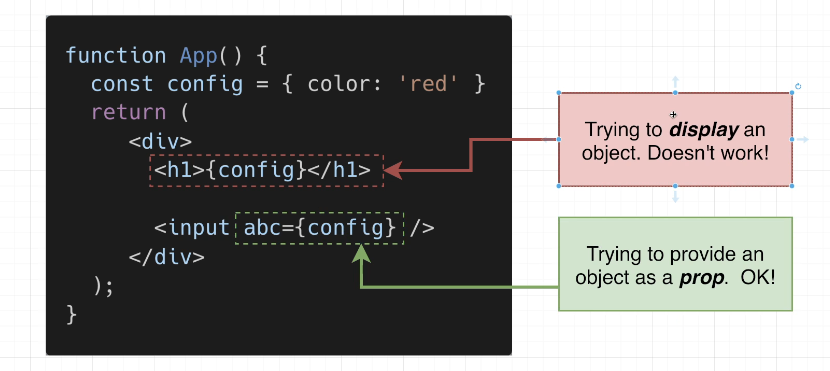


## 17. Customizing Elements with Props







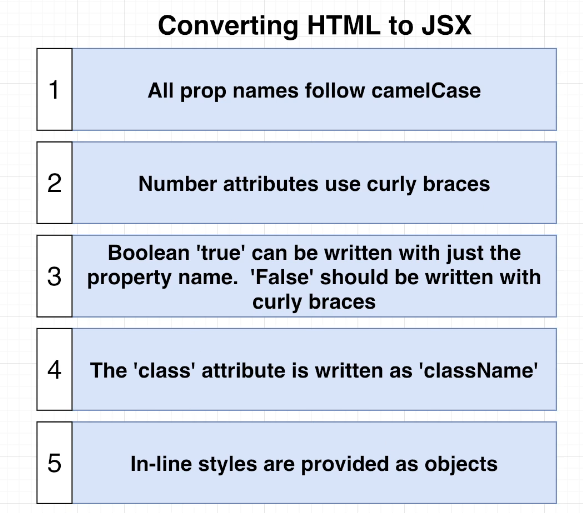
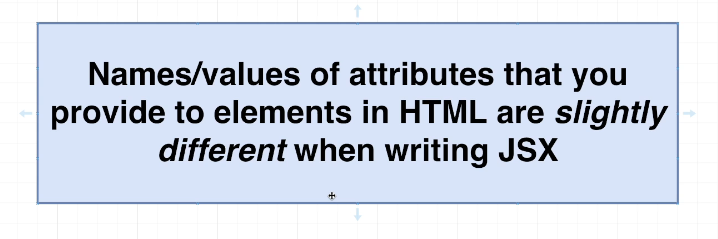


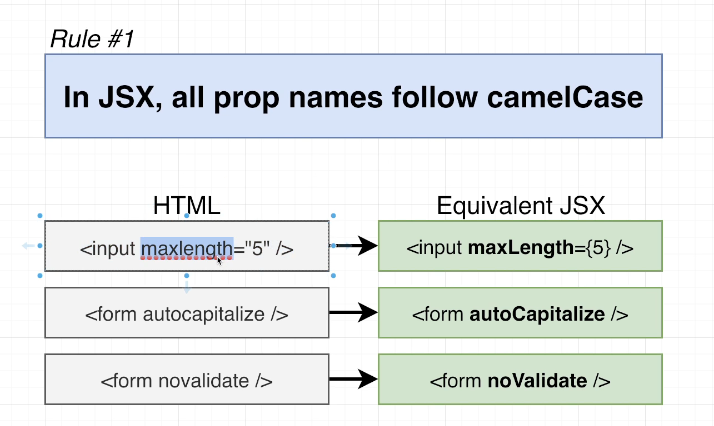
function App() {

  return <input style={{ border: '3px solid red' }} type="number" min={5} />;

}

## 18. Converting HTML to JSX





function App() {

  return <textarea autoFocus={true} />;

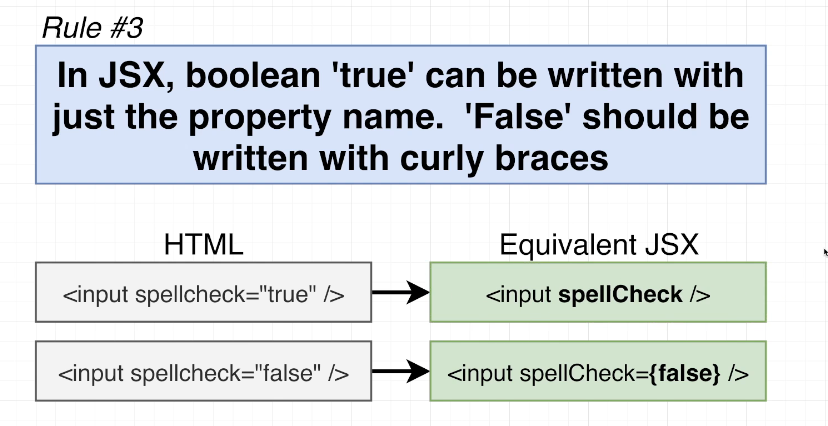
}

## 19. Applying Styling in JSX

function App() {

  return <input maxLength={5}  />;

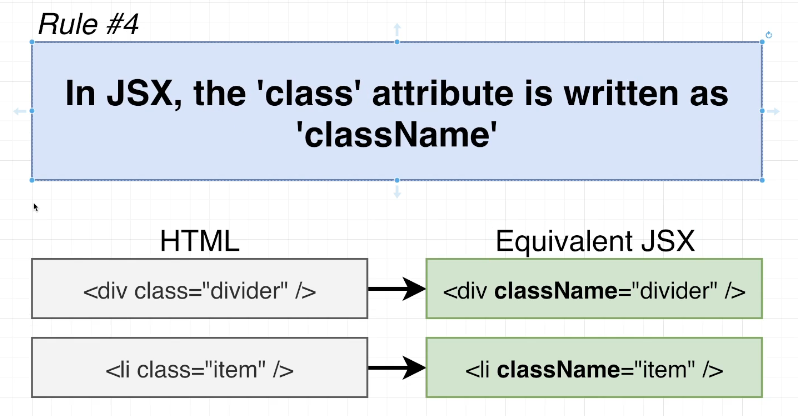
}

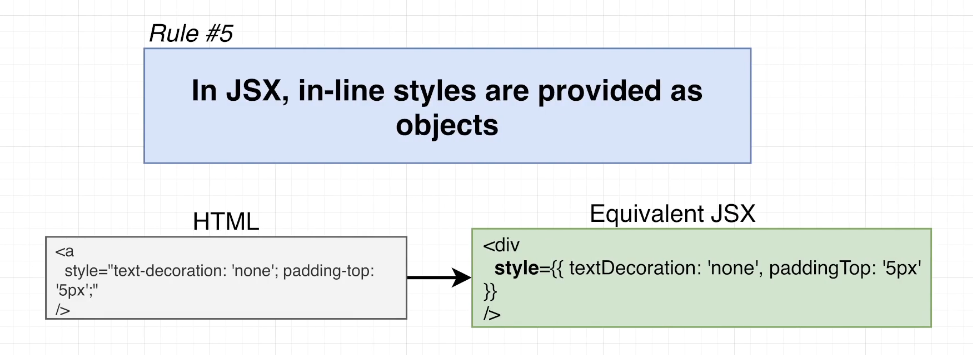


function App() {

  return <input spellCheck={false} />;

}





function App() {

  return (

    <div

      style={{

        textDecoration: "none",

        paddingTop: "10px",

        border: "1px solid black",

      }}

    >

      <p>

        test 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

        26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 4

      </p>

    </div>

  );

