

REACT.JS



PRZEMYSŁAW WISZOWATY

HELLO!



software HUT

TENDERHUT GROUP

meet.js

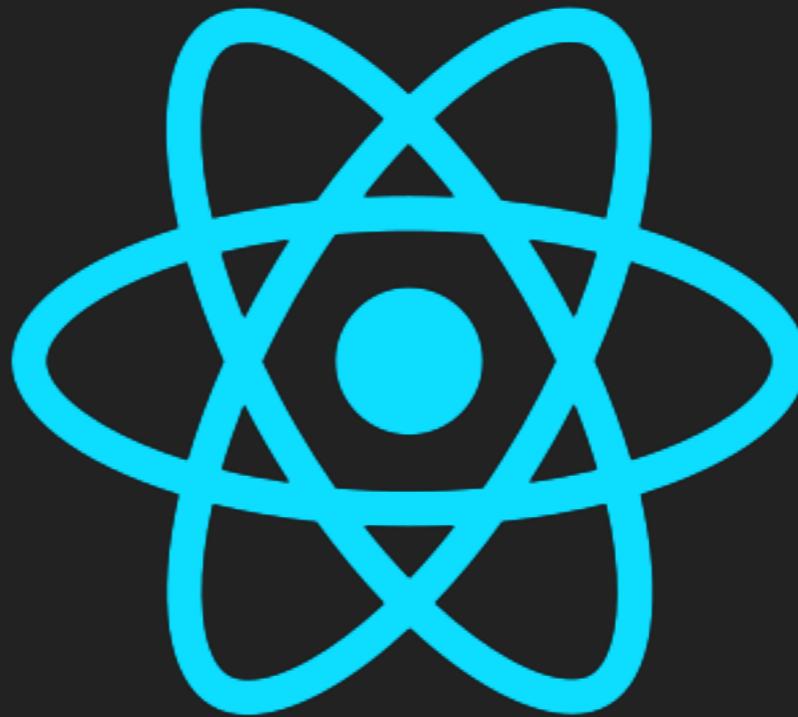
białystok



REACT IS EVERYWHERE

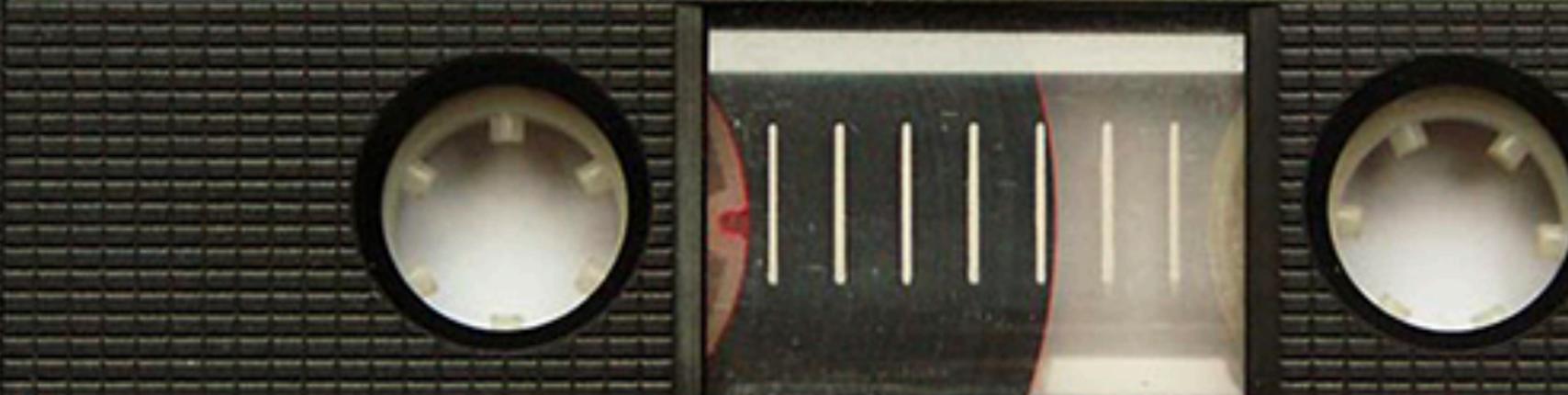
„Learn Once, Write Anywhere”

