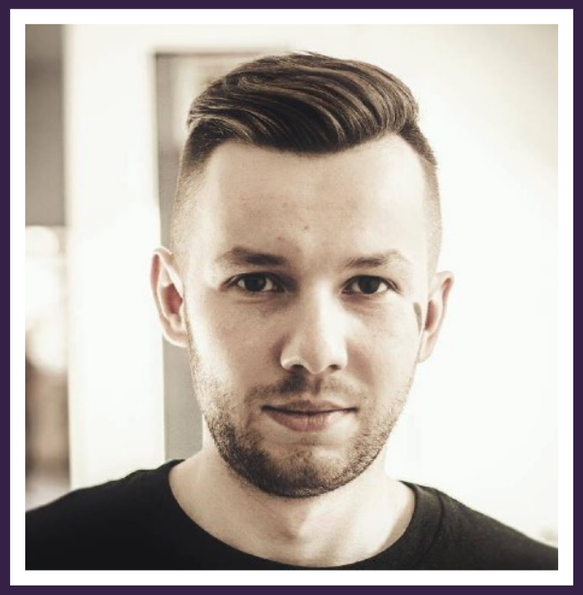
REACT JS



PRZEMYSŁAW WISZOWATY

HELLO!

SOFTware HUT

TENDERHUT GROUP

meet.js
białystok

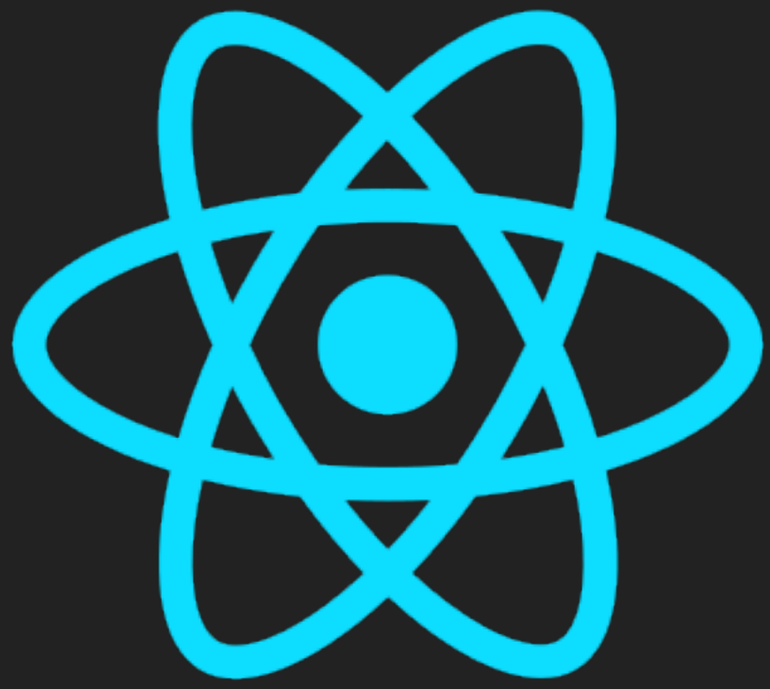
REACT IS EVERYWHERE

„Learn Once, Write Anywhere“

FRONTEND

BACKEND

MOBILE




A back to the 2000s
N.R. (06)
YES ☐ NO ☐
PHILIPS
UC*II
60
TYPE II · HIGH POSITION – 70µs EQ

The Geocities-izer

ge Look Like It Was Made By A 13 Year-Old In 1996

Type any URL in the box below and click Submit to see how it would look as a Geocities page.

Or Try one of these: The New York Times YouTube BoingBoing

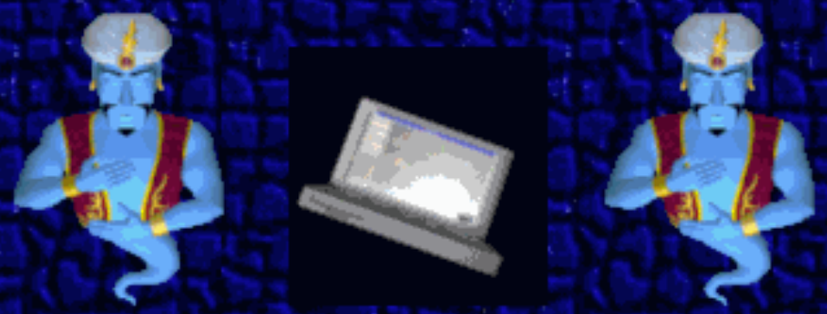
http://

Submit

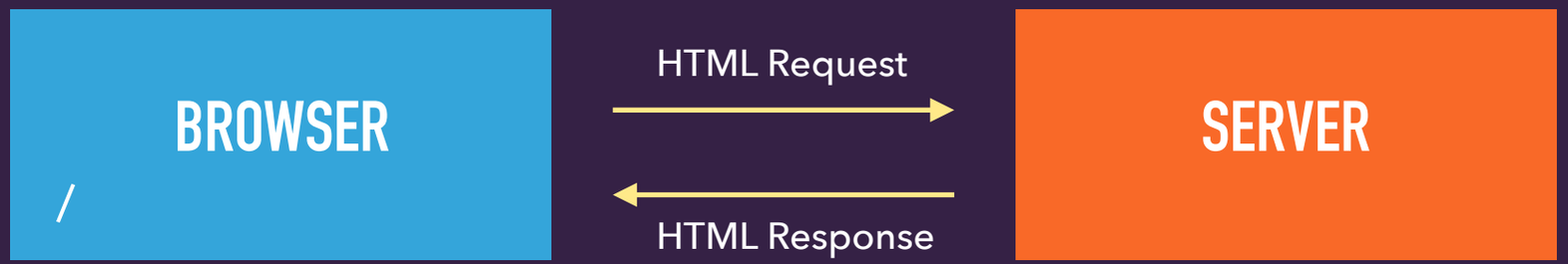
Some pages may work very slowly or not all. Many webapps are just too advanced for Geocities.

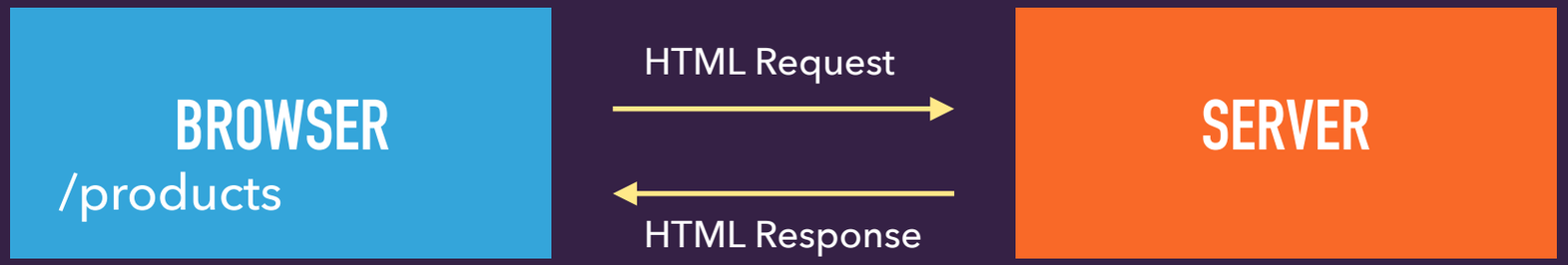
Turn your sound up for the full effect.

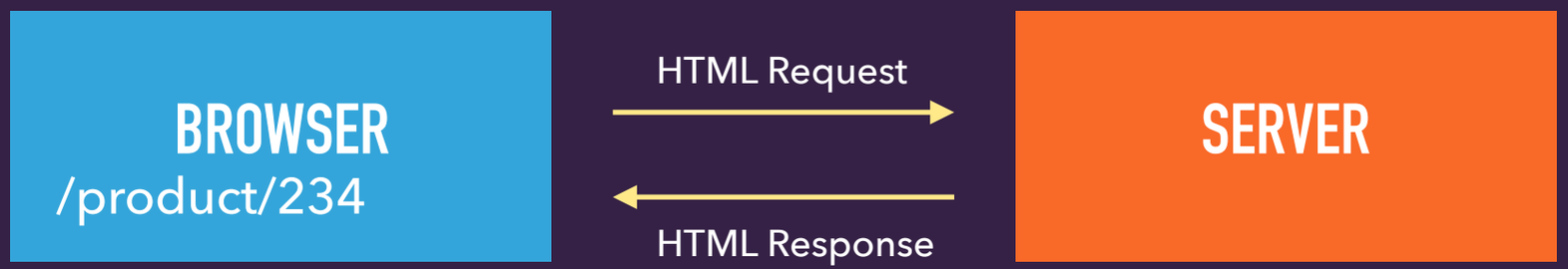
Created by Mike Lacher



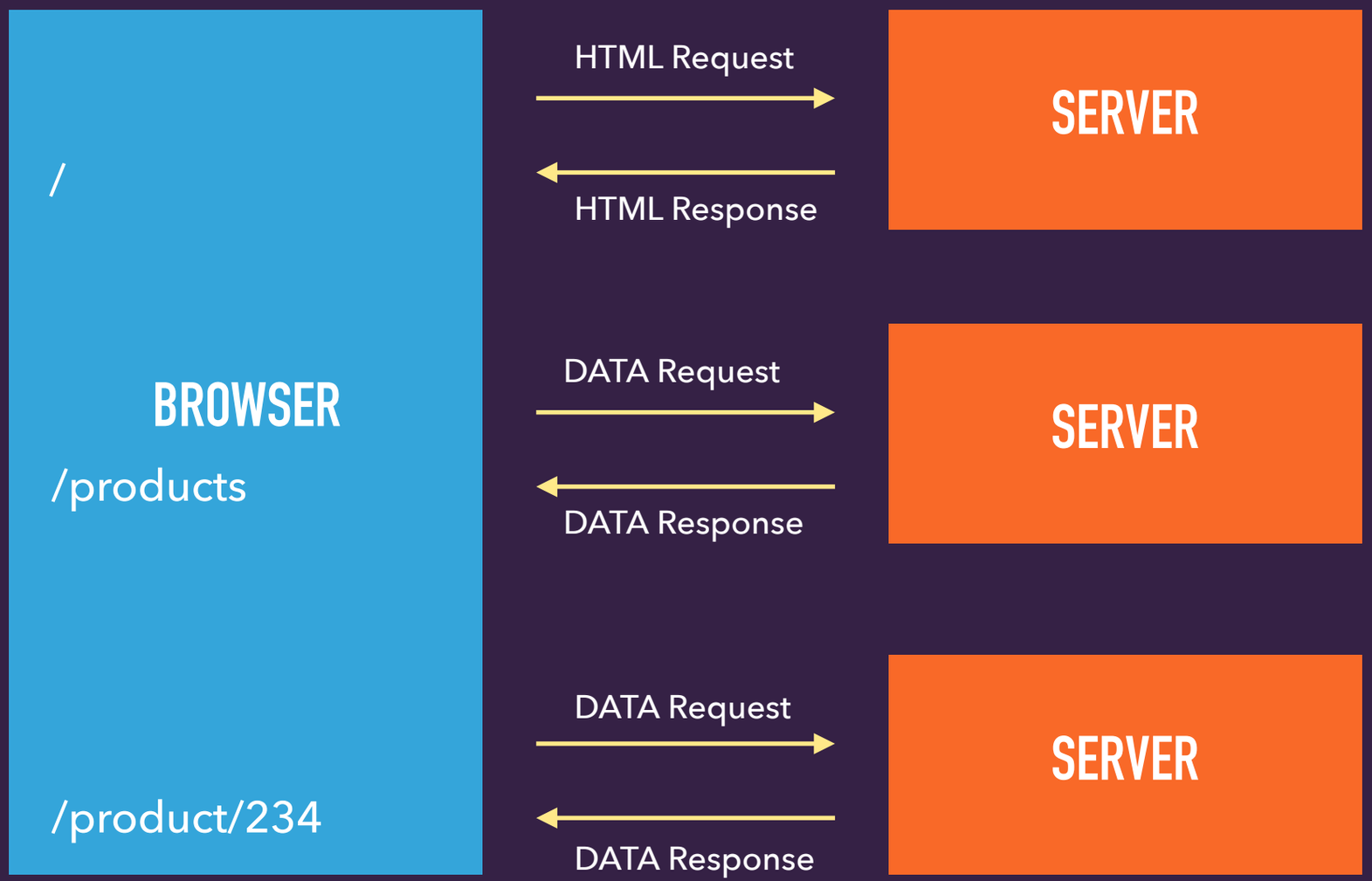
TRADITIONAL WEBSITE







SINGLE PAGE APPLICATION



Chrome Plik Edytuj Widok Historia Zakładki Osoby Okno Pomoc

TenderHut S.A. - Technologicz

https://tenderhut.com

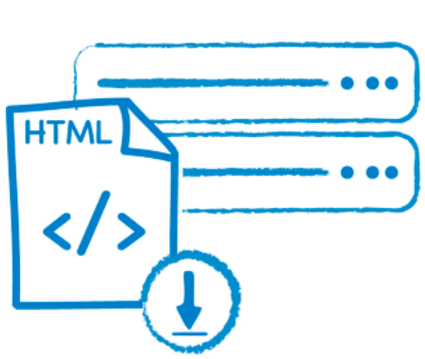
414 × 736

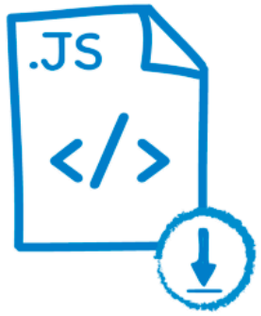
Elements Console Sources Network Performance Memory Application Security Audits

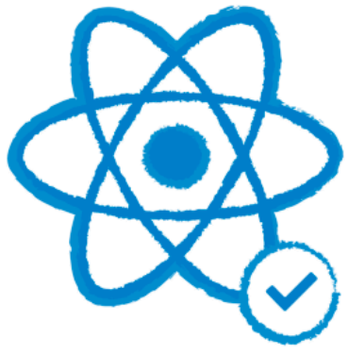
(index) :formatted

```
<!DOCTYPE html>
<html lang="en" prefix="og: http://ogp.me/ns#">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width,initial-scale=1,shrink-to-fit=no">
        <link rel="apple-touch-icon" sizes="180x180" href="/icons/apple-touch-icon.png?v=694AzX47gQ">
        <link rel="icon" type="image/png" sizes="32x32" href="/icons/favicon-32x32.png?v=694AzX47gQ">
        <link rel="icon" type="image/png" sizes="16x16" href="/icons/favicon-16x16.png?v=694AzX47gQ">
        <link rel="mask-icon" href="/icons/safari-pinned-tab.svg?v=694AzX47gQ" color="#5bbad5">
        <link rel="shortcut icon" href="/icons/favicon.ico?v=694AzX47gQ">
        <meta name="msapplication-TileColor" content="#2d89ef">
        <meta name="msapplication-config" content="/icons/browserconfig.xml?v=694AzX47gQ">
        <meta name="theme-color" content="#ffffff">
        <meta name="google-site-verification" content="NCJvpBpTbp9pUJdDk7t6dhIisOj4kZJEphgntRU-TbM"/>
        <link rel="manifest" href="/manifest.json">
        <title>TenderHut</title>
        <link href="/static/css/main.5ca042e5.css" rel="stylesheet">
    </head>
    <body>
        <noscript>You need to enable JavaScript to run this app.</noscript>
        <div id="app-root"></div>
        <div id="modal-root"></div>
        <script type="text/javascript" src="/static/js/main.dc65024f.js"></script>
    </body>
</html>
```