}

## Quiz 3: More on JSX

## Coding Exercise 2: Practice JSX Conversion

## 20. Exercise Solution

function App() {

  return (

    <div className="wrapper">

      <textarea

        readOnly={true}

        maxLength ={3}

        spellCheck ={true}

        style={{ backgroundColor: 'gray'}}

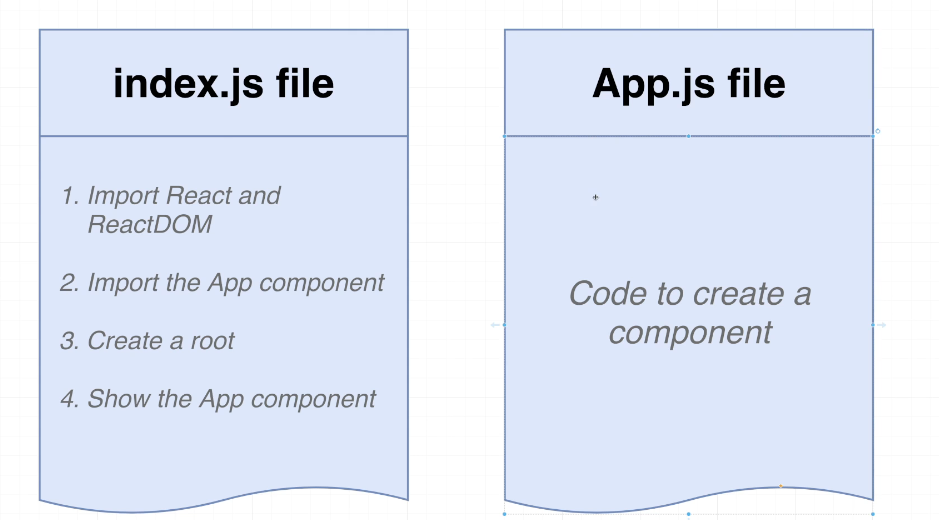
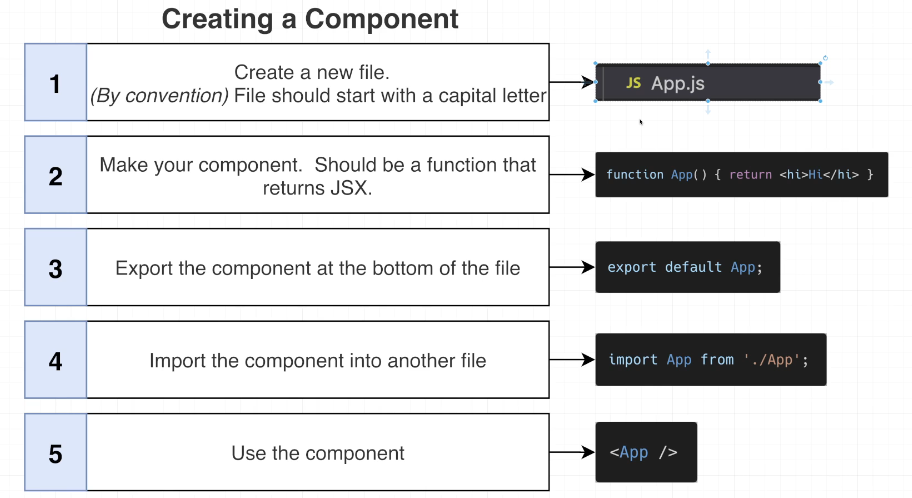
      />

    </div>

  );

}

## 21. Extracting Components

function App() {

  const name = 'Samantha';

  const age = 23;

  return (

    <h1>

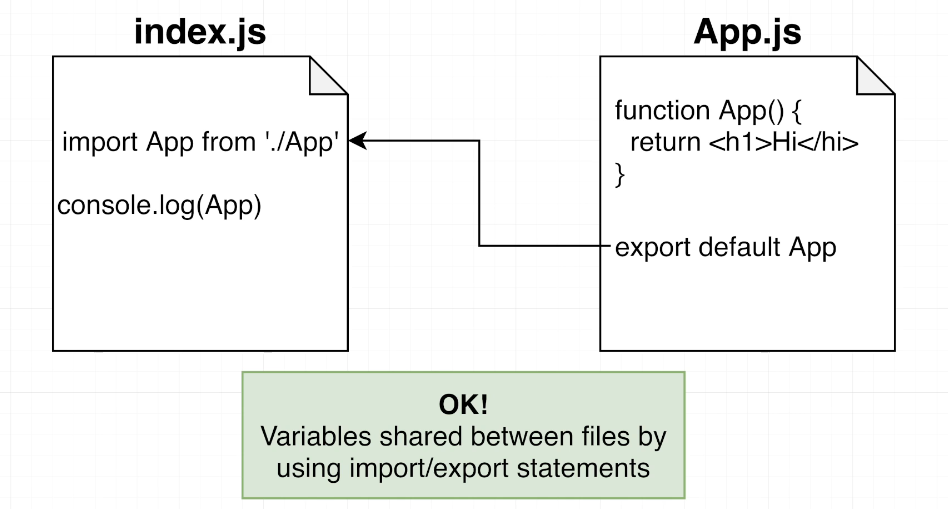
      Hi, my name is {name} and my age is {age}

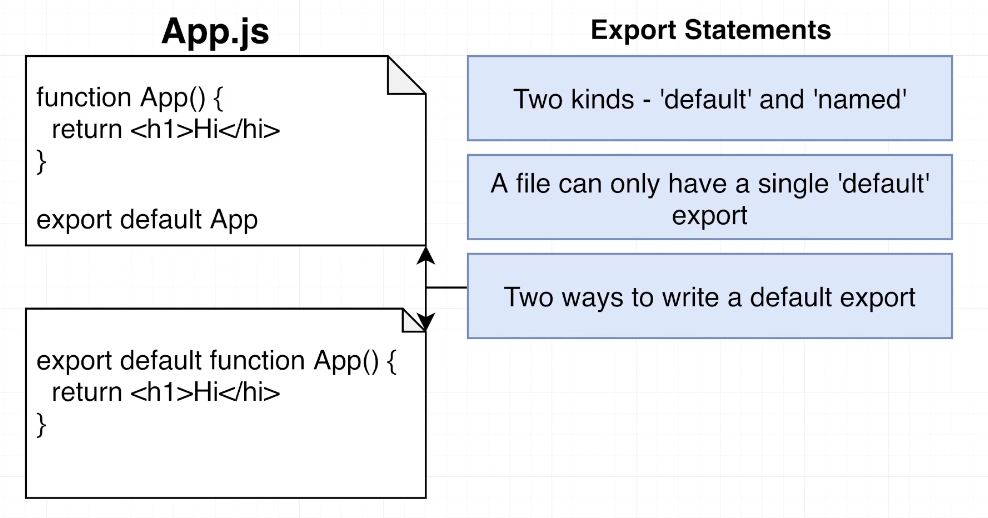
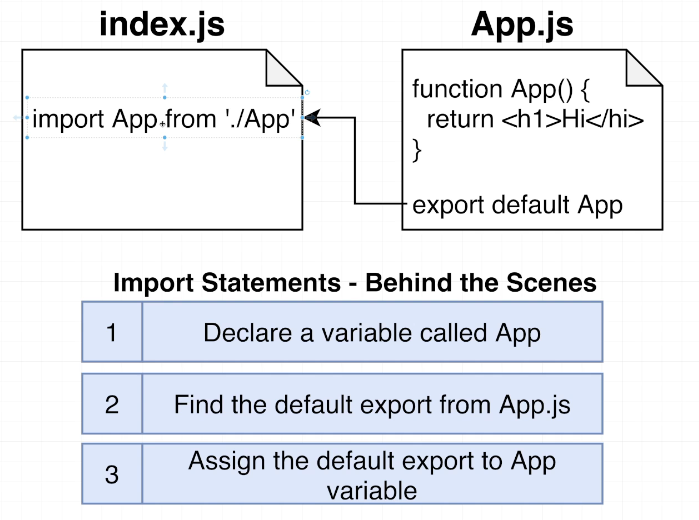
    </h1>