FRONTEND
BACKEND
MOBILE



A back to the 2000s

N.R. (OG)
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://

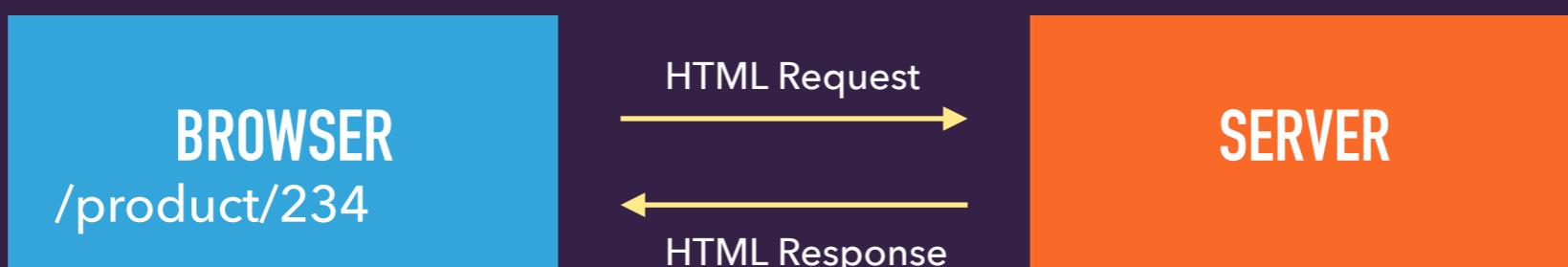
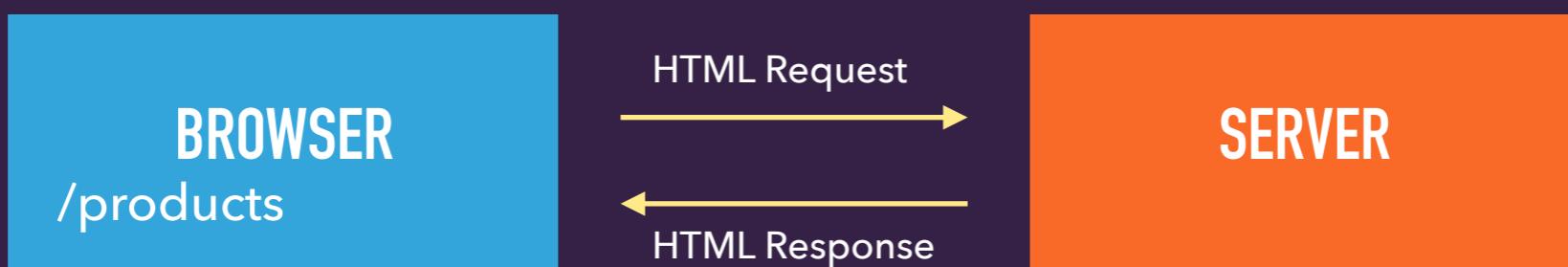
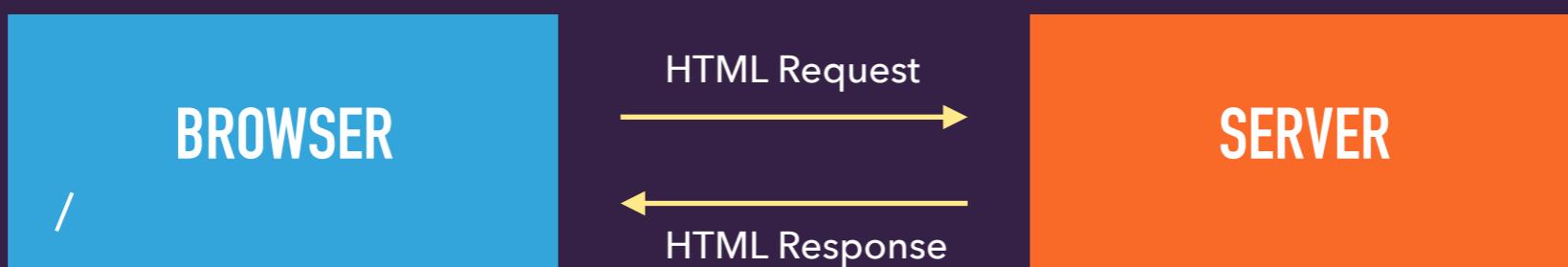
Some pages may work very slowly or not at all. Many webapps are just too advanced for Geocities.