CSR





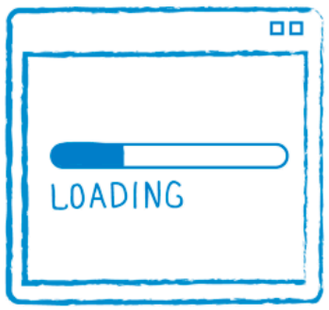


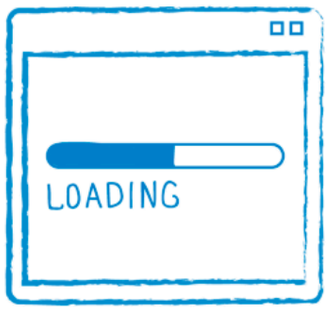
Server Sending Response to Browser

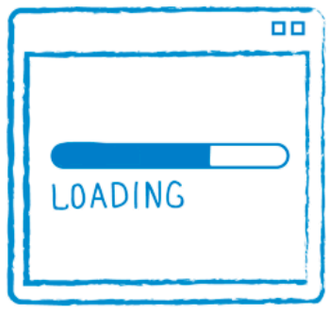
Browser Downloads JS

Browser executes React

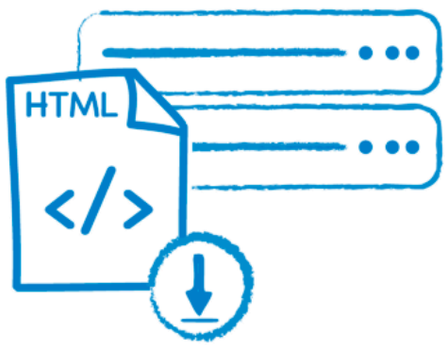
Page Now **Viewable** and **Interactable**



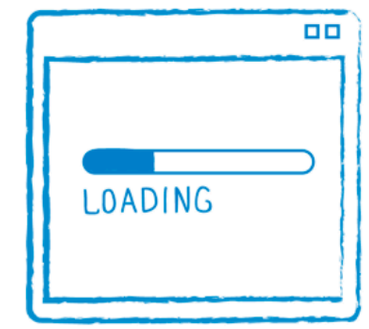


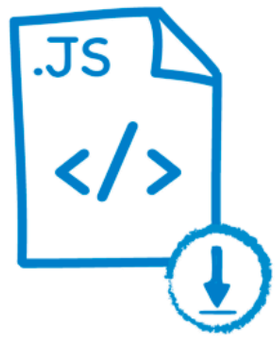


SSR

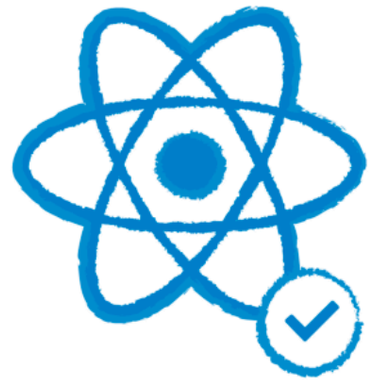


Server Sending Ready to be rendered HTML Response to Browser





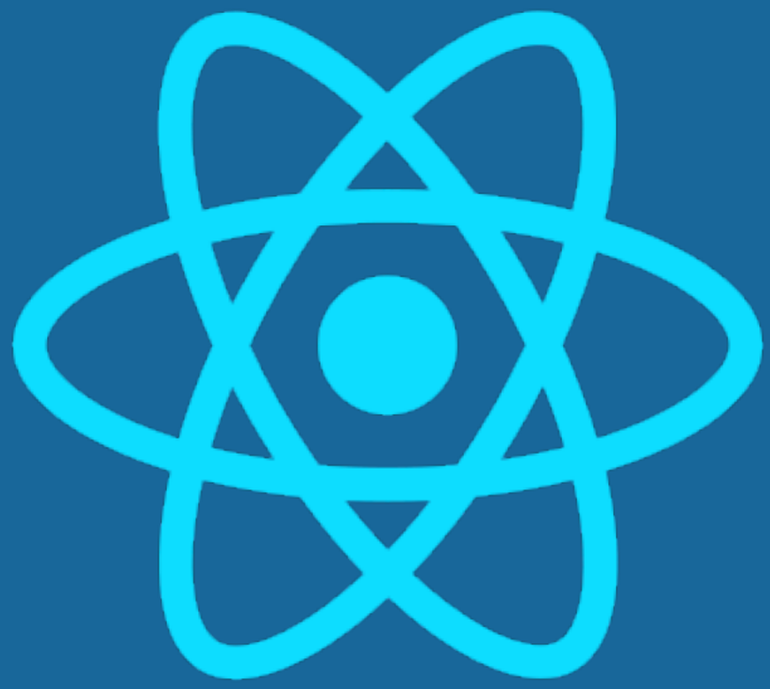
Browser Renders the page, **Now Viewable**, and Browser Downloads JS



Browser executes React

Page Now **Interactable**

WHAT IS REACT?

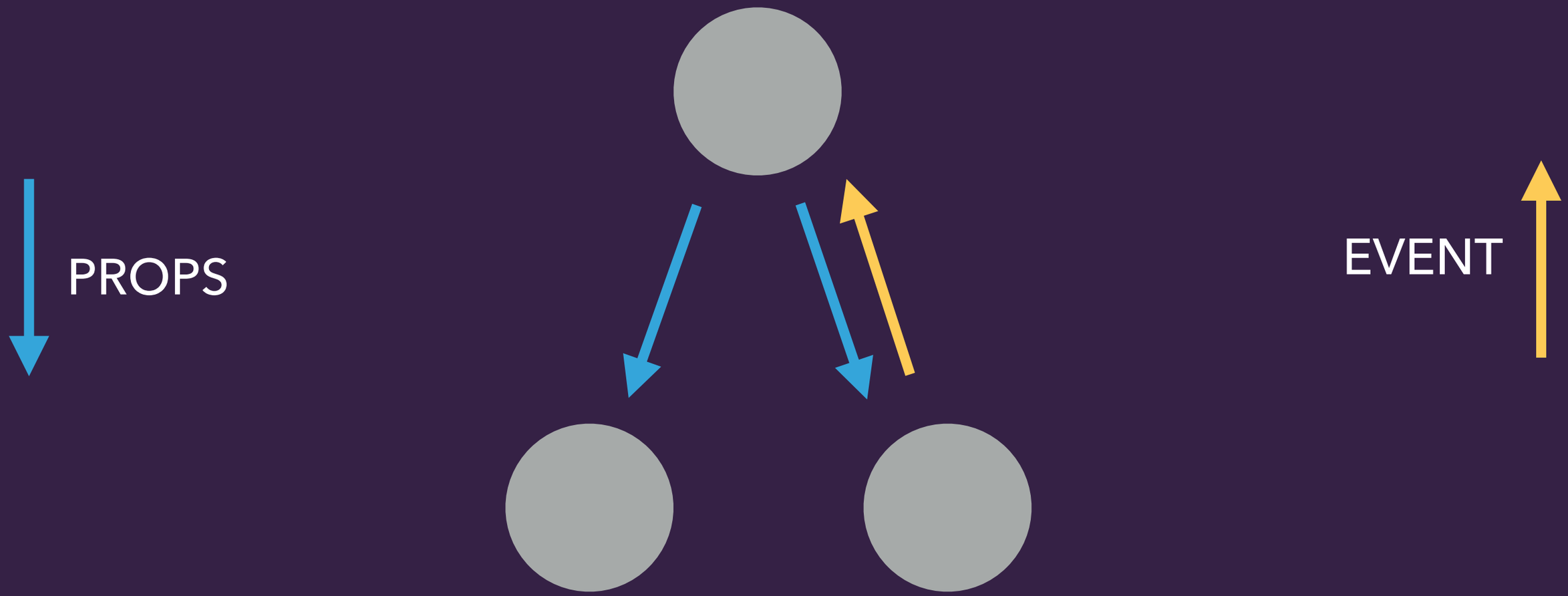


LIBRARY NOT ~~FRAMEWORK~~

LOW LEARNING CURVE

ONE WAY DATA FLOW

ONE WAY DATA FLOW



NO CONTROLLERS

NO MODELS

NO DIRECTIVES

NO GLOBAL EVENT LISTENER

JUST COMPONENT

COMPONENT

ISOLATED

REUSABLE

TESTABLE

COMPONENT TYPES

CLASS

```
class Welcome extends React.Component {
  render() {
    return <h1>Hello {this.props.name}</h1>
  }
}
```

FUNCTIONAL

```
function Welcome(props) {
    return <h1>Hello, {props.name}</h1>;
}
```

FUNCTIONAL

```
const Welcome = (props) => {
    return <h1>Hello, {props.name}</h1>;
};
```

FUNCTIONAL

```
const Welcome = (props) => {
    const {name} = props;
    return <h1>Hello, {name}</h1>;
}
```

FUNCTIONAL

```
const Welcome = ({name}) => {
    return <h1>Hello, {name}</h1>;
};
```

<Welcome />

EVERYTHING IS A COMPONENT

CardComponent

CardHeaderComponent

CardBodyComponent

UserPhotoComponent

```jsx
import React from 'react';

const FacebookComponent = () => (
    <CardComponent>
        <CardHeaderComponent>
            <SelectorComponent />
            <SelectorComponent />
            <SelectorComponent />
        </CardHeaderComponent>

        <CardBodyComponent>
            <UserPhotoComponent />
        </CardBodyComponent>

        <CardFooterComponent />
    </CardComponent>
);

export default FacebookComponent;

```

JSX

```jsx
const Welcome = () => {
  return <h1>Hello, World</h1>;
};
```

JSX

```js
const Welcome = function Welcome() {
    return React.createElement(
        type: "h1",
        props: null,
        children: "Hello, World"
    );
};
```

JS

REACT DOESN'T REQUIRE USING JSX

VIRTUAL DOM

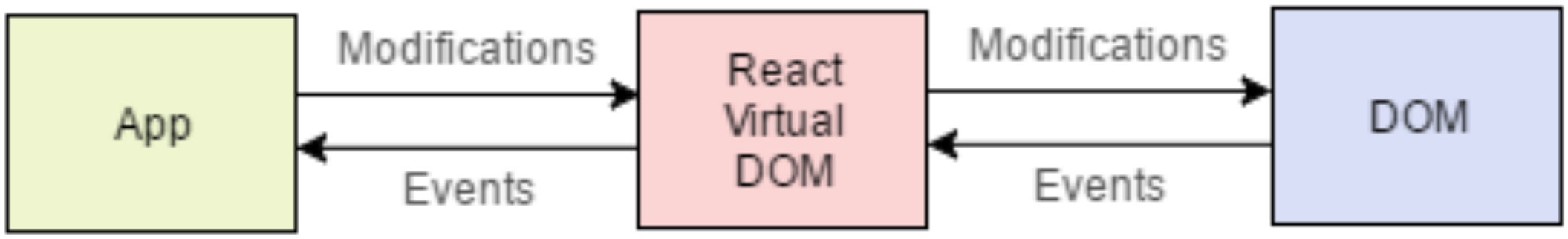
IT'S FAST

IT'S PURE

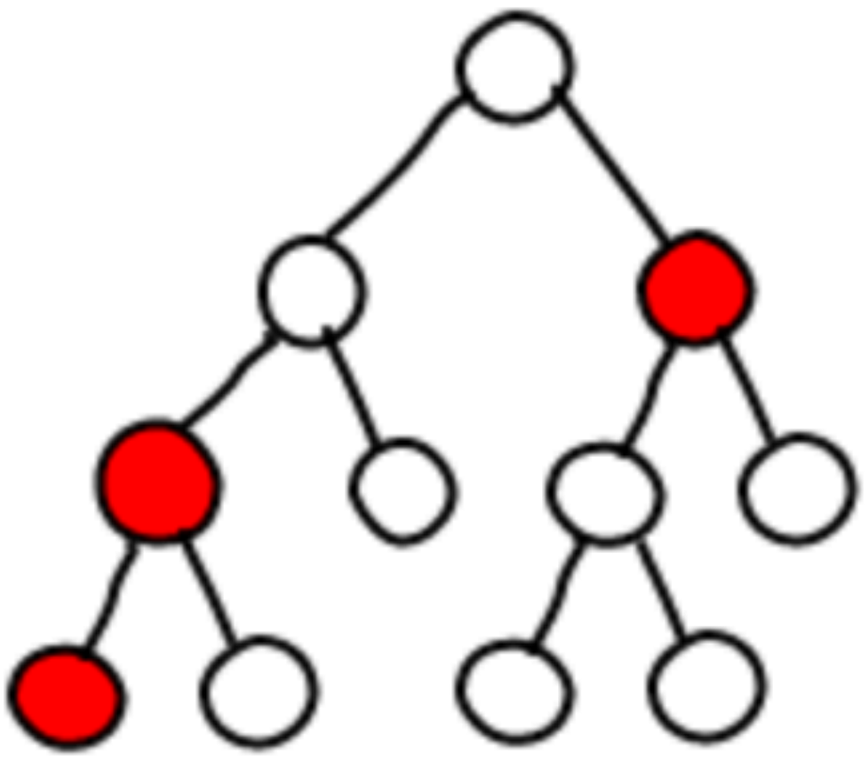
IT WORKS

Traditional Web Application

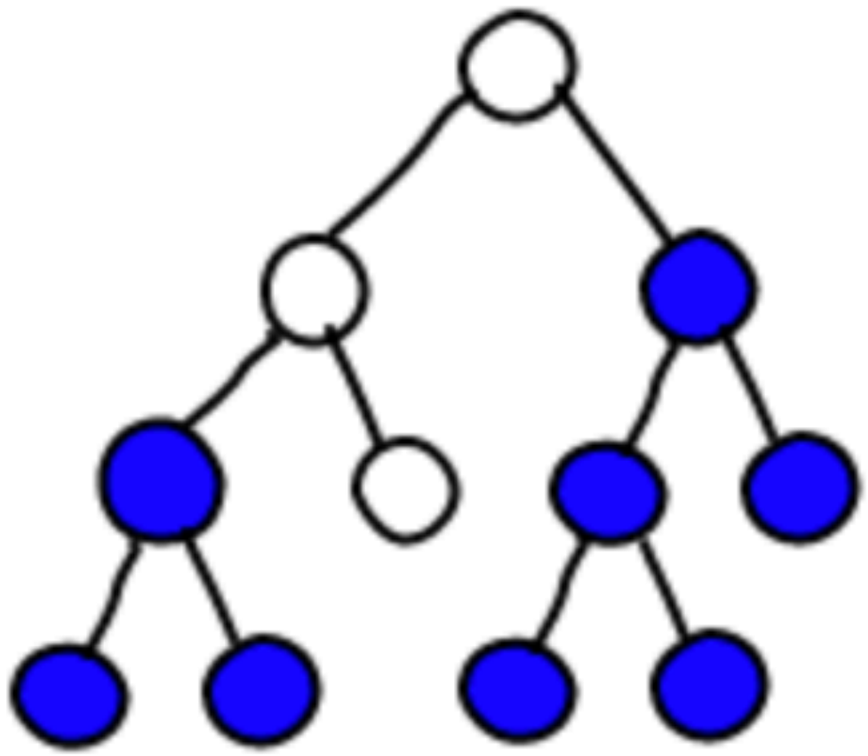
React.js



Dirty



Re-rendered



Hello, world!