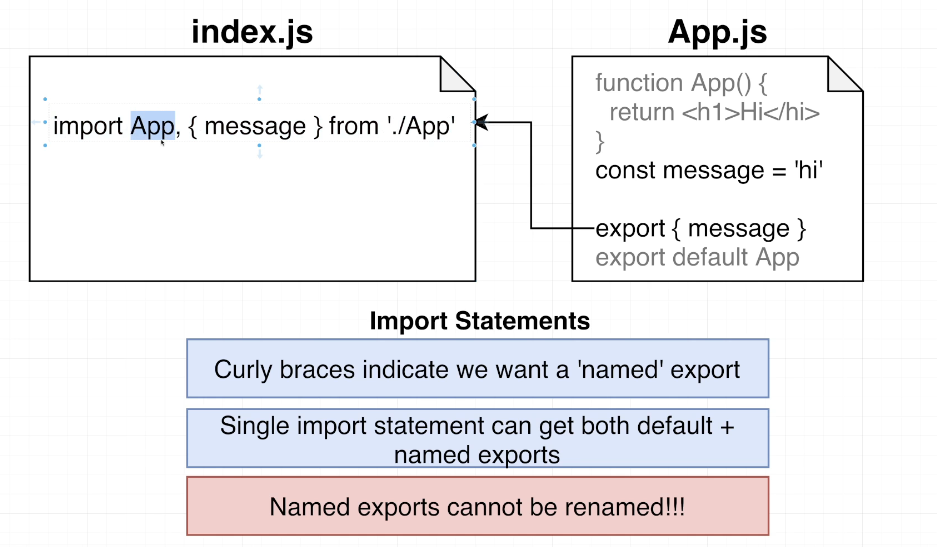
  );