Turn your sound up for the full effect.

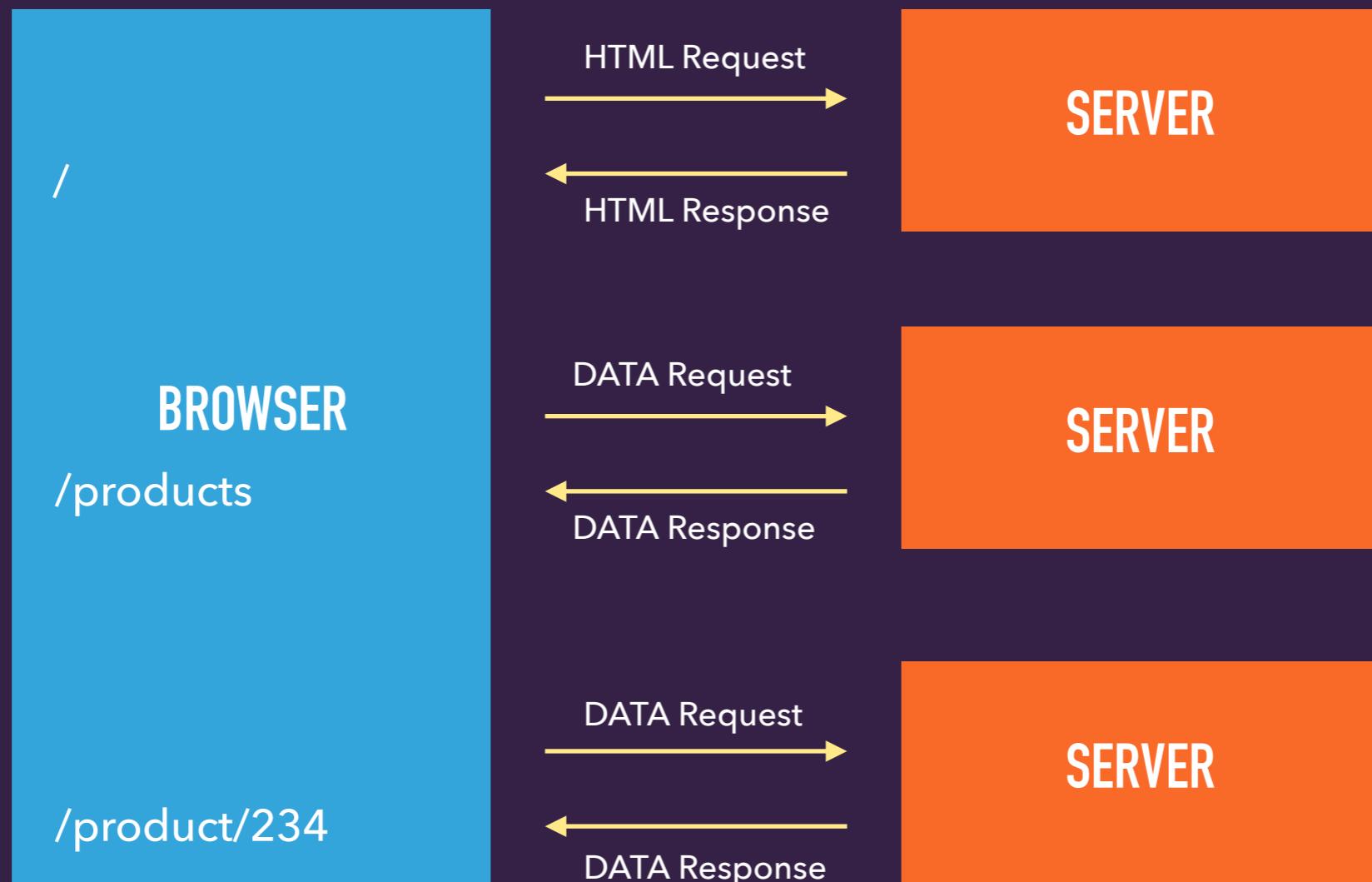
Created by [Mike Lacher](#)