It is 12:26:46 PM.

PROPS

PROPS = PRIMITIVE VALUES, REACT ELEMENTS FUNCTIONS

```
<Welcome />
```

```
props: {}
```

```
<Welcome name="Jack" />
```

```
props: {
    name: "Jack"
}
```

```
<Welcome>Jack</Welcome>
```

```
props: {
    children: "Jack"
}
```

```
const Welcome = (props) => {
    return <h1>Hello, {props.name}</h1>;
};
```

PROPS
DEFAULT IS TRUE

```jsx
<MyComponent dark />
```

```jsx
<MyComponent dark={true} />
```

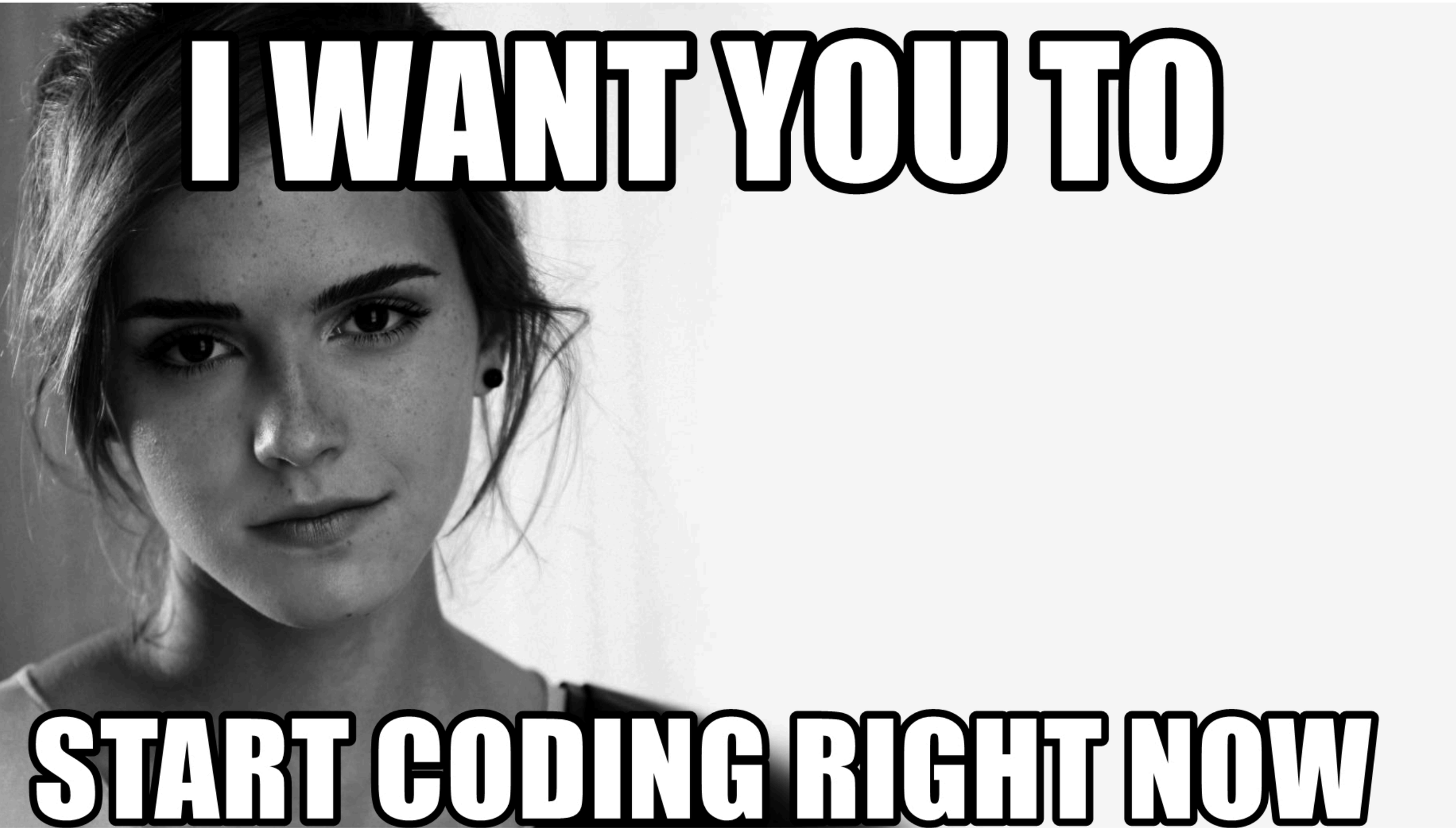
PROPS
SPREAD ATTRIBUTES

```
const postData = {
  title: "Post Title",
  image: "https://pic.com/1.jpg",
  text: "Lorem ipsum dolor sit."
};
```

```
<Post {...postData} />
```

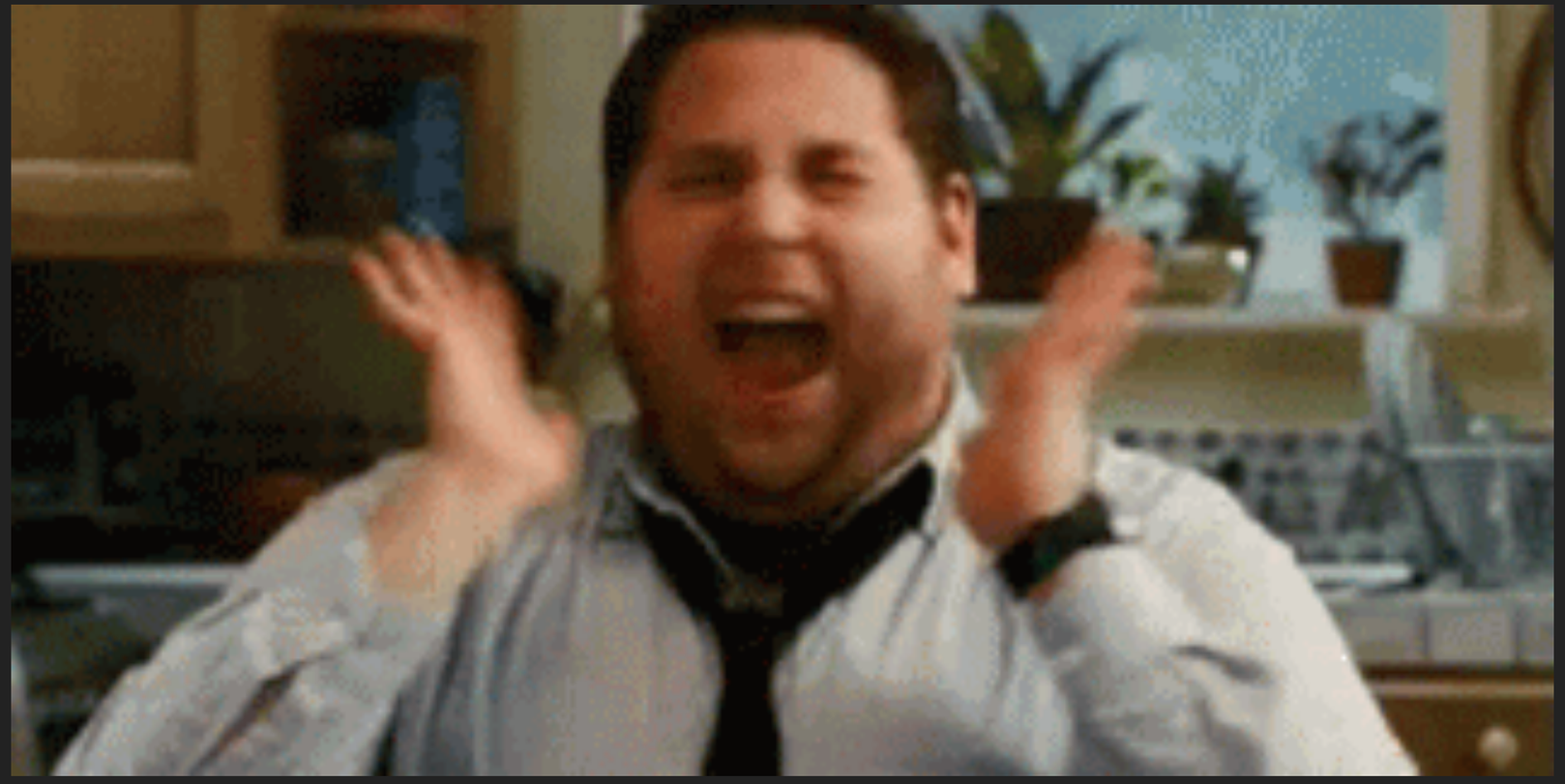
=

```
<Post
    title="Post Title"
    image="https://pic.com/1.jpg"
    text="Lorem ipsum dolor sit."
/>
```


I WANT YOU TO
START CODING RIGHT NOW

CREATE REACT APP

npx create-react-app myapp



DEMO

REACT DEV TOOLS

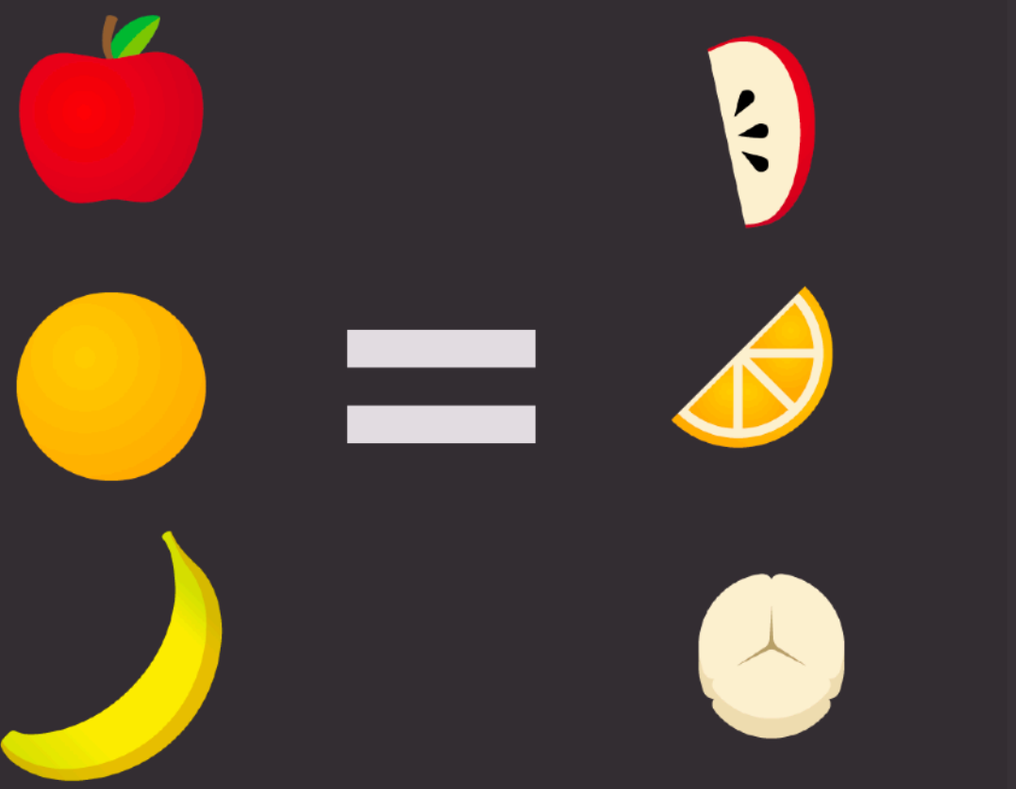
DEMO

THANKS

REACT JS

FUNCTIONAL PROGRAMMING

MAP



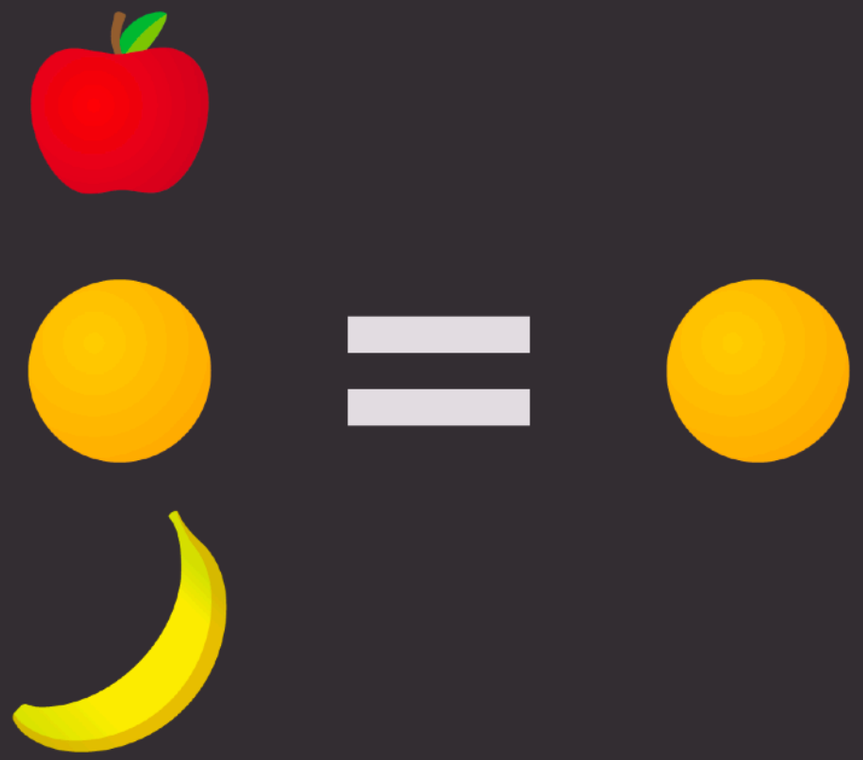
MAP

```
const array = [1,2,3,4,5];
```

```
const newArray = array.map(element => element + 1);
```

```
> newArray[2,,,,]
> newArray[2,3,,,]
> newArray[2,3,4,,]
> newArray[2,3,4,5,]
> newArray[2,3,4,5,6]
```