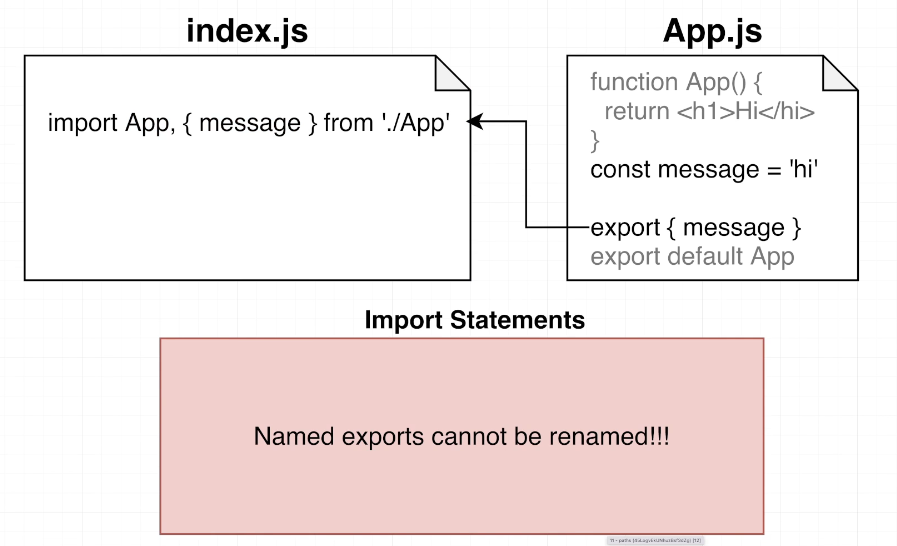
}

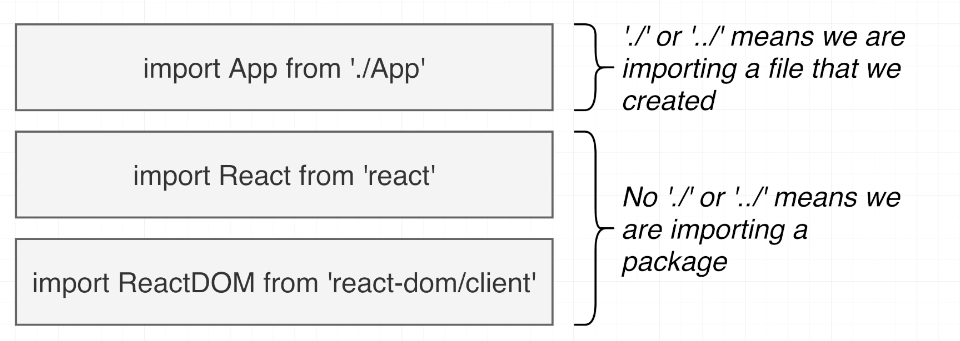
## 22. Module Systems Overview



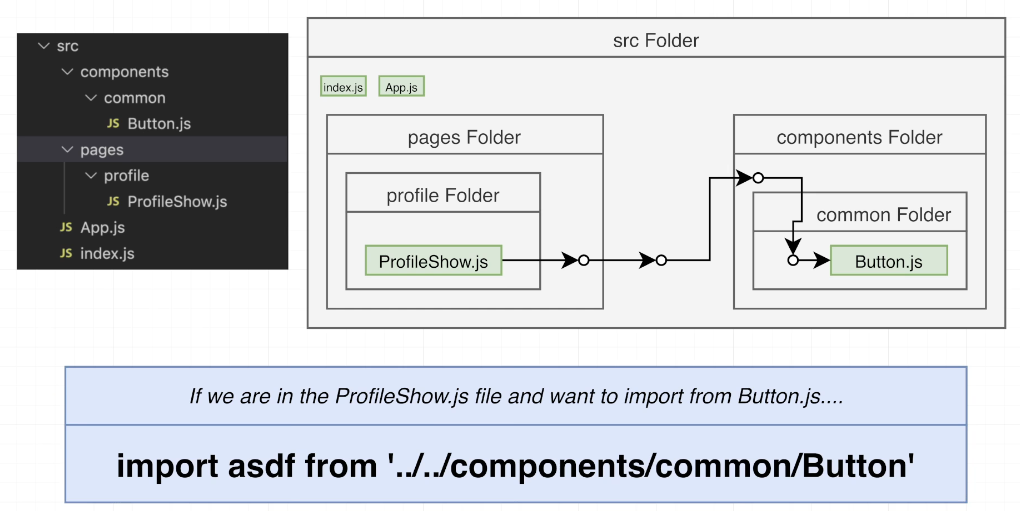
 







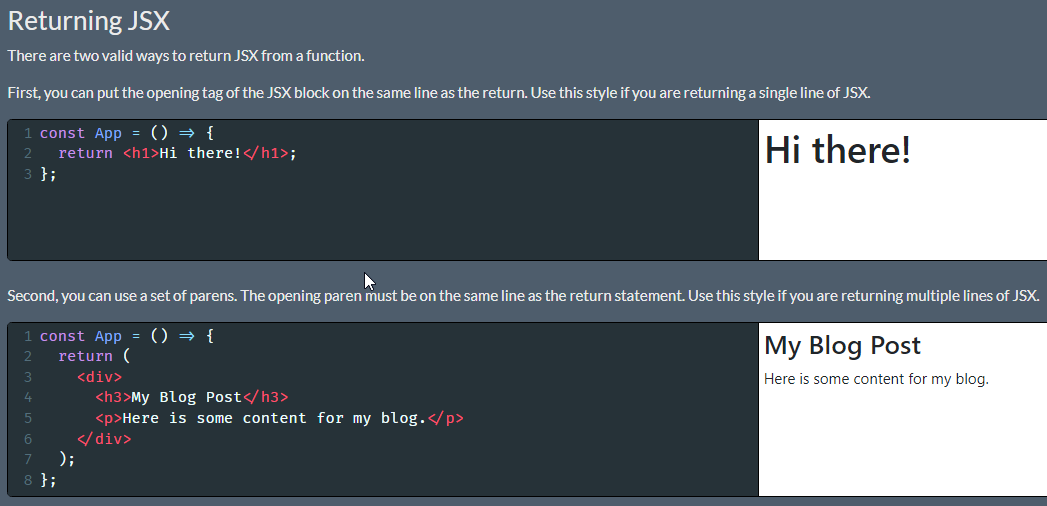




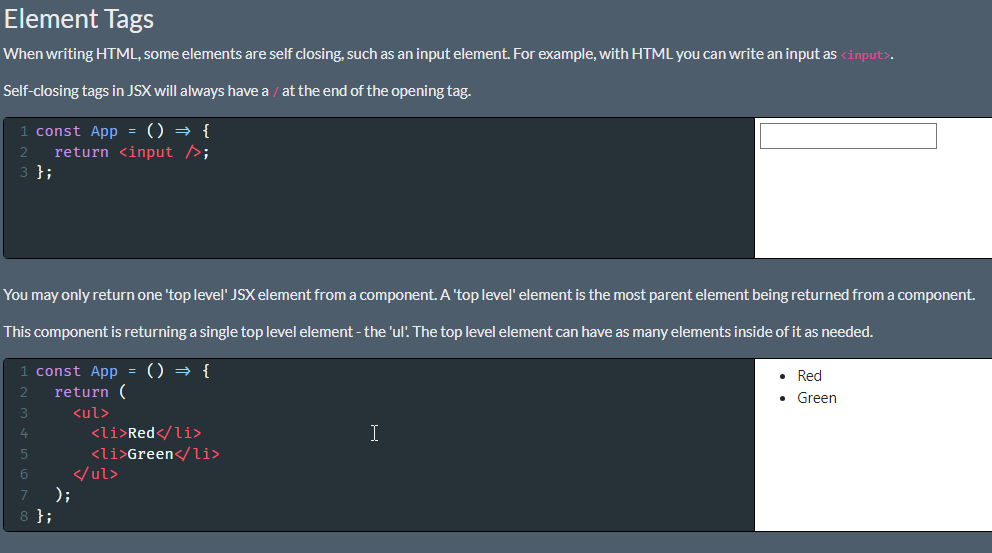
## Quiz 4: Imports and Exports

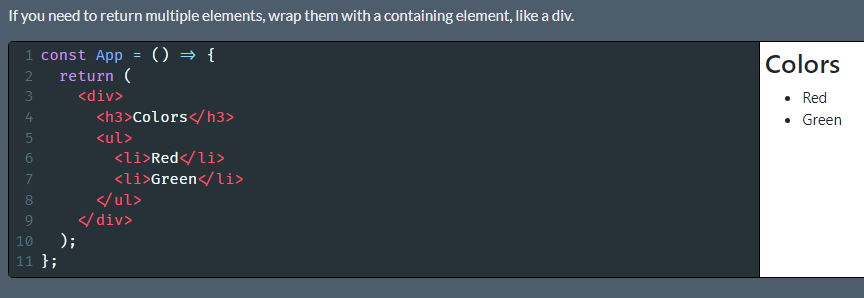
## 23. Cheatsheet for JSX

<https://jsx-notes.vercel.app/>

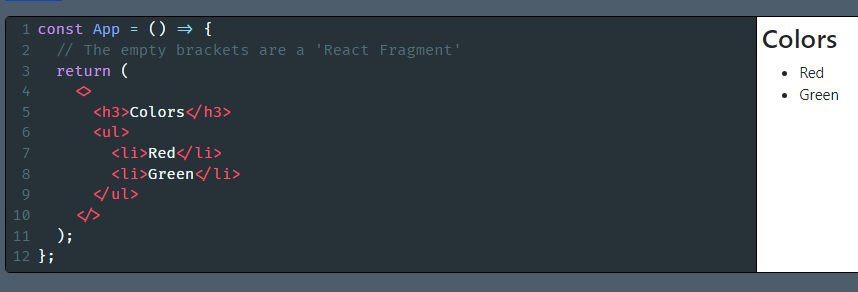


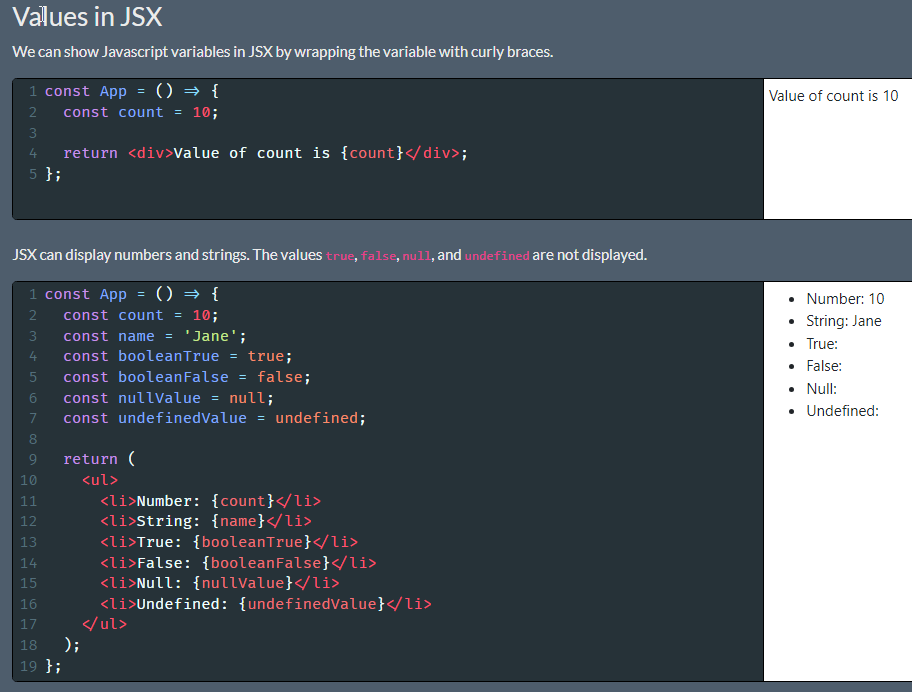
Make sure you always put the opening JSX element or paren on the same line as the return statement. If you don't, Javascript will assume you want to return nothing at all!

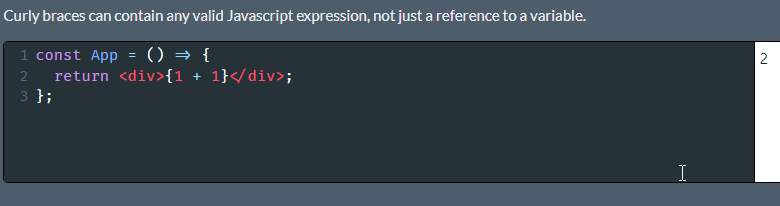


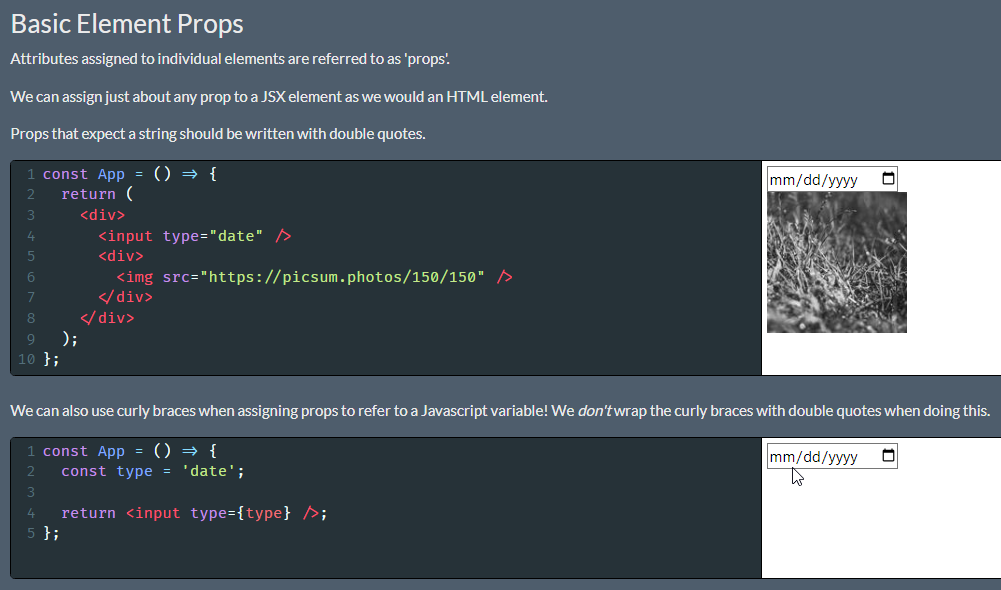


Sometimes, you don't want to insert an extra wrapping element, since it can conflict with some CSS styles. If you don't want to add an extra element, you can return wrap the content in a 'fragment', which looks like an empty JSX element.