TRADITIONAL WEBSITE



SINGLE PAGE APPLICATION





414 x 736



Elements

Console

Sources

Network

Performance

Memory

Application

Security

Audits

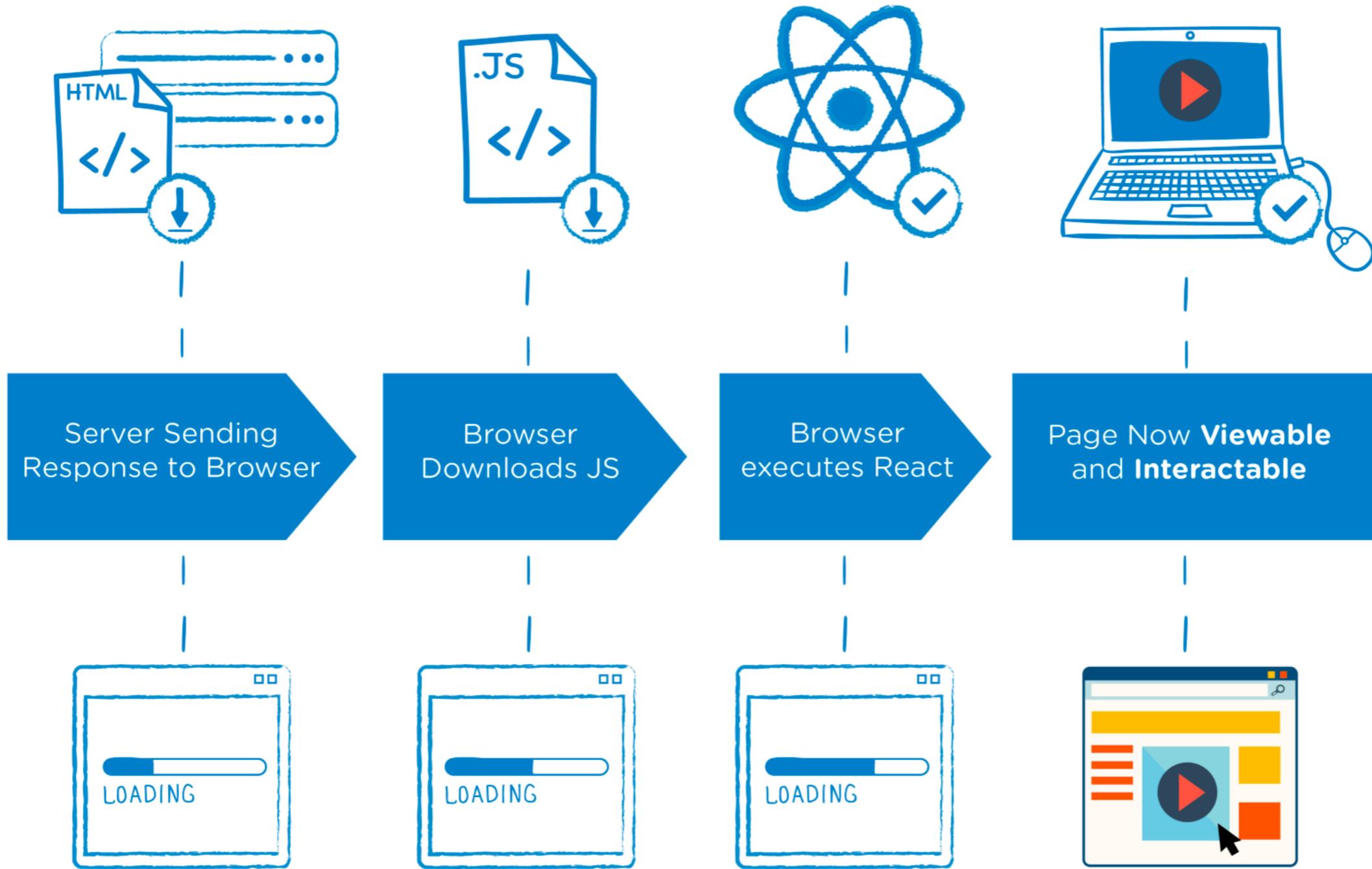


(index) :formatted x

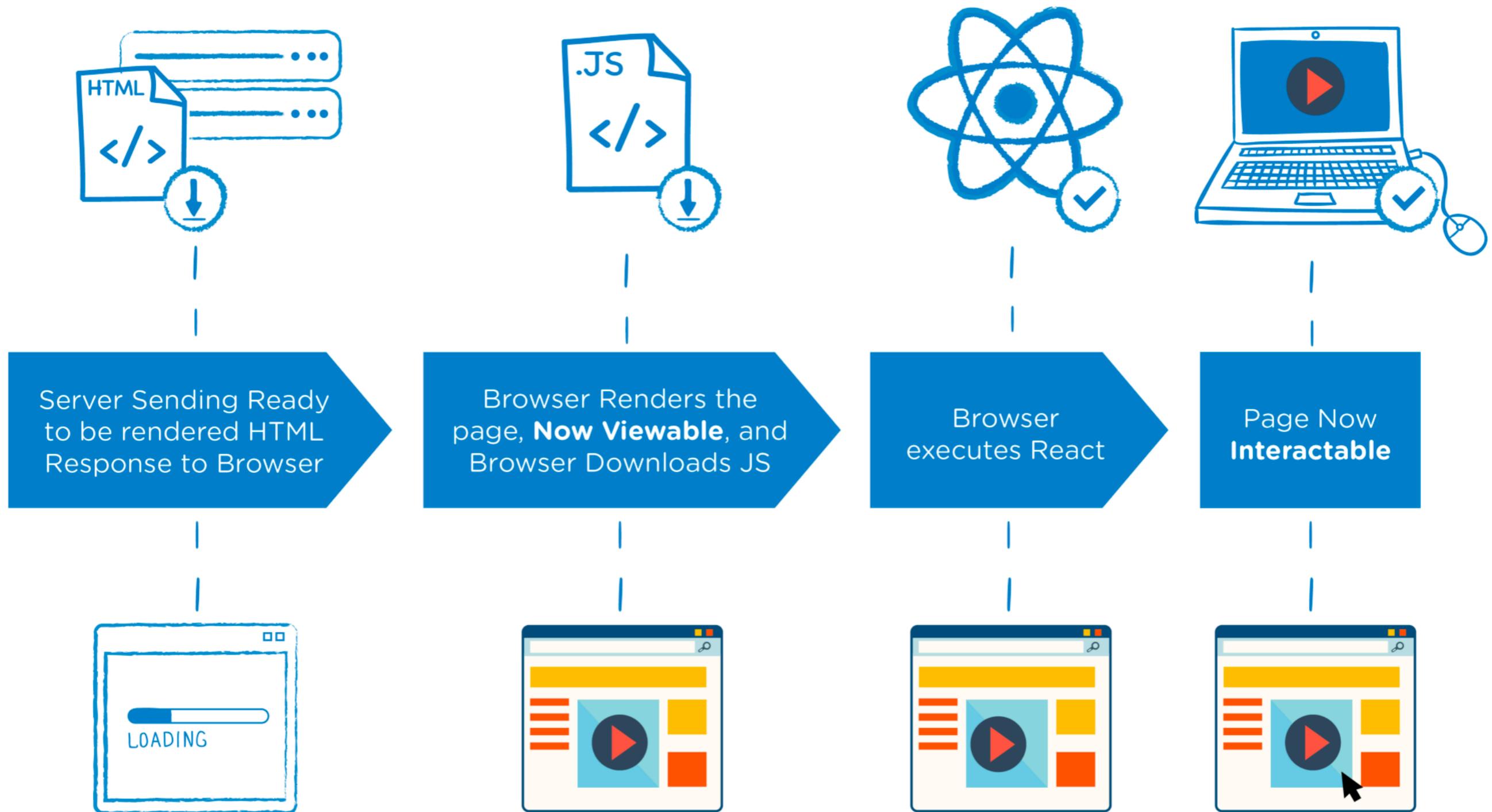
```
1 <!DOCTYPE html>
2 <html lang="en" prefix="og: http://ogp.me/ns#>
3   <head>
4     <meta charset="utf-8">
5     <meta name="viewport" content="width=device-width,initial-scale=1,shrink-to-fit=no">
6     <link rel="apple-touch-icon" sizes="180x180" href="/icons/apple-touch-icon.png?v=694AzX47gQ">
7     <link rel="icon" type="image/png" sizes="32x32" href="/icons/favicon-32x32.png?v=694AzX47gQ">
8     <link rel="icon" type="image/png" sizes="16x16" href="/icons/favicon-16x16.png?v=694AzX47gQ">
9     <link rel="mask-icon" href="/icons/safari-pinned-tab.svg?v=694AzX47gQ" color="#5bbad5">
10    <link rel="shortcut icon" href="/icons/favicon.ico?v=694AzX47gQ">
11    <meta name="msapplication-TileColor" content="#2d89ef">
12    <meta name="msapplication-config" content="/icons/browserconfig.xml?v=694AzX47gQ">
13    <meta name="theme-color" content="#ffffff">
14    <meta name="google-site-verification" content="NCJvpBpTbp9pUJdDk7t6dhIis0j4kZJEphgntRU-TbM"/>
15    <link rel="manifest" href="/manifest.json">
16    <title>TenderHut</title>
17    <link href="/static/css/main.5ca042e5.css" rel="stylesheet">
18  </head>
19  <body>
20    <noscript>You need to enable JavaScript to run this app.</noscript>
21    <div id="app-root"></div>
22    <div id="modal-root"></div>
23    <script type="text/javascript" src="/static/js/main.dc65024f.js"></script>
24  </body>
25</html>
```