FILTER



FILTER

```
const array = [1,2,3,4,5];
const newArray = array.filter(element => element > 3);
```

```
> newArray[]
> newArray[]
> newArray[]
> newArray[4]
> newArray[4,5]
```

```
[🐮, 🥔, 🐔, 🌽].map(cook) ⇒ [🍔, 🍟, 🍗, 🍿]
[🍔, 🍟, 🍗, 🍿].filter(isVegetarian) ⇒ [🍟, 🍿]
[🍔, 🍟, 🍗, 🍿].reduce(eat) ⇒ 💩
```

RENDER MULTIPLE COMPONENT

RENDER MULTIPLE COMPONENT

```jsx
const App = () => (
  <div>
    <h4>Asia</h4>
    <h4>Paweł</h4>
    <h4>Gosia</h4>
    <h4>Darek</h4>
  </div>
);
```

```jsx
const App = () => (
  <div>
  {[
    <h4>Asia</h4>,
    <h4>Paweł</h4>,
    <h4>Gosia</h4>,
    <h4>Darek</h4>
  ]}
  </div>
);
```

RENDER MULTIPLE COMPONENT

```
const names = [
    "Asia",
    "Paweł",
    "Gosia",
    "Darek"
];
```

map()

```
  {[
    <h4>Asia</h4>,
    <h4>Paweł</h4>,
    <h4>Gosia</h4>,
    <h4>Darek</h4>
  ]}
```

RENDER MULTIPLE COMPONENT

```
const names = ["Asia", "Paweł", "Gosia", "Darek"];
```

```
names.map(name => <h4>{name}</h4>);
```

```
{[
  <h4>Asia</h4>,
  <h4>Paweł</h4>,
  <h4>Gosia</h4>,
  <h4>Darek</h4>
]}
```

RENDER MULTIPLE COMPONENT

```
const names = ["Asia", "Paweł", "Gosia", "Darek"];

const App = () => (
  <div>
    {names.map(name => (
      <h4>{name}</h4>
    ))}
  </div>
);
```

```
00:35:03.447 ▶Warning: Each child in an array or iterator should have  index.js:1452
a unique "key" prop.

Check the render method of `App`. See https://fb.me/react-warning-keys
for more information.
    in h4 (at App.js:10)
    in App (at src/index.js:7)
```

KEY SHOULD BE UNIQUE

NOT

~~TIMESTAMP~~

~~RANDOM~~

GOOD

```
▼ <App> == $r
  ▼ <div>
    ▼ <div className="content">
        <h4 key="0">Asia</h4>
        <h4 key="1">Paweł</h4>
        <h4 key="2">Gosia</h4>
        <h4 key="3">Darek</h4>
      </div>
    ▼ <div className="sidebar">
        <h4 key="0">Skoda</h4>
        <h4 key="1">VW</h4>
        <h4 key="2">BMW</h4>
        <h4 key="3">Porsche</h4>
      </div>
    </div>
</App>
```

ANTI PATTERN

KEY={INDEX}

BUT.....

DEMO

KEY ISN'T PASSED TO COMPONENT

NOTHING RENDER

```
<div />
```

```
<div></div>
```

```
<div>{false}</div>
```

```
<div>{null}</div>
```

```
<div>{undefined}</div>
```

```
<div>{true}</div>
```

CONDITIONAL RENDERING

CONDITIONAL RENDERING

```
const App = () => {
  if (!post) {
    return null;
  }
  return <Post {...post} />;
};
```

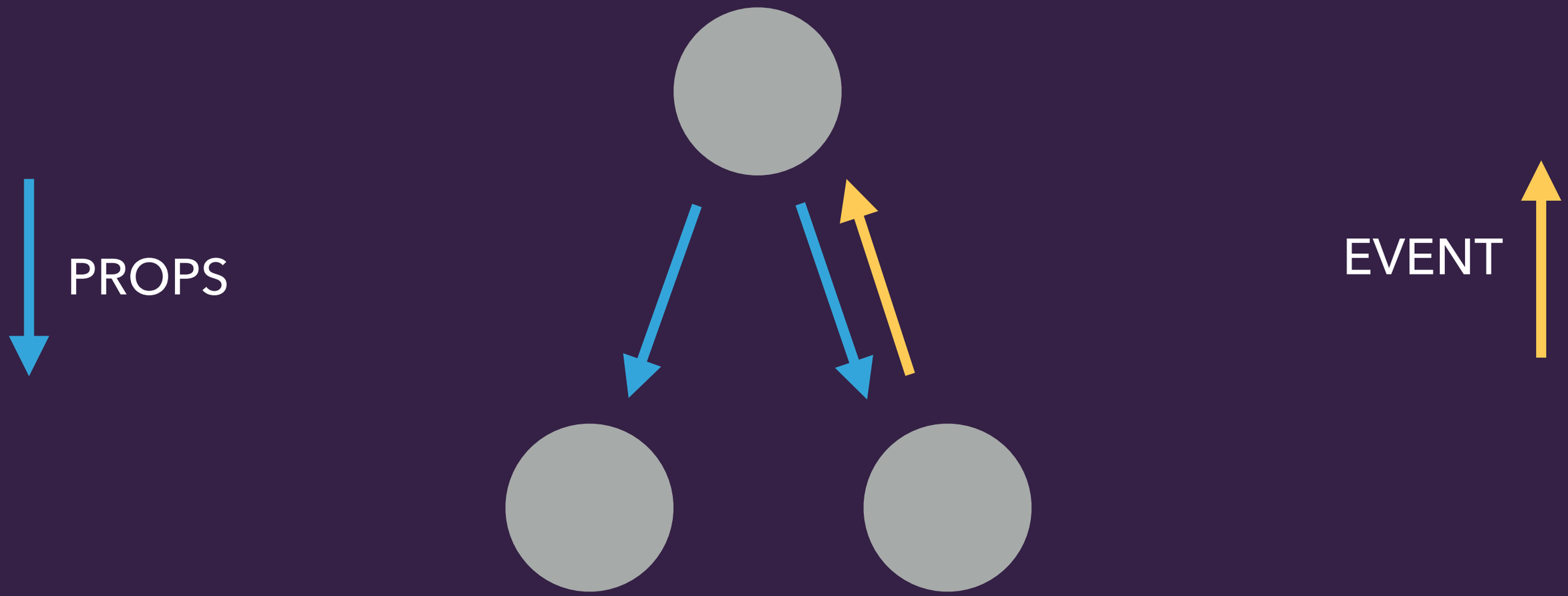
```
const App = () => {
  return post ? <Post {...post} /> : null;
};
```

CONDITIONAL RENDERING

```
const App = () => {
  return post && <Post {...post} />;
};
```

```
true && expression = expression
false && expression = false
```

ONE WAY DATA FLOW



REACT

EVENTS

event props of React element

```
// ReactElement
<button onClick={ this.handleClick }>Click!</button>
<input type="text" defaultValue="" onBlur={ this.handleBlur } />

// HTML DOM element
<button onclick="handle_click();">Click!</button>
<input type="text" value="" onblur="handle_blur();" />
```

event attributes of HTML DOM element

REACT
EVENTS

```
const App = () => (
  <button onClick={() => alert("Click Button”)}>
    Click Me!
  </button>
);
```

REACT
EVENTS

```
const App = () => {
  return <Button name="Save" />;
};
```

```
const Button = ({ name }) => (
  <button onClick={() => alert("Click Button”)}>
    {name}
  </button>
);
```

REACT
EVENTS

```
const onButtonClick = () => alert("Click Button");

const App = () => {
  return <Button name="Save" onClick={onButtonClick}/>;
};

const Button = ({ name, onClick }) => (
  <button onClick={onClick}>{name}</button>
);
```

PROPTYPES

PROPTYPES

```
import PropTypes from "prop-types";

const Hello = ({ name }) => <h1>Hello, {name}</h1>;

Hello.propTypes = {
  name: PropTypes.string
};

Hello.defaultProps = {
  name: 'unknown'
};
```

PROPTYPES

```
<Hello name={[]} />
```

```
22:52:11.651 ▶Warning: Failed prop type: Invalid prop `name` of type `array`
supplied to `Hello`, expected `string`.
    in Hello (at App.js:6)
    in App (at src/index.js:7)
```

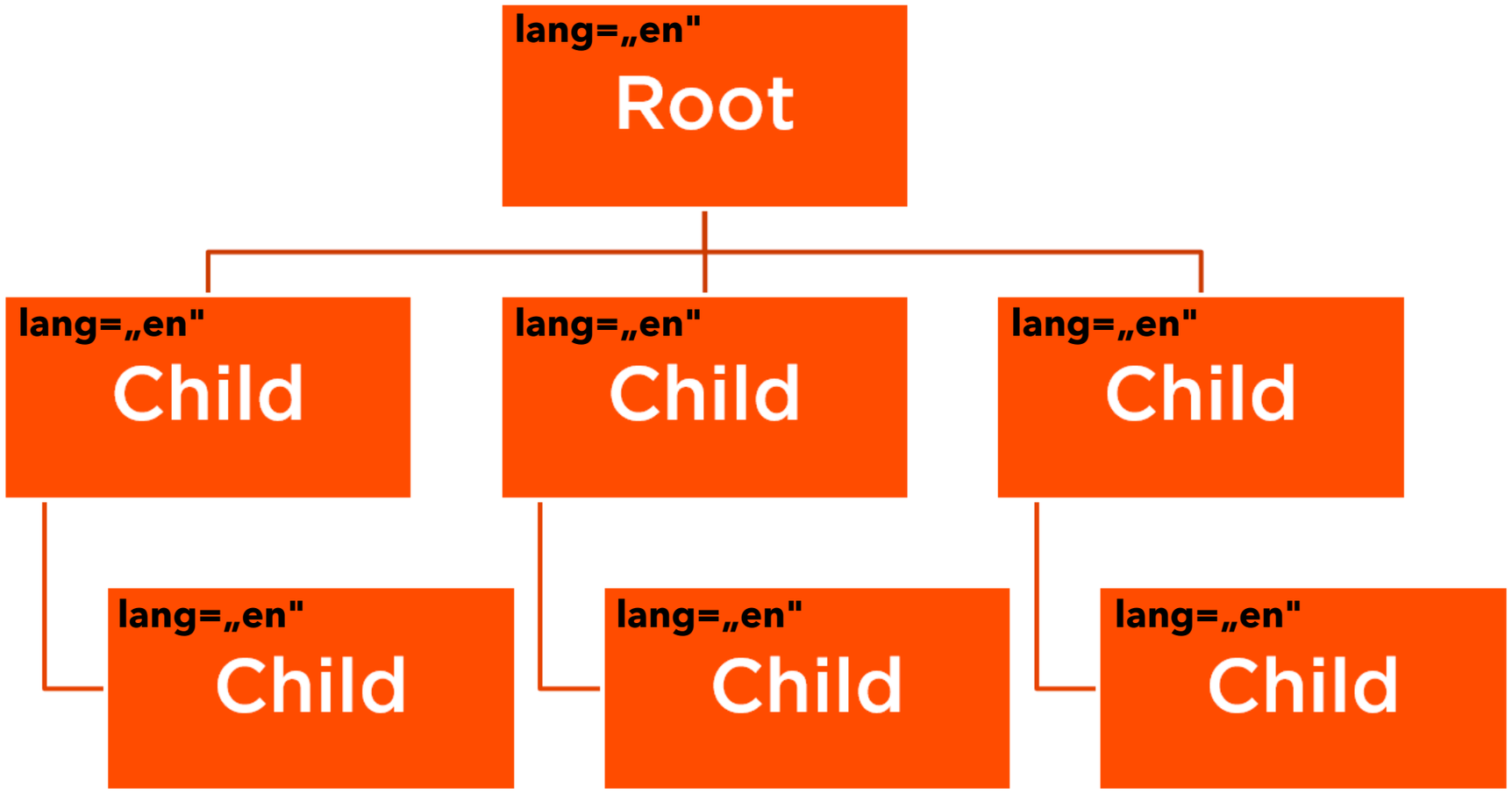
PROPTYPES

https://github.com/facebook/prop-types

CONTEXT API

CONTEXT

WHEN?


lang=„en"
Root
lang=„en"
Child
lang=„en"
Child
lang=„en"
Child
lang=„en"
Child
lang=„en"
Child
lang=„en"
Child

CONTEXT

```
const App = () => (
  <Wrapper lang="pl">
    <Content lang={props.lang}>
      <BlogPosts lang={props.lang}>
        <Post lang={props.lang} />
      </BlogPosts>
    </Content>
    <Sidebar lang={props.lang}>
      <SomeComponent lang={props.lang}>
        <NewsletterForm lang={props.lang} />
      </SomeComponent>
    </Sidebar>
  </Wrapper>
);
```

CONTEXT

```
const Context = React.createContext();
```

```
<Context.Provider />
```