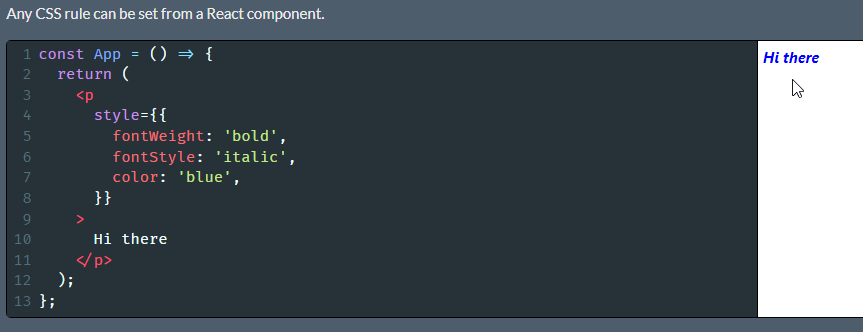






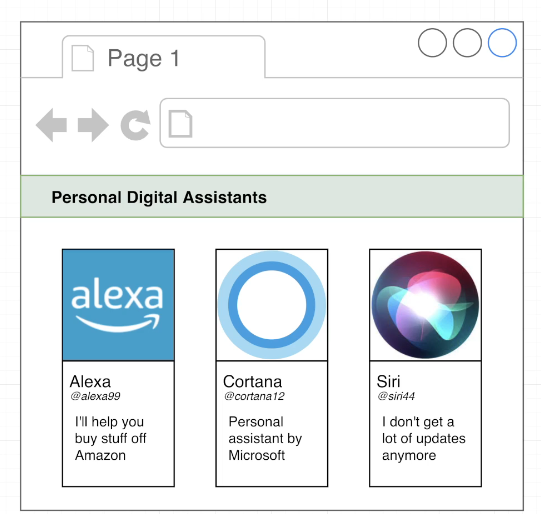
Notice the two sets of curly braces. The outer set means that we are about to write some Javascript in JSX. The inner set creates the object.

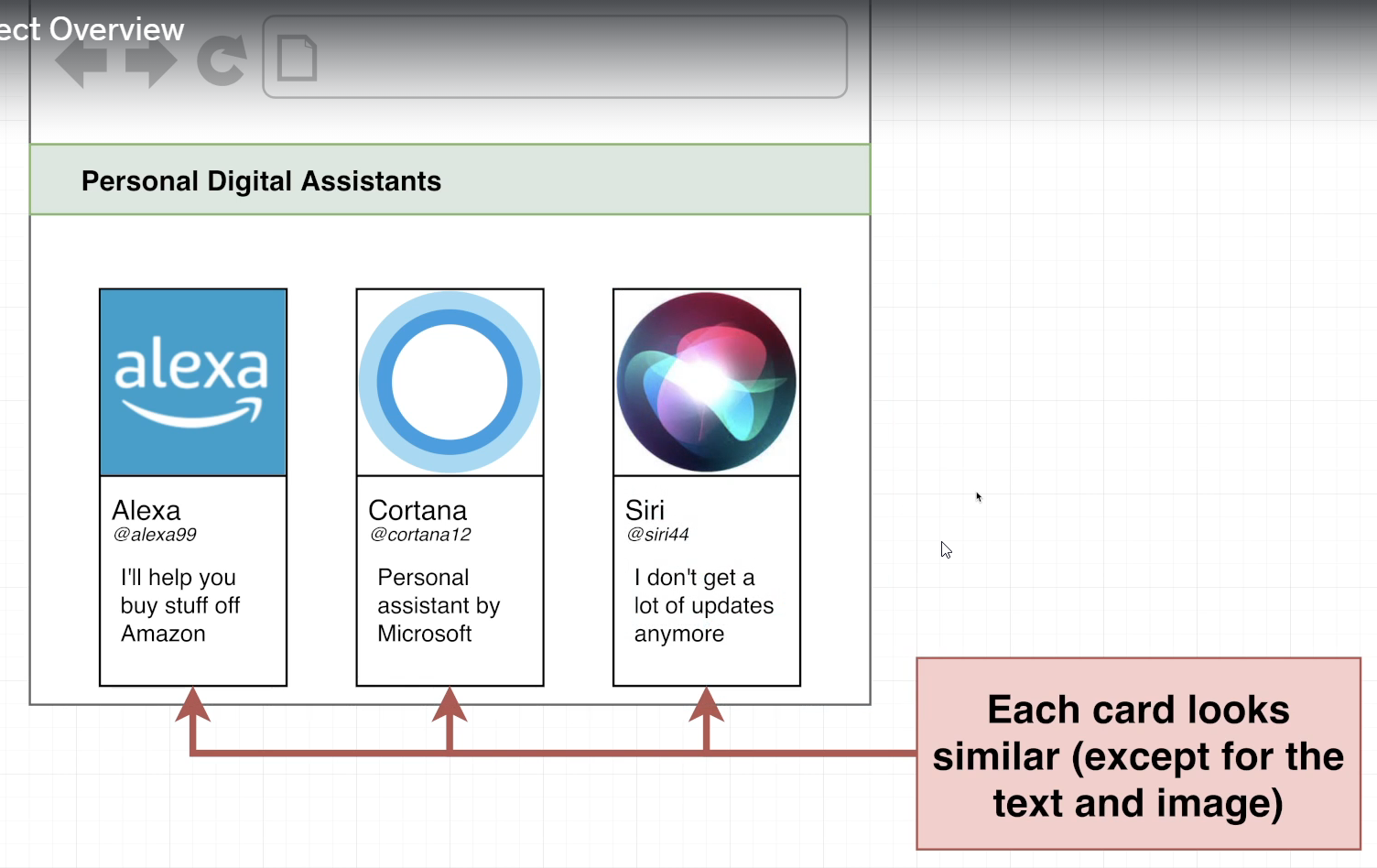
Earlier, we saw that objects can't be displayed in JSX. The key word there is 'displayed'. We can't show an object on the screen, but we can provide an object as a prop to an element.

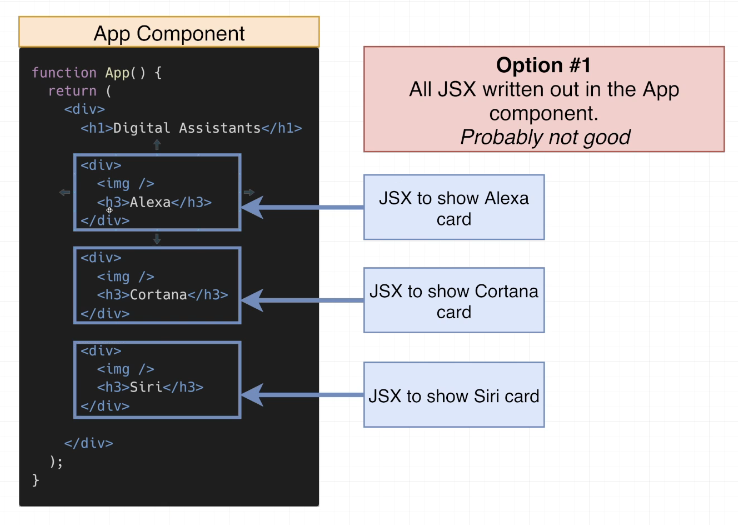


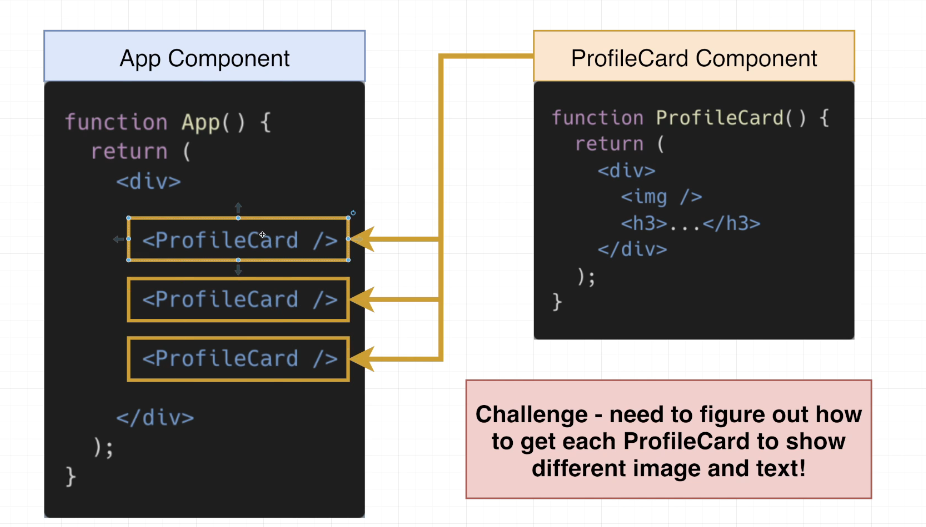
# Section 3: Building with Reusable Components