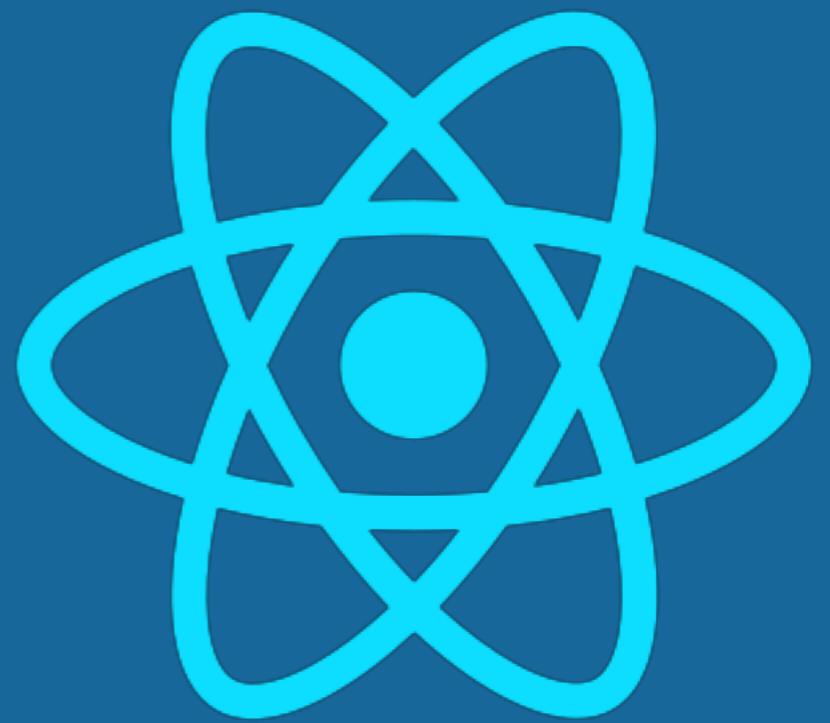
CSR



SSR



WHAT IS REACT?