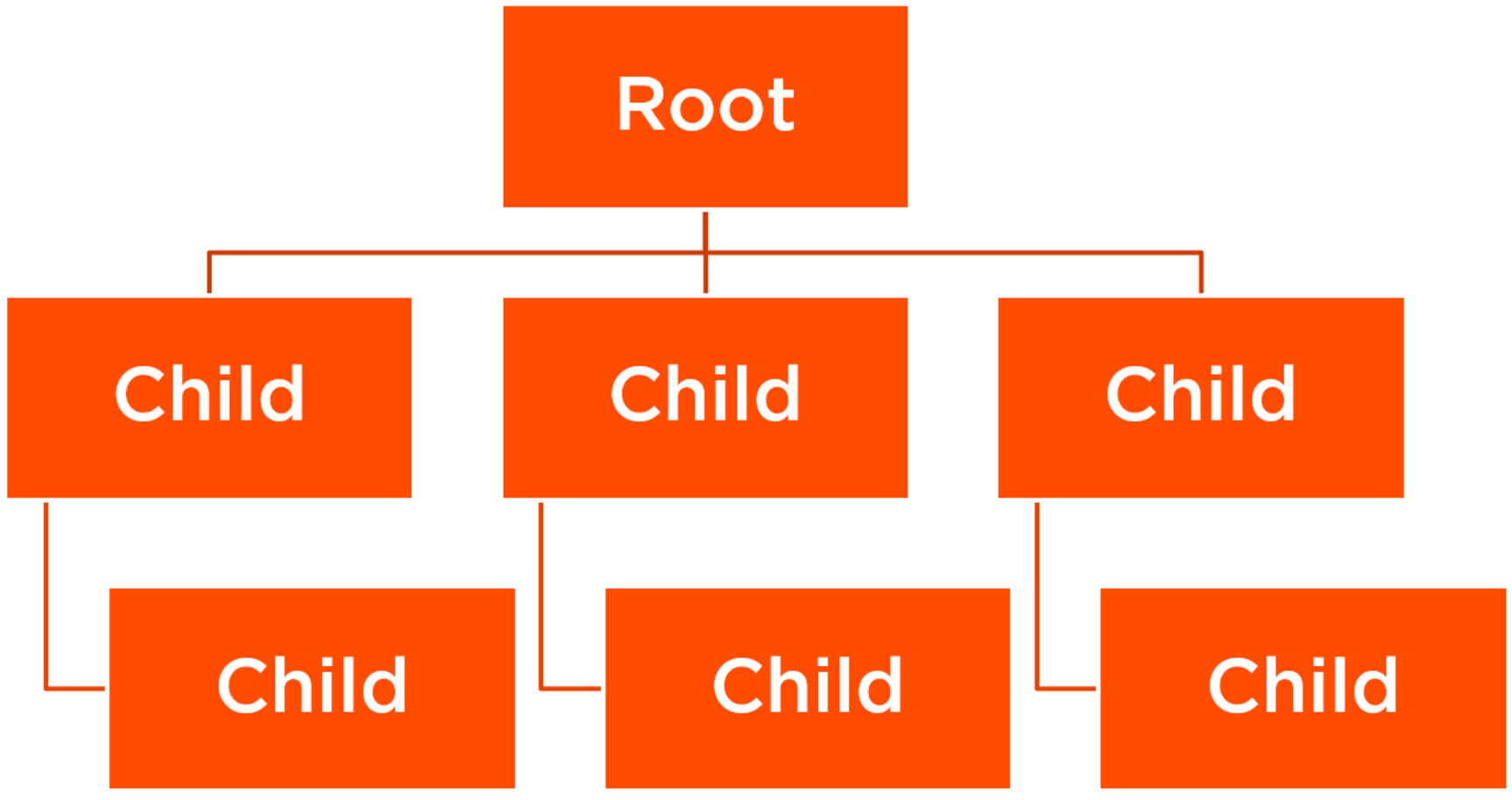
```
<Context.Consumer />
```

<PROVIDER>

<CONSUMER>

Language

Context



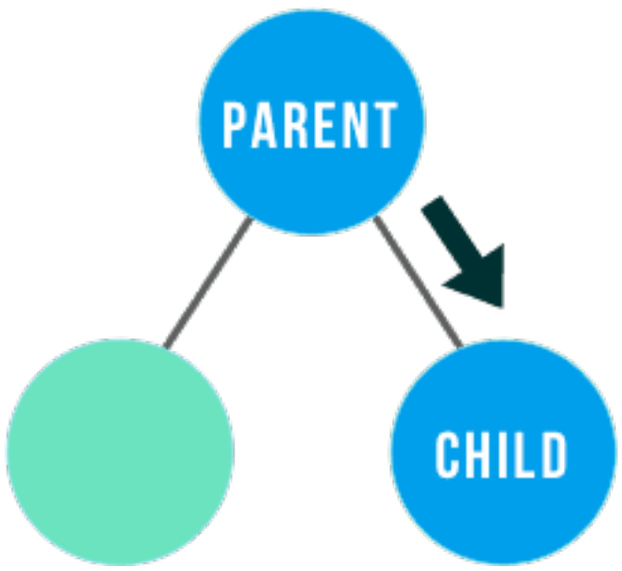
```jsx
const LanguageContext = React.createContext();

const App = () => (
  <LanguageContext.Provider value="pl">
    <Wrapper>
      <Content>
        <BlogPosts>
          <LanguageContext.Consumer>
            {lang => <Post lang={lang} />}
          </LanguageContext.Consumer>
        </BlogPosts>
      </Content>
      <Sidebar>
        <SomeComponent>
          <LanguageContext.Consumer>
            {lang => <NewsletterForm lang={lang} />}
          </LanguageContext.Consumer>
        </SomeComponent>
      </Sidebar>
    </Wrapper>
  </LanguageContext.Provider>
);
```

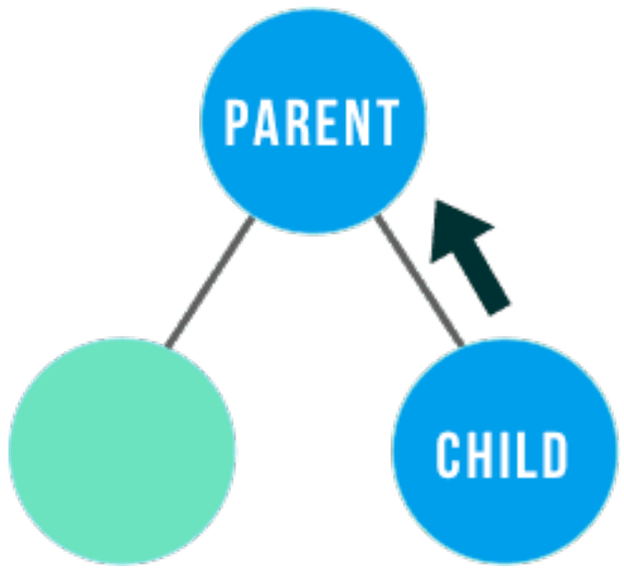
DEMO

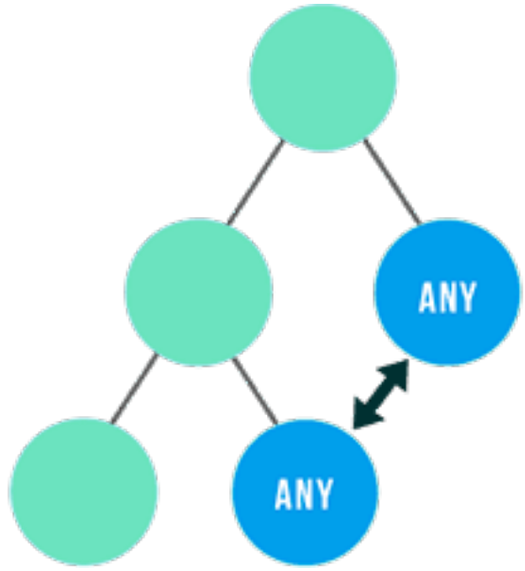
THANKS

REACT JS

SUMMARY







PROPS

EVENTS

CONTEXT

HOC

HIGH ORDER COMPONENT

HOC

```
component => componentOnSteroids;
```

HOC

```
const EnhancedComponent = higherOrderComponent(WrappedComponent);
```

HOC

```
const App = () => (
    <>
        <ComponentA name="Kasia" />
        <ComponentB name="Asia" />
    </>
);
```

HOC

```
const withLanguage = WrappedComponent => {
  return props => (
    <WrappedComponent {...props} language="pl" />
  );
};
```

```
const ComponentALang = withLanguage(ComponentA);
const ComponentBLang = withLanguage(ComponentB);

const App = () => (
  <>
    <ComponentALang name="Kasia" />
    <ComponentBLang name="Asia" />
  </>
);
```

HOC

DEMO

HOC

DON'T MUTATE THE ORIGINAL COMPONENT.

USE COMPOSITION.

HOC

```
const EnhancedComponent = hoc(Component);

const EnhancedComponent = anotherHoc(Component, config);
```

```
const ConnectedComponent = connect(param1, param2)(Component);

const enhance = connect(param1, param2);
const ConnectedComponent = enhance(Component);
```

HOC

```
const EnhancedComponent = withRouter(connect(param)(WrappedComponent))
```

```
const enhance = compose(
    withRouter,
    connect(commentSelector)
)
const EnhancedComponent = enhance(WrappedComponent)
```

HOC

react-fns

https://github.com/jaredpalmer/react-fns

HOC
BUT...

Not very intuitive to implement

Lots of code

Wrapper hell

```
▼ <Route>
  ▼ <main>
    ▼ <TransitionGroup component={null}>
      ▼ <CSSTransition key=".0" timeout={0} in={true}>
        ▼ <Transition timeout={0} in={true} exit={true} enter={true} mountOnEnter={false} unmountOnExit={false} appear={false}>
          ▼ <div className="page-wrap">
            ▼ <Route>
              ▼ <Switch>
                ▼ <AuthorisedRoute exact={true} path="/">
                  ▼ <Connect(RoutePrivate) path="/" exact={true}>
                    ▼ <RoutePrivate path="/" exact={true} token="eyJraWQiOiJjZGVFei1qTzFNNmhsT3BCUzM5VjJkQWRocU9oZ1…" isRefreshing={false} to="/login">
                      ▼ <Route path="/" exact={true} to="/login">
                        ▼ <withRouter(ScrollToTopRoute)>
                          ▼ <Route>
                            ▼ <ScrollToTopRoute>
                              ▼ <LoadableComponent>
                                ▼ <withRouter(Connect(Dashboard))>
▼ <Route>
  ▼ <Connect(Dashboard)>
    ▼ <Dashboard>
      ▼ <Section home={true} color="yellow-dark">
        ▼ <div className="section section--yellow-dark section--home">
          ▼ <div className="section__inner">
            ▼ <Wrapper>
              ▼ <Container tag="div">
                ▼ <div className="container">
                  ▼ <Row className="justify-content-center" tag="div">
                    ▼ <div className="justify-content-center row">
                      ▼ <Col md={15} xl={10} tag="div">
                        ▼ <div className="col-md-15 col-xl-10">
                            <h1>Welcome 123_test,</h1>
                          ▶ <Connect(QuickActionContainer)>…</Connect(QuickActionContainer)>
```


There’s something you should know.

COMPONENT

BASIC LIFECYCLE

MOUNTING

UPDATING

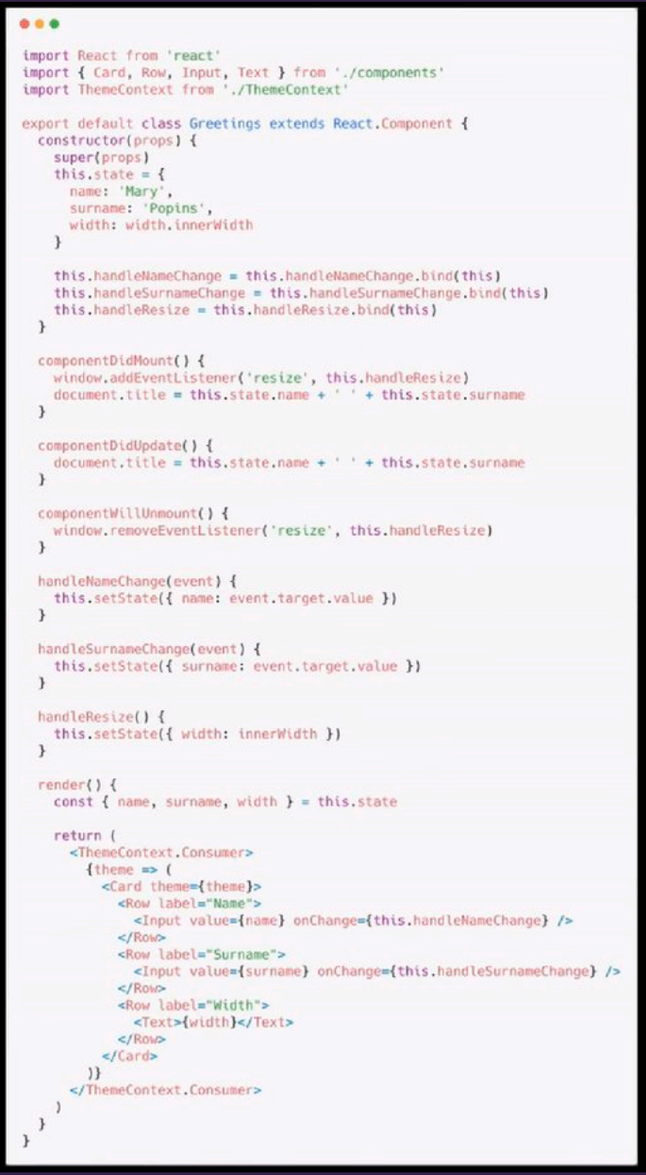
UNMOUNTING

Hooks

HOOKS

Hooks is a special function that lets you “hook into” React features like React state and lifecycle.

HOOKS



```
import React from 'react'
import { Card, Row, Input, Text } from './components'
import ThemeContext from './ThemeContext'

export default class Greetings extends React.Component {
  constructor(props) {
    super(props)
    this.state = {
      name: 'Mary',
      surname: 'Popins',
      width: width.innerWidth
    }

    this.handleNameChange = this.handleNameChange.bind(this)
    this.handleSurnameChange = this.handleSurnameChange.bind(this)
    this.handleResize = this.handleResize.bind(this)
  }

  componentDidMount() {
    window.addEventListener('resize', this.handleResize)
    document.title = this.state.name + ' ' + this.state.surname
  }

  componentDidUpdate() {
    document.title = this.state.name + ' ' + this.state.surname
  }

  componentWillUnmount() {
    window.removeEventListener('resize', this.handleResize)
  }

  handleNameChange(event) {
    this.setState({ name: event.target.value })
  }

  handleSurnameChange(event) {
    this.setState({ surname: event.target.value })
  }

  handleResize() {
    this.setState({ width: innerWidth })
  }

  render() {
    const { name, surname, width } = this.state

    return (
      <ThemeContext.Consumer>
        {theme => (
          <Card theme={theme}>
            <Row label="Name">
              <Input value={name} onChange={this.handleNameChange} />
            </Row>
            <Row label="Surname">
              <Input value={surname} onChange={this.handleSurnameChange} />
            </Row>
            <Row label="Width">
              <Text>{width}</Text>
            </Row>
          </Card>
        )}
      </ThemeContext.Consumer>
    )
  }
}
```

```
import React from 'react'
import { Card, Row, Input, Text } from './components'
import ThemeContext from './ThemeContext'

export default class Greetings extends React.Component {
  constructor(props) {
    super(props)
    this.state = {
    }

  }

  componentDidMount() {
  }

  componentDidUpdate() {
  }

  componentWillUnmount() {
  }

  render() {
    const { name, surname, width } = this.state

    return (
          <Card theme={theme}>
            <Row label="Name">
              <Input value={name} onChange={this.handleNameChange} />
            </Row>
            <Row label="Surname">
              <Input value={surname} onChange={this.handleSurnameChange} />
            </Row>
            <Row label="Width">
              <Text>{width}</Text>
            </Row>
          </Card>
    )
  }
}
```

```
import React, { useState, useContext, useEffect } from 'react'
import { Card, Row, Input, Text } from './components'
import ThemeContext from './ThemeContext'

export default function Greetings(props) {

  return (
    <Card theme={theme}>
      <Row label="Name">
        <Input {...name} />
      </Row>
      <Row label="Surname">
        <Input {...surname} />
      </Row>
      <Row label="Width">
        <Text>{width}</Text>
      </Row>
    </Card>
  )
}
```

```
import React, { useState, useContext, useEffect } from 'react'
import { Card, Row, Input, Text } from './components'
import ThemeContext from './ThemeContext'

function useFormInput(initialValue) {
  const [value, setValue] = useState(initialValue)

  function handleChange(e) {
    setValue(e.target.value)
  }

  return {
    value,
    onChange: handleChange,
  }
}

function useDocumentTitle(title) {
  useEffect(() => {
    document.title = title
  })
}

function useWindowWidth() {
  const [width, setWidth] = useState(window.innerWidth)
  useEffect(() => {
    const handleResize = () => setWidth(window.innerWidth)
    window.addEventListener('resize', handleResize)
    return () => {
      window.removeEventListener('resize', handleResize)
    }
  })

  return width
}

export default function Greetings(props) {
  const name = useFormInput('Mary')
  const surname = useFormInput('Poppins')
  const theme = useContext(ThemeContext)
  const width = useWindowWidth()

  useDocumentTitle(name.value + ' ' + surname.value)

  return (
    <Card theme={theme}>
      <Row label="Name">
        <Input {...name} />
      </Row>
      <Row label="Surname">
        <Input {...surname} />
      </Row>
      <Row label="Width">
        <Text>{width}</Text>
      </Row>
    </Card>
  )
}
```

HOOKS

Functional Stateless Components

now is

Functional Components

HOOKS

Don’t waist your time to rewriting your existing components overnight but you can start using Hooks in the new ones if you’d like

HOOKS

Don’t call Hooks inside loops, conditions, or nested functions

Only call Hooks at the top level.