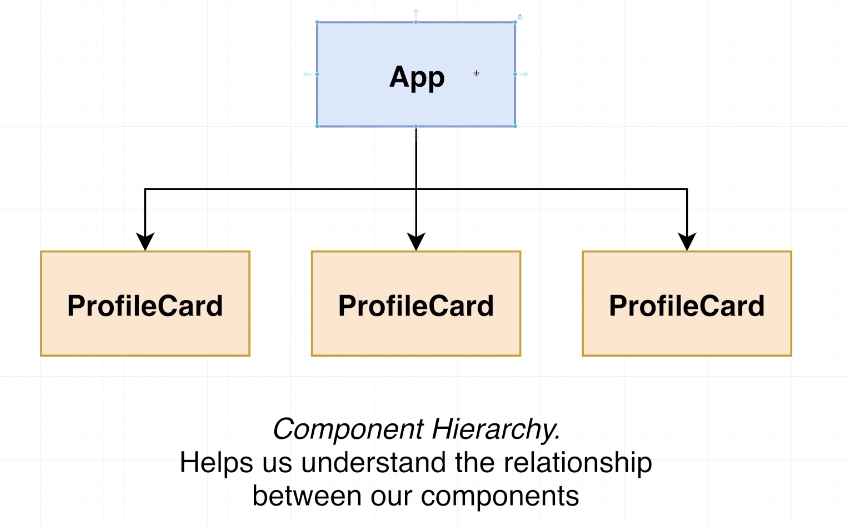
## 24. Project Overview

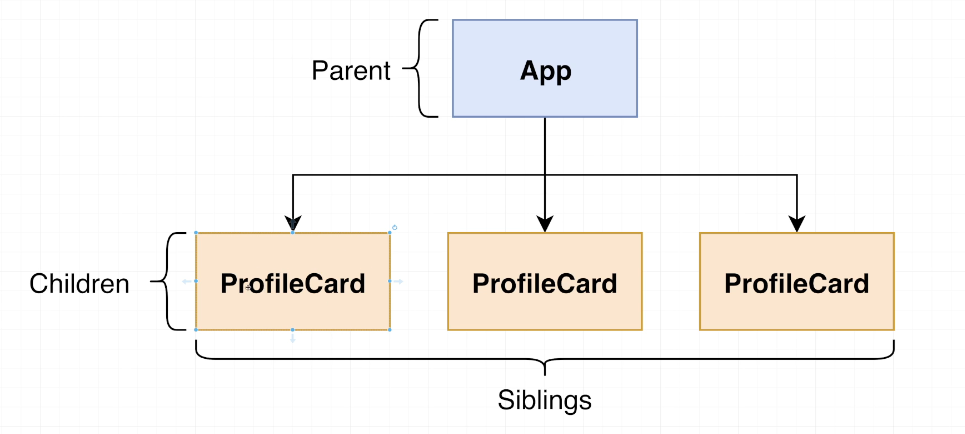


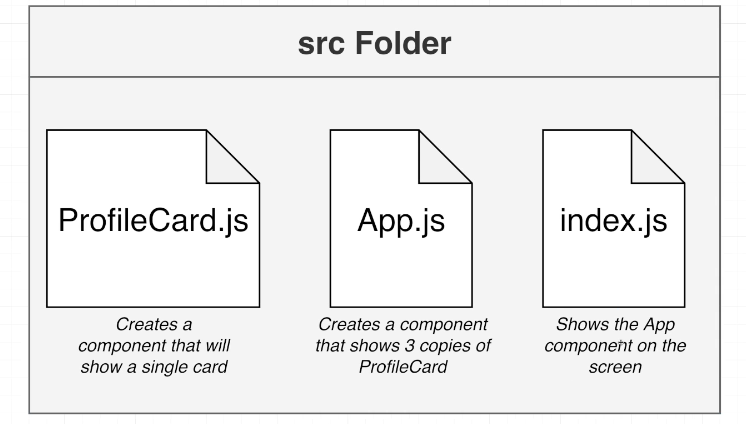












npx create-react-app pdas

cd pdas

npm start

## 25. Creating Core Components

ProfileCard.js

function ProfileCard(){

    return <div>Profile Card</div>

}

export default ProfileCard;

App.js

import ProfileCard from "./ProfileCard";

function App() {

  return (

    <div>

      <div>Personal Digital Assistans</div>

      <ProfileCard />

      <ProfileCard />

      <ProfileCard />

    </div>

  )

}

export default App;

index.js

import React from 'react';

import ReactDOM from 'react-dom/client';

import App from './App';

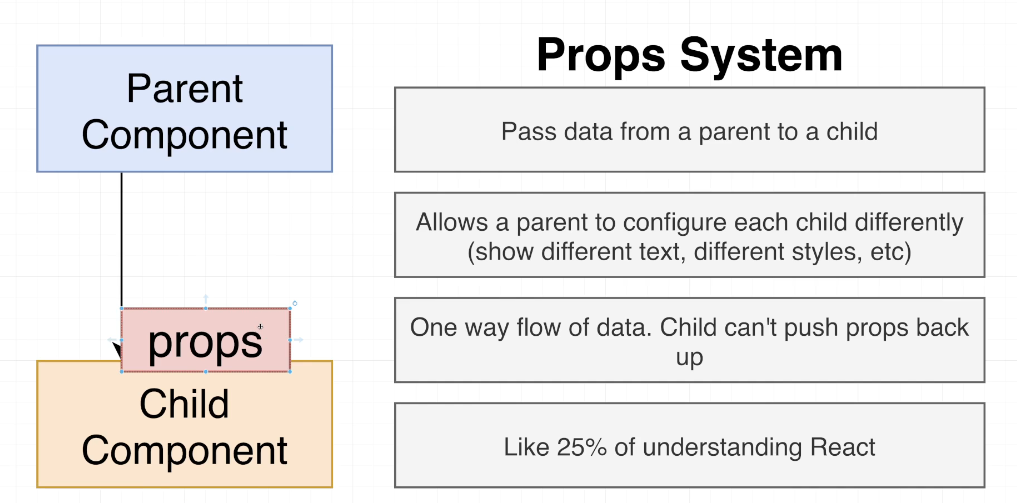
const el = document.getElementById('root');

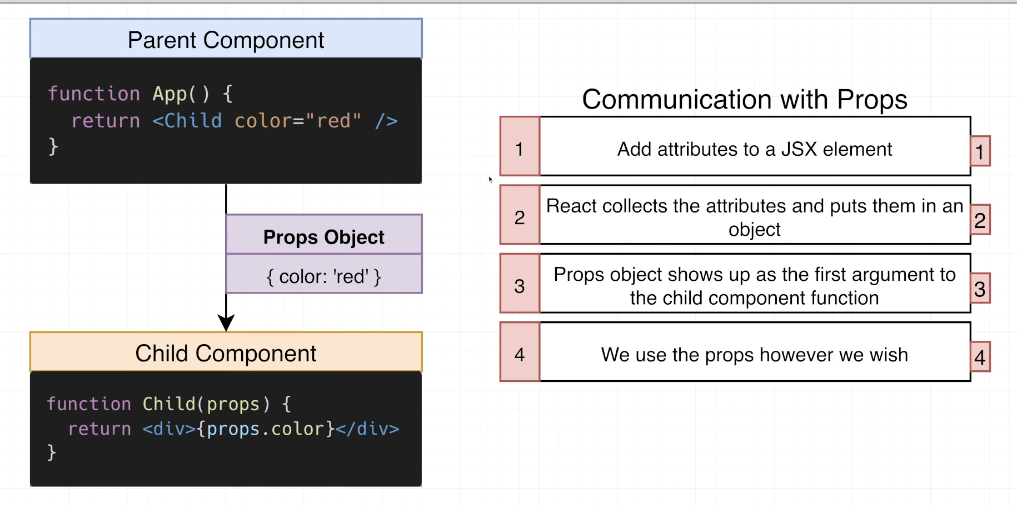
const root = ReactDOM.createRoot(el);