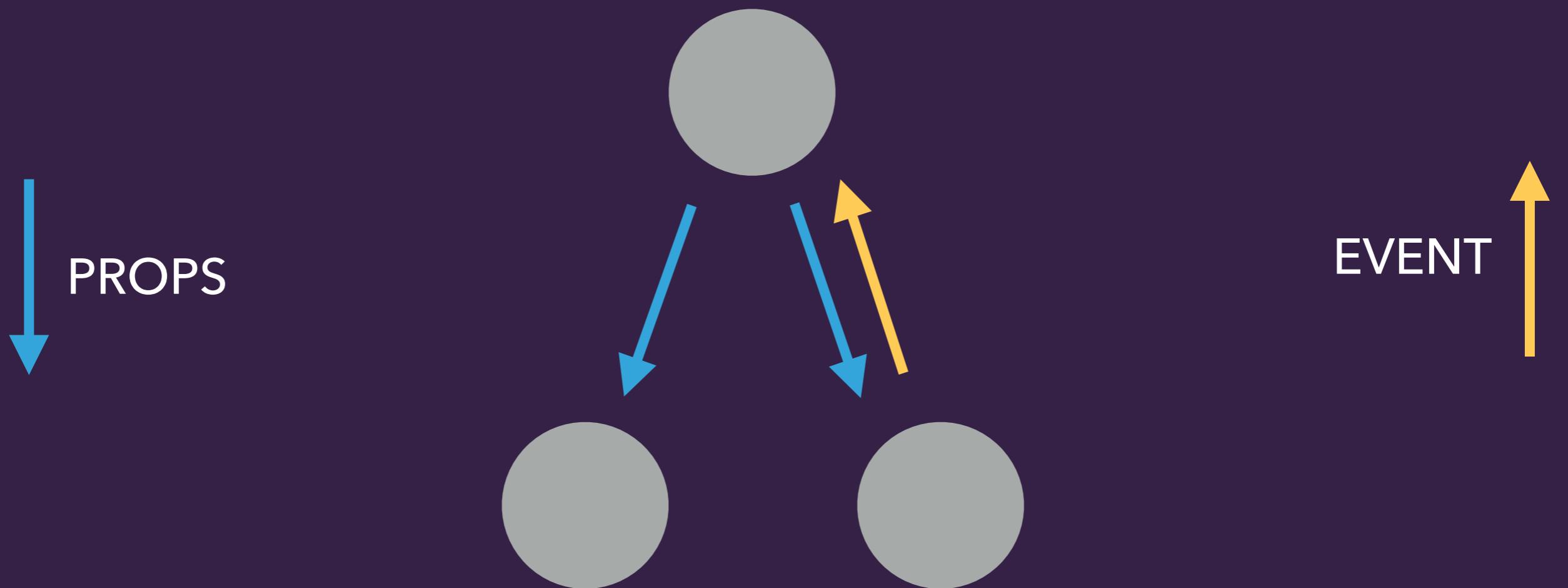


**LIBRARY
NOT
~~FRAMEWORK~~**

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

FUNCTIONAL

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

FUNCTIONAL

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

FUNCTIONAL

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

CLASS

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

**EVERYTHING
IS A COMPONENT**

CardComponent

CardHeaderComponent

COMPONENT...

COMPONENT EVERYWHERE

makeameme.org

CardBodyComponent

UserPhotoComponent

```
1 import React from 'react';
2
3 const FacebookComponent = () => (
4   <CardComponent>
5     <CardHeaderComponent>
6       <SelectorComponent />
7       <SelectorComponent />
8       <SelectorComponent />
9     </CardHeaderComponent>
10
11    <CardBodyComponent>
12      <UserPhotoComponent />
13    </CardBodyComponent>
14
15    <CardFooterComponent />
16  </CardComponent>
17);
18
19 export default FacebookComponent;
20
```