Don’t call Hooks from regular JavaScript functions

Only call Hooks from React function components.

HOOKS
useState

useState is a Hook that lets you add React state to function components

CURRE
useState

```
const [count, setCount] = useState(0);
```

current value

update function

Initial value

HOOKS
useState

```jsx
const App = () => {
  const [count, setCount] = useState(0);

  return (
    <div>
      <h2>Counter: {count}</h2>
    </div>
  );
};
```

HOOKS
useState

```jsx
const App = () => {
  const [count, setCount] = useState(0);
  return (
    <div>
      <h2>Counter: {count}</h2>
      <button onClick={() => setCount(count + 1)}>Click</button>
    </div>
  );
};
```

HOOKS
useState

```
const App = () => {
  //multiple state variables!
  const [count, setCount] = useState(0);
  const [name, setName] = useState("");
  const [email, setEmail] = useState({});
```

HOOKS

useState

DEMO

HOOKS
useContext

Accepts a context object and returns the current value for that context.

```
const value = useContext(MyContext);
```

HOOKS

useContext

DEMO

HOOKS
useEffect

Accepts a function that contains imperative, possibly effectful code.

```
useEffect(() => {
  // add listener for feature (setup)
  // return function to remove listener for feature (clean up)
});
```

HOOKS
useEffect

```
useEffect(() => {
  window.document.title = `Counter: ${count}`
});
```

HOOKS

useEffect

```
useEffect(() => {
});
```

Always

```
useEffect(() => {
}, []);
```

Only Once

```
useEffect(() => {
}, [count]);
```

When deps change

HOOKS
useEffect

```
useEffect(() => {
   ...
   return () => {
      console.log('Component will unmount')
   }
});
```

HOOKS
useEffect

DEMO

HOOKS
useMemo

```
useMemo(() => {
},[]);
```

factory

dependency

HOOKS
useMemo

```
const memoizedValue = useMemo(
  () => {
    return computeExpensiveValue(a, b);
  },
  [a, b]
);
```

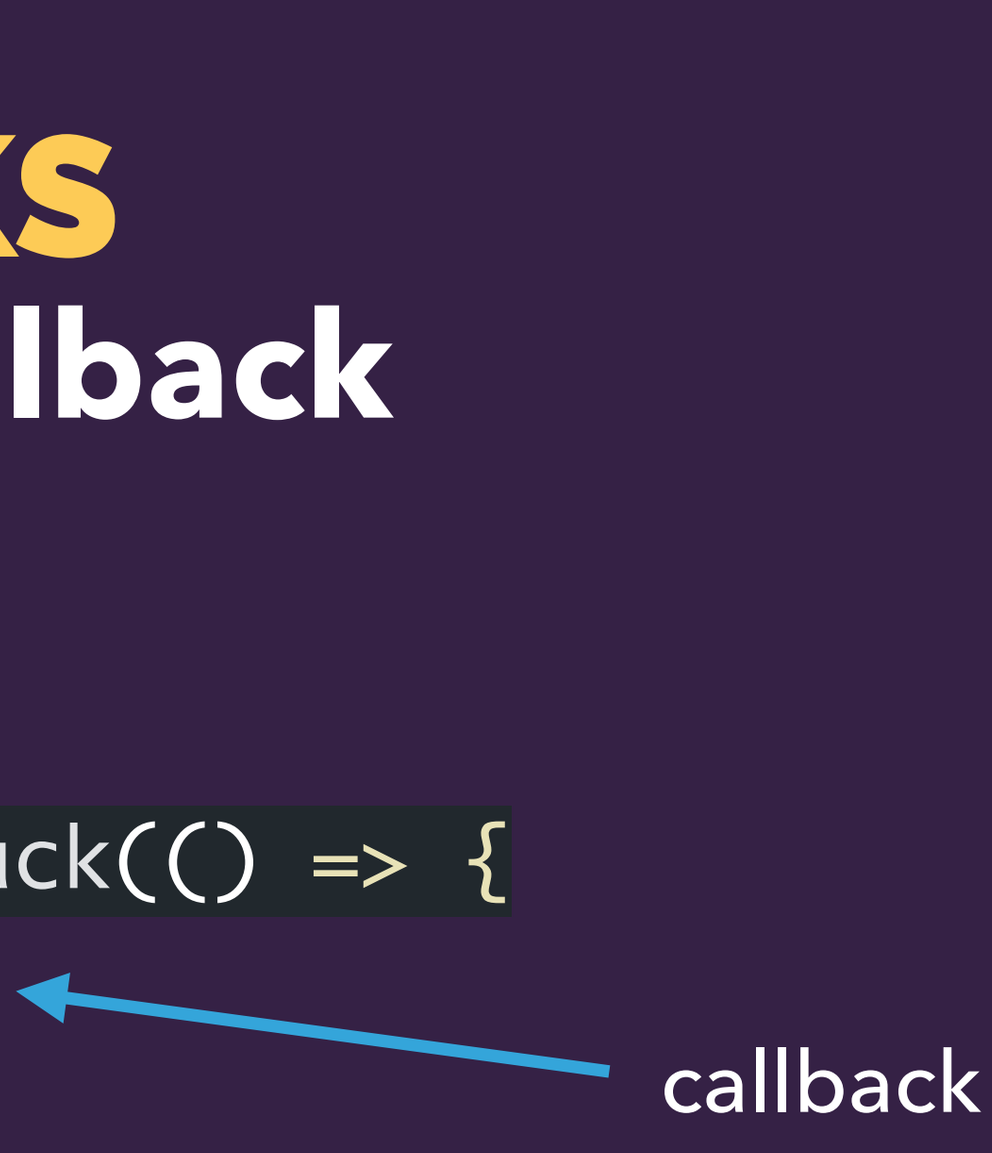
HOOKS

useMemo

DEMO

HOOKS
useCallback

```
useCallback(() => {
},[]);
```



HOOKS
useCallback

```
const removeFromCart = () => removeItem(product.id);

return (
    <Button onClick={removeFromCart}>Delete</Button>
);
```

HOOKS
useCallback

```
const removeFromCart = useCallback(
  () => {
      removeItem(product.id);
  },
  [product.id]
);

return (
    <Button onClick={removeFromCart}>Delete</Button>
);
```

React Hooks Lifecycle

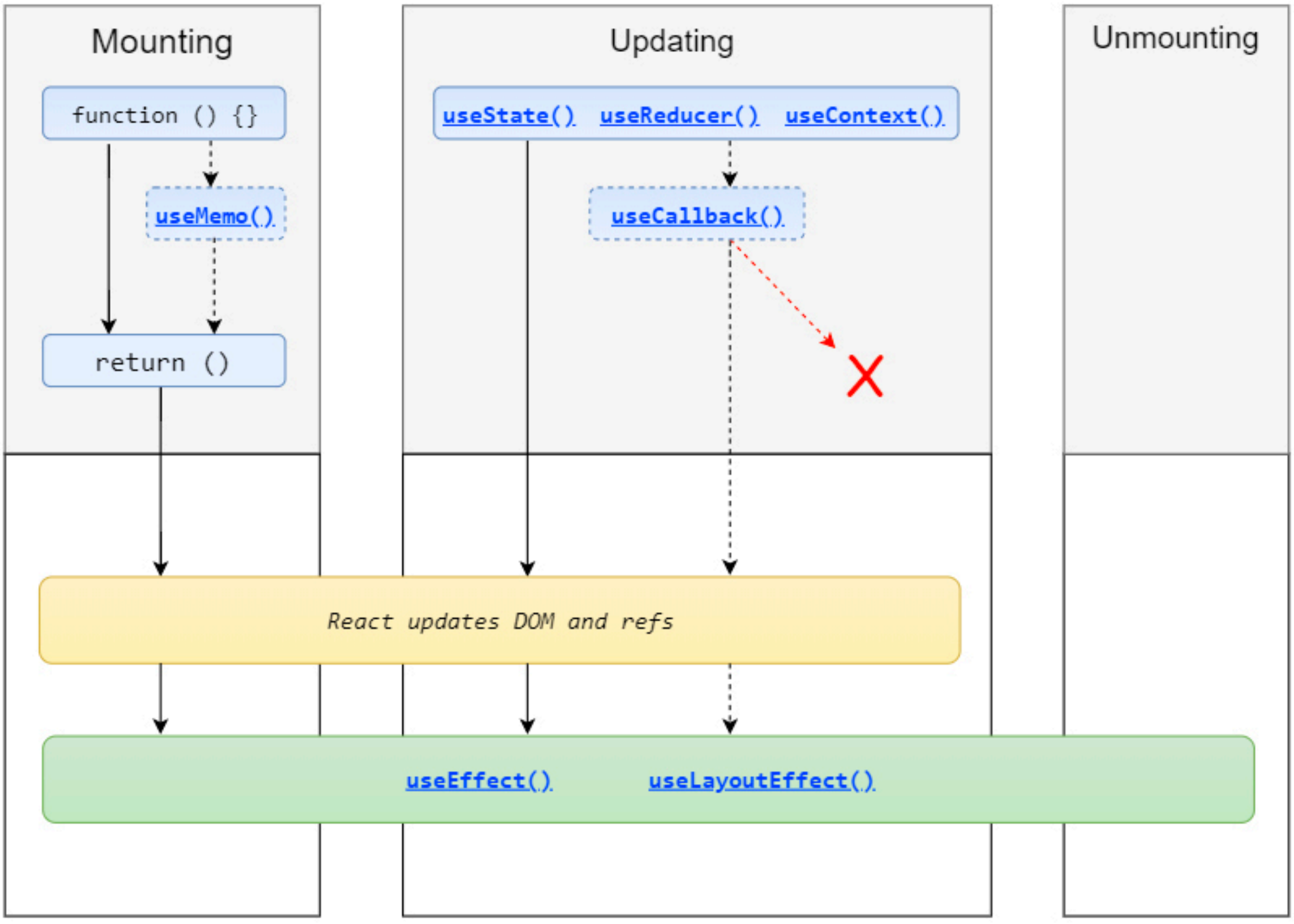
version: latest, 16.9.x, 16.8.x

"Render phase"

Pure and has no side effects. May be paused, aborted or restarted by React

"Commit phase"

Can work with DOM, run side effects, schedule updates.



REACT JS

REFS

Refs make it possible to access DOM nodes directly within React.

REFS

When use?

Managing focus, text selection, or media playback.

Triggering imperative animations.

Integrating with third-party DOM libraries.

HOOKS
useRef

```jsx
const App = () => {
  const inputRef = useRef(null);

  const onButtonClick = () => {
    inputRef.current.focus();
  };

  return (
    <>
      <input ref={inputRef} type="text" />
      <button onClick={onButtonClick}>Focus the input</button>
    </>
  );
};
```

HOOKS

useRef

DEMO

HOOKS
useYourImagination

Write custom hook

HOOKS

useYourImagination

useCounter

HOOKS
useCounter

```
const useCounter = () => {
  const [count, setCount] = useState(0);

  useEffect(
    () => {
      window.document.title = `Count ${count}`;
    },
    [count]
  );

  const increase = () => setCount(count + 1);

  return [count, increase];
};
```

HOOKS
useCounter

```
const App = () => {
  const [count, increase] = useCounter();

  return (
    <div>
      <h2>Count {count}</h2>
      <button onClick={() => increase()}>Click</button>
    </div>
  );
};
```

STYLE REACT COMPONENT

96* Ways to Style React Components in 2019

INLINE CSS

INLINE CSS

```
const helloStyle = {
  color: "red",
  fontSize: "20px"
};

const Hello = () => <h1 style={helloStyle}>Hello</h1>;
```

CLASS NAME

CLASS NAME

Style.css

```css
.red {
  color: red;
  font-size: 20px
}
```

App.js

```jsx
import "./Style.css";

const Hello = () => <h1 className="red">Hello</h1>;
```