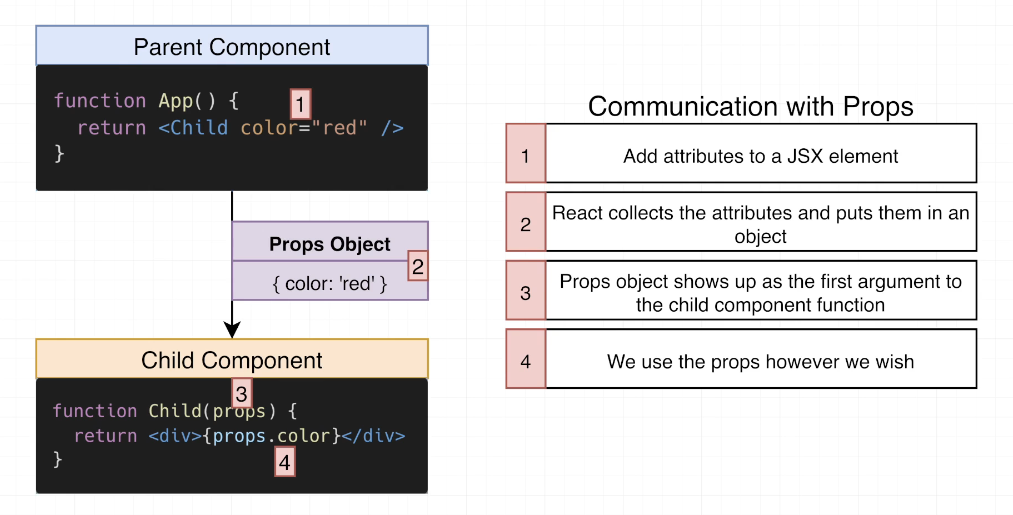
root.render(<App />);

## 26. Introducing the Props System

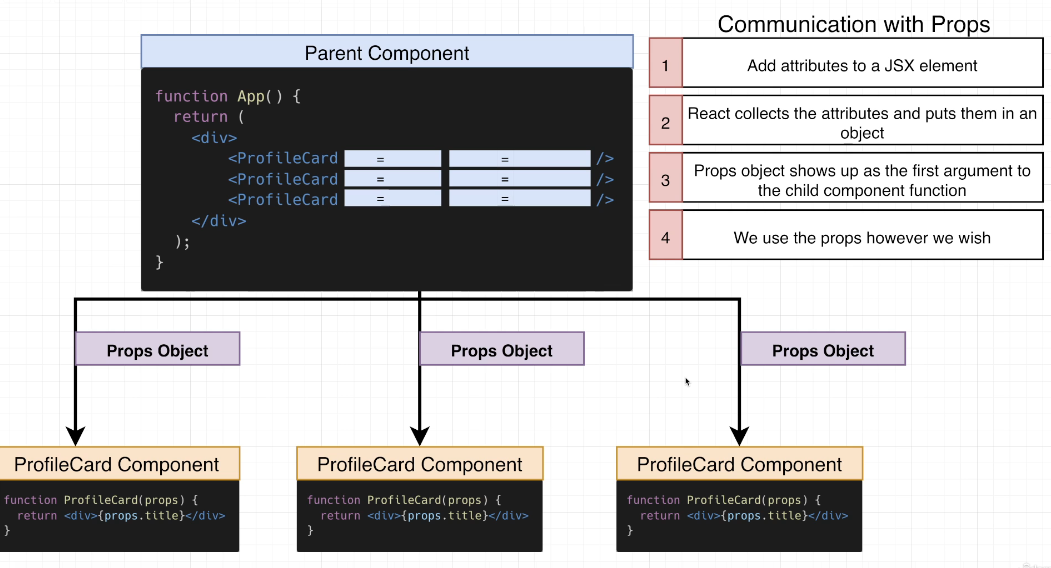


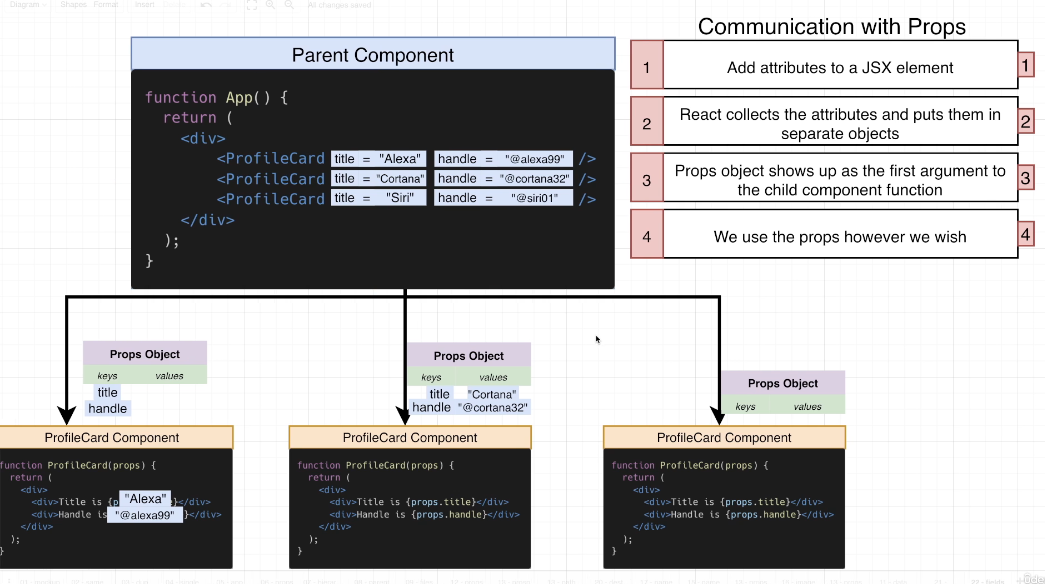






## 27. Picturing the Movement of Data





## 28. Adding Props

Index.js

import React from 'react';

import ReactDOM from 'react-dom/client';

import App from './App';

const el = document.getElementById('root');

const root = ReactDOM.createRoot(el);

root.render(<App />);

App.js

import ProfileCard from "./ProfileCard";

function App() {

  return (

    <div>

      <div>Personal Digital Assistans</div>

      <ProfileCard title="Alexa" handle="@alexa99" />

      <ProfileCard title="Cortana" handle="@cortana32" />

      <ProfileCard title="Siri" handle="@siri01" />

    </div>

  )

}

export default App;

ProfileCard.js

function ProfileCard(props){

    console.log('props', props);

    return (

    <div>

        <div>Title is {props.title}</div>

        <div>Handle is {props.handle}</div>

    </div>

    )

}

export default ProfileCard;

## 29. Using Argument Destructuring

function ProfileCard({title , handle}) {

return (

    <div>

        <div>Title is {title}</div>

        <div>Handle is {handle}</div>

    </div>

    )

}

export default ProfileCard;

## Coding Exercise 3: Practice with Props

BrightText.js

import React from 'react';

// This is all done!  No changes needed!

export default function BrightText({ color }) {

  const style = { color: color };

  // You might see an error on this line from the code editor

  // There is no error, no changes are needed

  return <h1 style={style}>Hi!</h1>

}

App.js

import React from 'react';

import BrightText from './BrightText'

function App() {

    return (

        <div>

          <BrightText color = "red" />

          <BrightText color = "green" />

          <BrightText color = "blue" />

        </div>

    );