CSS MODULES

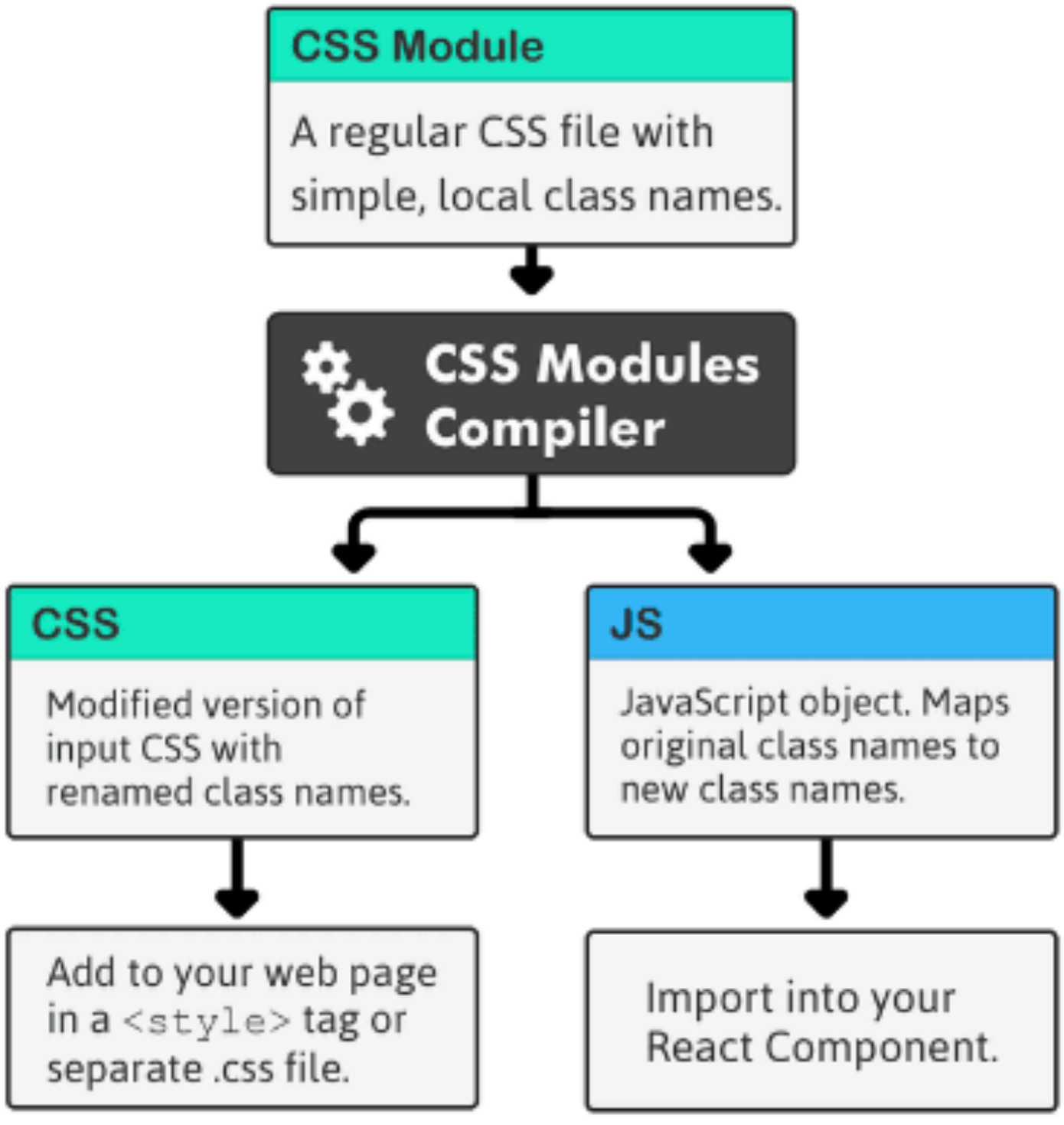
CSS MODULES

Style.module.css

```
.Red {
  color: red;
  font-size: 20px
}
```

App.js

```
import styles from "./Style.module.css";

const Hello = () => <h1 className={styles.Red}>Hello</h1>;
```


CSS Module
A regular CSS file with simple, local class names.
CSS Modules Compiler
CSS
Modified version of input CSS with renamed class names.
Add to your web page in a <style> tag or separate .css file.
JS
JavaScript object. Maps original class names to new class names.
Import into your React Component.

CSS MODULES

Style.module.css

```
.Red {
  color: red;
  font-size: 20px
}
```

```
<style type="text/css">.Style_Red___55GV {
  color: red;
  font-size: 20px
}</style>
```

App.js

```
import styles from "./Style.module.css";
```

console.log(styles);

```
{
  Red: "Style_Red___55GV"
};
```

```
const Hello = () => <h1 className={styles.Red}>Hello</h1>;
```

CSS IN JS

STYLED COMPONENTS

```
import styled from 'styled-components';

const Hello = styled.h1`
  color: red;
  font-size: 20px
`;

const App = ()  => <Hello>Hello world</Hello>;
```

STYLED COMPONENTS

```
const Hello = styled.h1`
  color: red;
  font-size: 20px
`;
```

```
const App = ()  => <Hello>Hello world</Hello>;
```

```
<style data-styled="" data-styled-version="4.1.3">
  /* sc-component-id: sc-bdVaJa */
  .cSRbkb {
    color:red;
    font-size:20px;
  }
</style>

<h1 class="sc-bdVaJa cSRbkb”>Hello world</h1>
```

STYLED COMPONENTS

```
const Hello = styled.h1`
  color: ${props => props.color};
  font-size: 20px;
`;
```

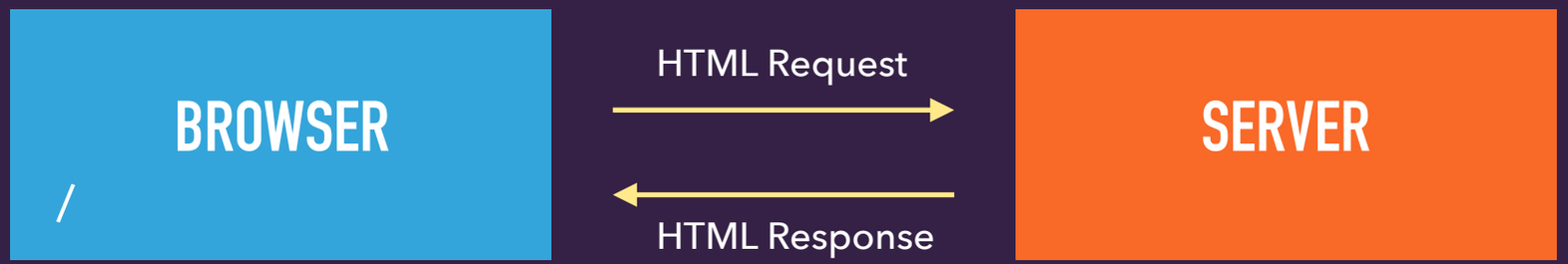
```
<Hello color="red">Hello world</Hello>;
```

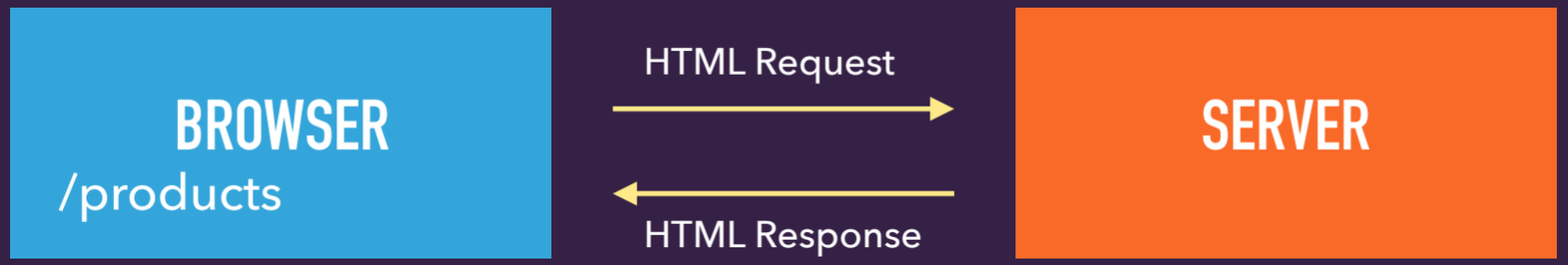
Other libaries...

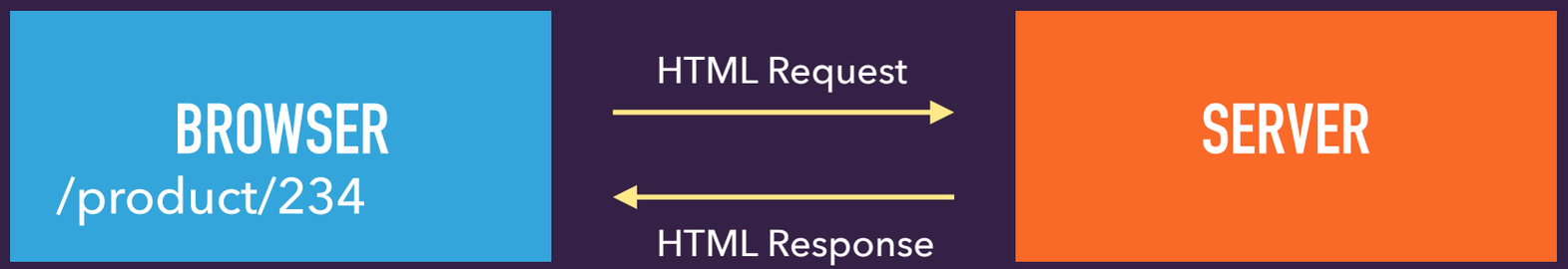
https://github.com/tuchk4/awesome-css-in-js

ROUTING

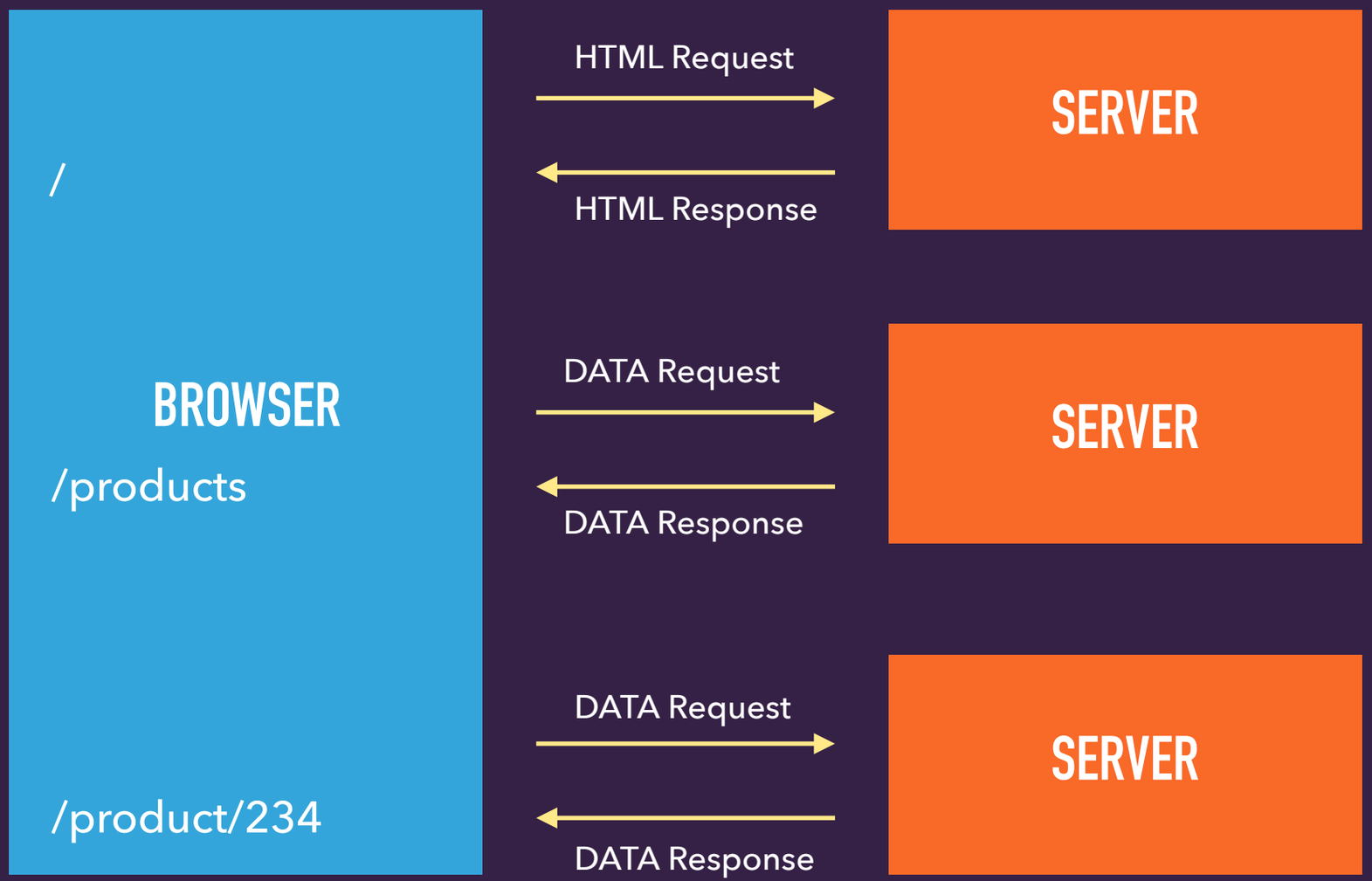
TRADITIONAL WEBSITE







SINGLE PAGE APPLICATION



STATIC ROUTING

DYNAMIC ROUTING

„When we say dynamic routing, we mean routing that takes place as your app is rendering, not in a configuration or convention outside of a running app. That means almost everything is a component in React Router.“

— React Training

localhost:3000

Website

/

footer

localhost:3000/catalog

Website

Catalog Component

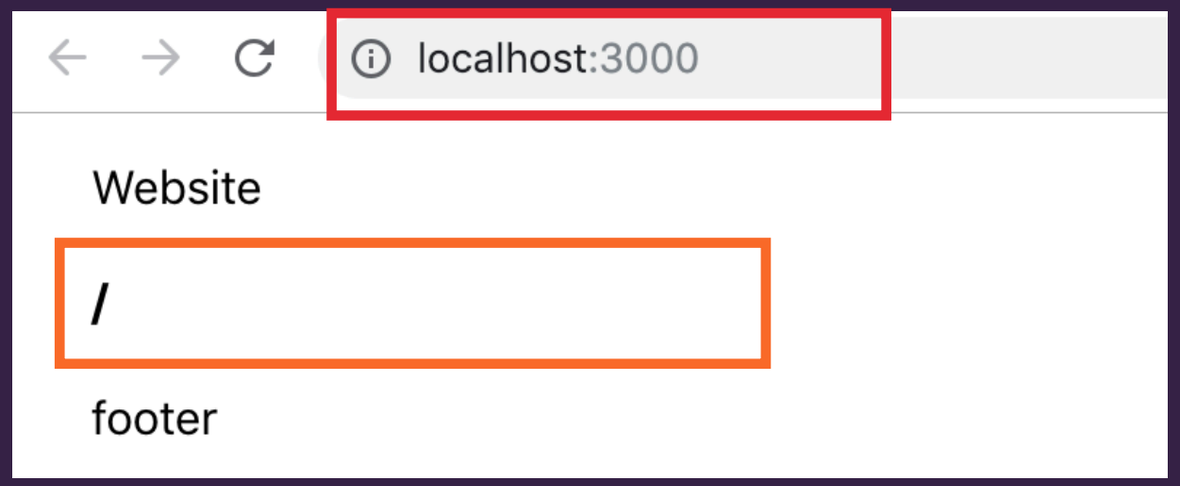
footer

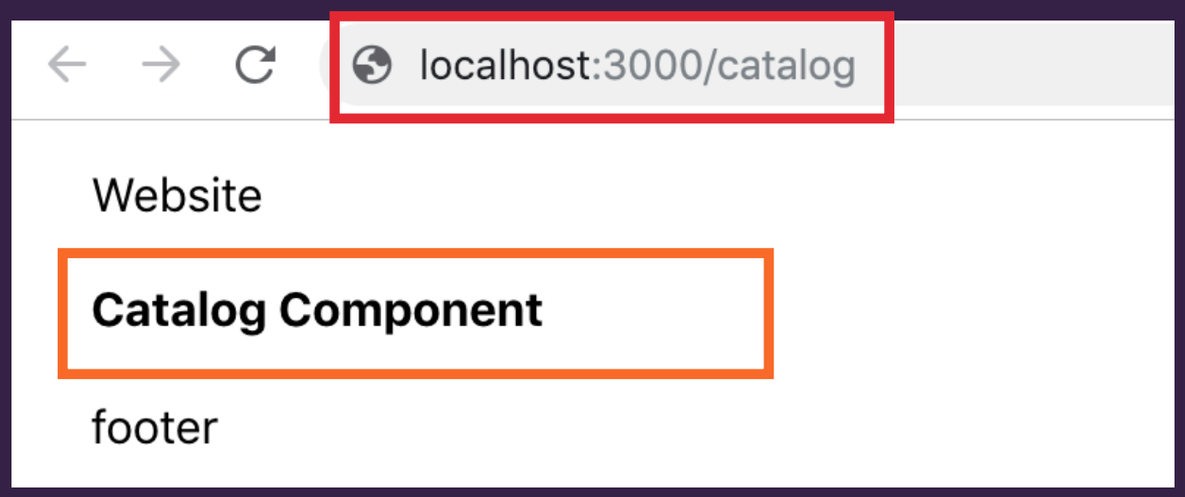
localhost:3000/contact

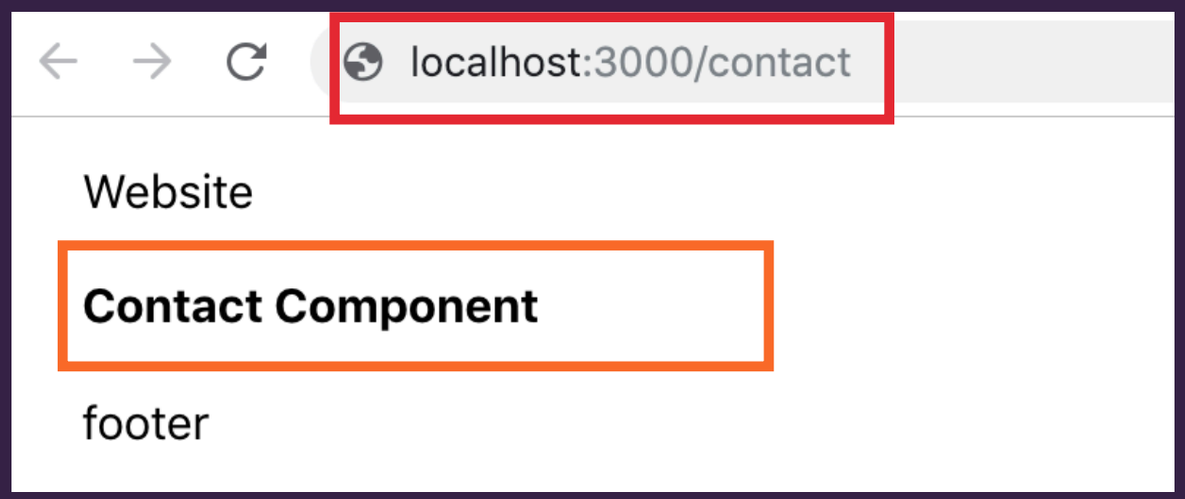
Website

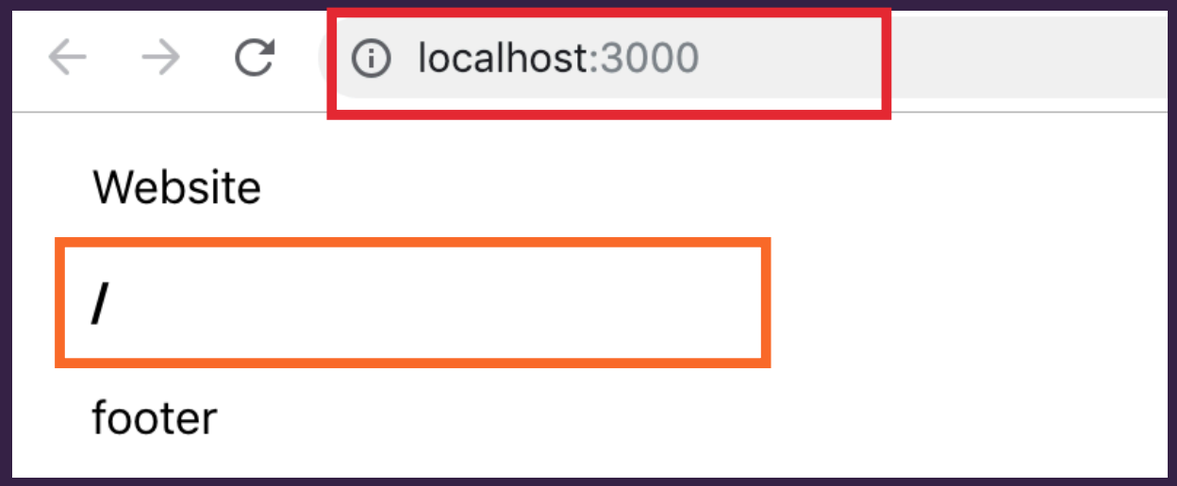
Contact Component

footer

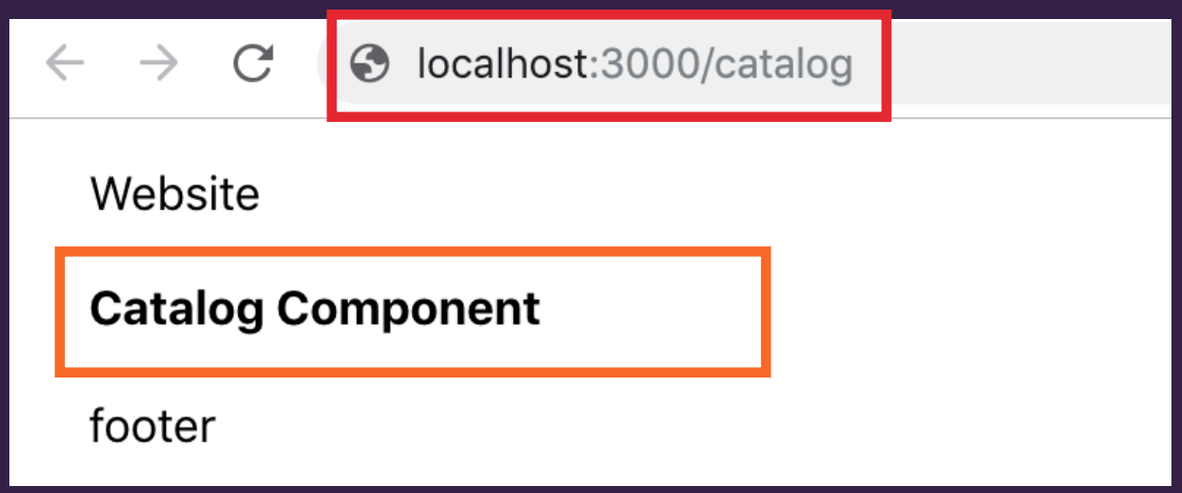

localhost:3000
Website
/
footer


localhost:3000/catalog
Website
Catalog Component
footer

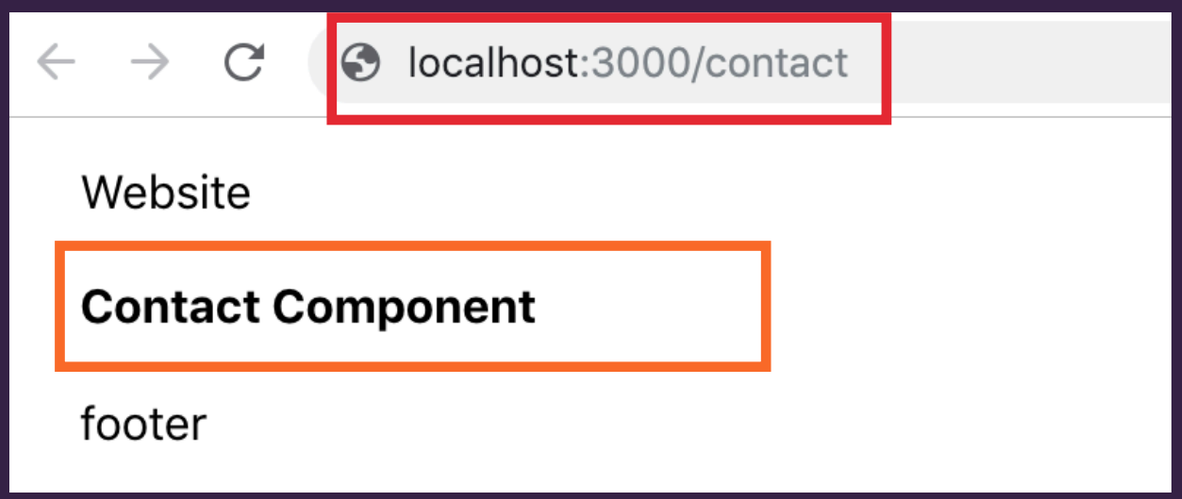

localhost:3000/contact
Website
Contact Component
footer



path: /
component: Home



path: /catalog
component: Catalog



path: /contact
component: Contact

REACT ROUTER V5

PACKAGES

REACT-ROUTER The core of React Router

REACT-ROUTER-DOM DOM bindings for React Router

REACT-ROUTER-NATIVE React Native bindings for React Router

REACT ROUTER API

<Router>

```
<Router history={history}>
  <App />
</Router>
```

REACT ROUTER API

<BrowserRouter>

```
<BrowserRouter>
  <App />
</BrowserRouter>
```

REACT ROUTER API

<BrowserRouter>

example.com

example.com/catalog

example.com/contact

REACT ROUTER API

<HashRouter>

```
<HashRouter>
  <App />
</HashRouter>
```

REACT ROUTER API

<HashRouter>

example.com

example.com/#/catalog

example.com/#/contact

REACT ROUTER API

<MemoryRouter>

```
<MemoryRouter>
  <App />
</MemoryRouter>
```

REACT ROUTER API

<Route>

```
<Router>
  <div>
    <Route path="/" component={Home}/>
    <Route path="/catalog" component={Catalog}/>
    <Route path="/contact" component={Contact}/>
  </div>
</Router>
```

REACT ROUTER API

<Route>

```
<Route exact path="/" component={Home}/>
<Route strict path="/catalog/" component={Catalog}/>
<Route sensitive path="/Contact" component={Contact}/>
```

REACT ROUTER API

<Route>

```
<Route path="/" component={Home}/>
<Route path="/" render={() => <Home />}/>
<Route path="/" children={() => <Home />}/>
```

PARAMETERS ROUTERS

```
<Route path="/catalog/:id" component={Catalog} />
```

```
const Catalog = ({ match }) => (
  <div>
    <h1>I am Catalog Component, id: {match.params.id}</h1>
  </div>
);
```

REACT ROUTER API

<Route>

```
<Router>
  <div>
    <Route path="/" component={Home}/>
  </div>
</Router>


const Home = ({ match, location, history }) => (
  ……
)
```

REACT ROUTER API

<Link>

```
<Link to="/contact">Contact</Link>
```

```
<Link
  to={{
    pathname: "/contact",
    search: „?from=pl",
    hash: "#hash",
    state: { fromDashboard: true }
  }}
/>
```

REACT ROUTER API

<NavLink>

```
<NavLink to="/contact" activeClassName="selected">
  Contact
</NavLink>

<NavLink to=„/contact" activeStyle={{color: "red"}}>
  Contact
</NavLink>
```

REACT ROUTER API

<Redirect>

```
<Redirect to=„/contact"/>

<Redirect
  to={{
    pathname: „/contact"
  }}
/>
```

REACT ROUTER API

<Switch>

```
<Switch>
  <Route path="/" />
  <Route path="/" />
  <Route path="/" />
</Switch>
```

REACT ROUTER API

{match, location, history} = props

REACT ROUTER API HOOKS

useHistory

REACT ROUTER API HOOKS

```
import { useHistory } from "react-router-dom";

function HomeButton() {
    const history = useHistory();

    function handleClick() {
        history.push("/home");
    }

    return (
        <button type="button" onClick={handleClick}>
            Go home
        </button>
    );
}
```

REACT ROUTER API HOOKS

useLocation

REACT ROUTER API HOOKS

```
const location = useLocation();
```

```
useEffect(() => {
    //send information to google analytics
    console.log('You are here:', location.pathname)
}, [location]);
```

REACT ROUTER API HOOKS

useParams

REACT ROUTER API HOOKS

```
function BlogPost() {
  const { postId } = useParams();
  return <div>Now showing post {postId}</div>;
}
```

DOCUMENTATION

reacttraining.com/react-router/

/catalog => <Catalog />
/contact => <Contact />

```
function App() {
  return (
    <div>
      <h2>Website 2019</h2>
      <Catalog />
      <Contact />
    </div>
  );
}
```

npm i **react-router-dom**

```
import {
    BrowserRouter as Router,
    Route
} from "react-router-dom";

function App() {
  return (
    <div>
      <h2>Website 2019</h2>
      <Catalog />
      <Contact />
    </div>
  );
}
```

```
import {
    BrowserRouter as Router,
    Route
} from "react-router-dom";

function App() {
  return (
    <Router>
      <div>
        <h2>Website 2019</h2>
        <Route path="/catalog" component={Catalog} />
        <Route path="/contact" component={Contact} />
      </div>
    </Router>
  );
}
```

DEMO

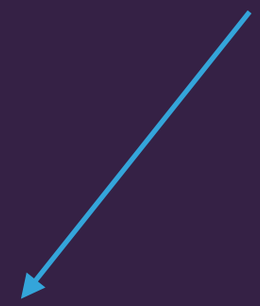
404

```
<Switch>
    <Route exact path=„/" component={Home} />
    <Route path="/catalog" component={Catalog} />
    <Route path="/contact" component={Contact} />
    <Route component={NotFound} />
</Switch>
```

NESTED ROUTERS

url: http://localhost:3000/catalog

match.url: /catalog



```
const Catalog = ({ match }) => (
  <div>
    <h1>I am Catalog Component {match.url}</h1>
    <Route path={`${match.url}/a`} component={CatalogA} />
    <Route path={`${match.url}/b`} component={CatalogB} />
  </div>
);

<Router>
  <Switch>
    <Route path="/catalog" component={Catalog} />
    <Route path="/contact" component={Contact} />
    <Route component={NotFound} />
  </Switch>
</Router>
```

path=/catalog/a

path=/catalog/b