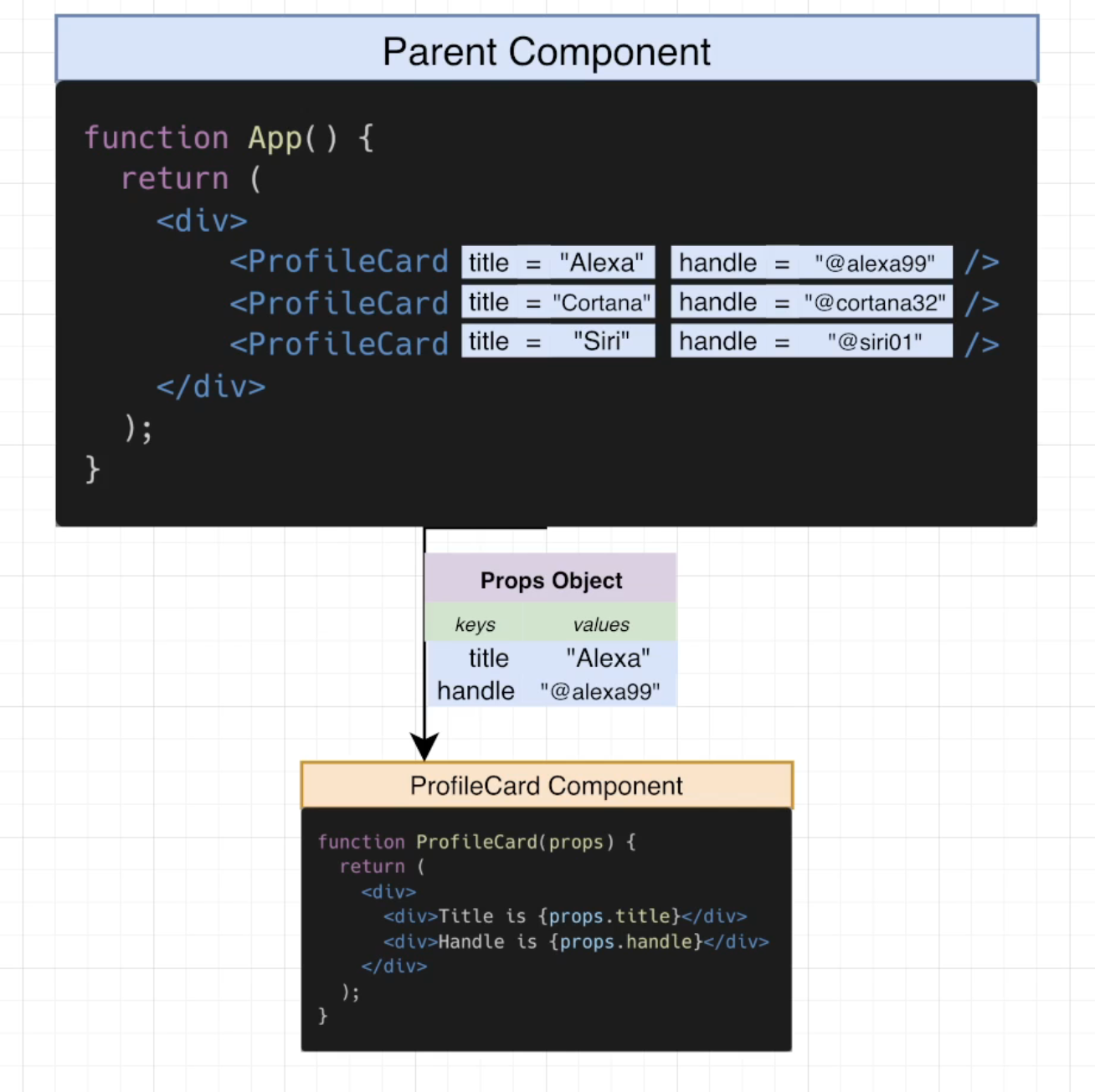
}

export default App;

## 31. The React Developer Tools

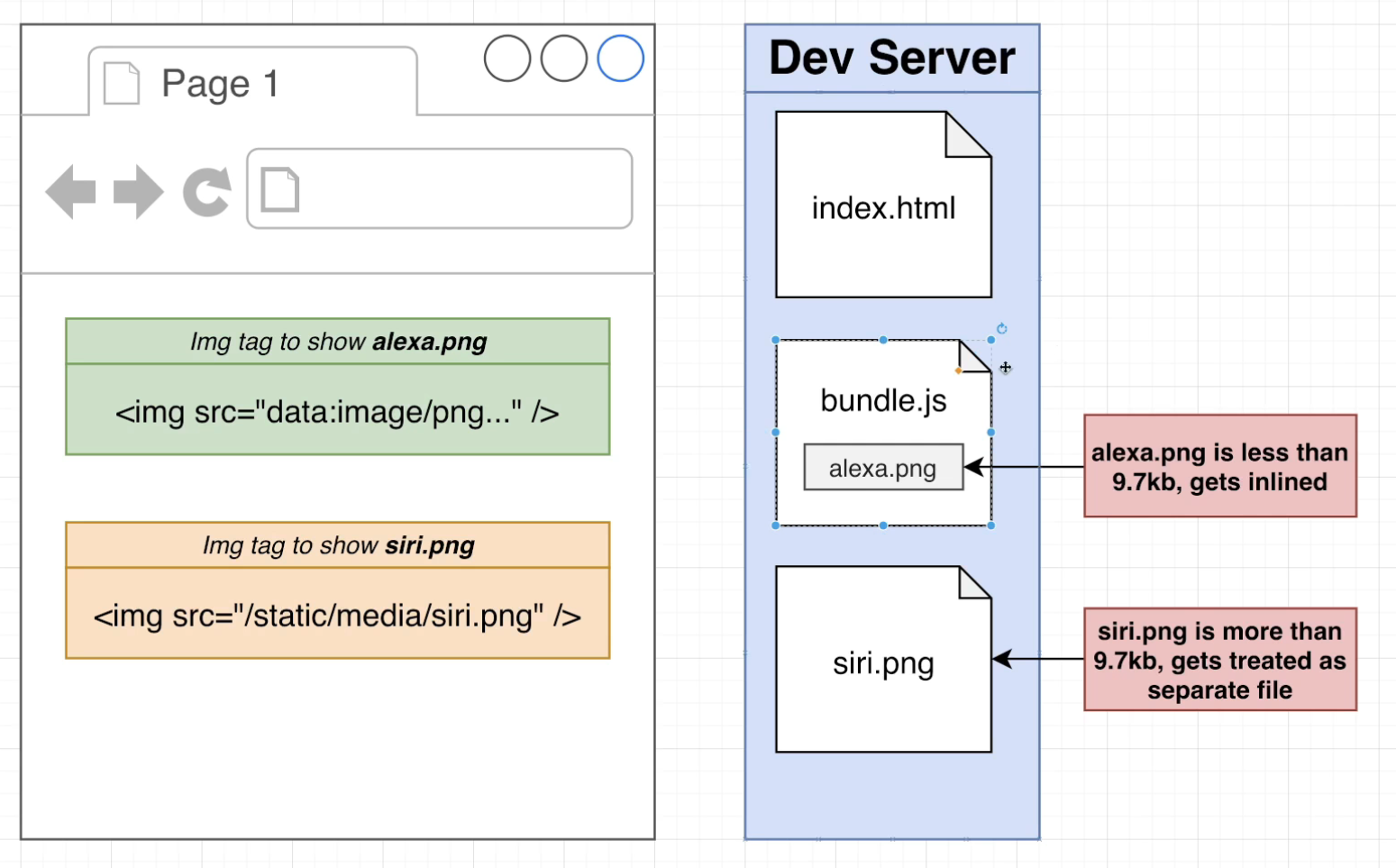
Install react-developer-tools from the <https://react.dev/learn/react-developer-tools> to your default internet browser for debugging react app.

## 32. The Most Common Props Mistake



## Quiz 5: Communicating With Props

## 33. Images for the App



## 34. Including Images

## 35. Handling Image Accessibility

## 36. Review on how CSS Works

## 37. Adding CSS Libraries with NPM

## 38. A Big Pile of HTML!

## 39. Last Bit of Styling