JSX

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

JSX

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

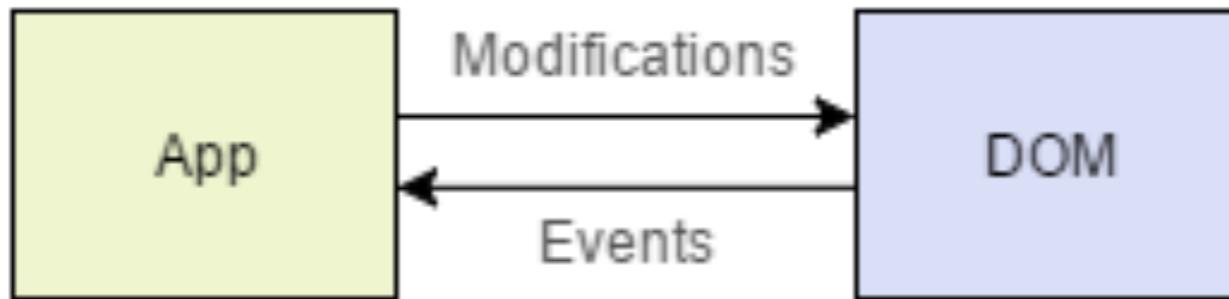
JS

**REACT DOESN'T
REQUIRE USING JSX**

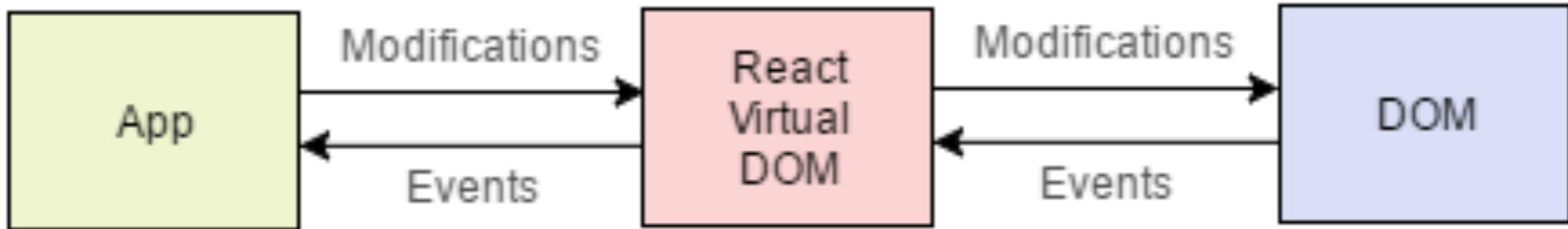
**VIRTUAL
DOM**

IT'S FAST
IT'S PURE
IT WORKS

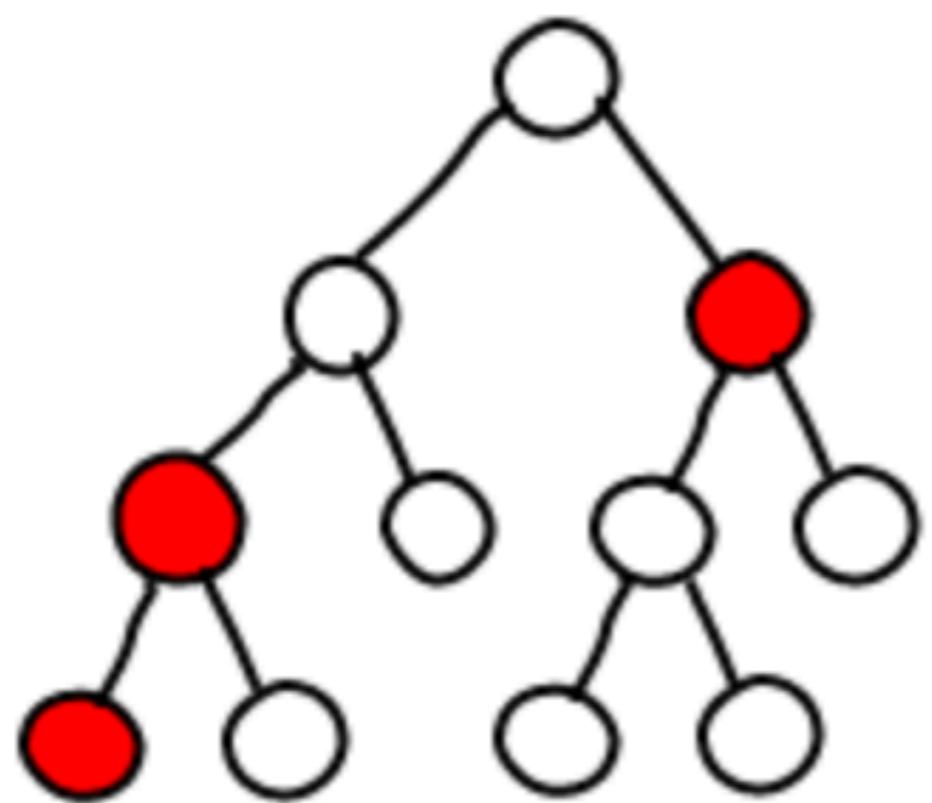
Traditional Web Application



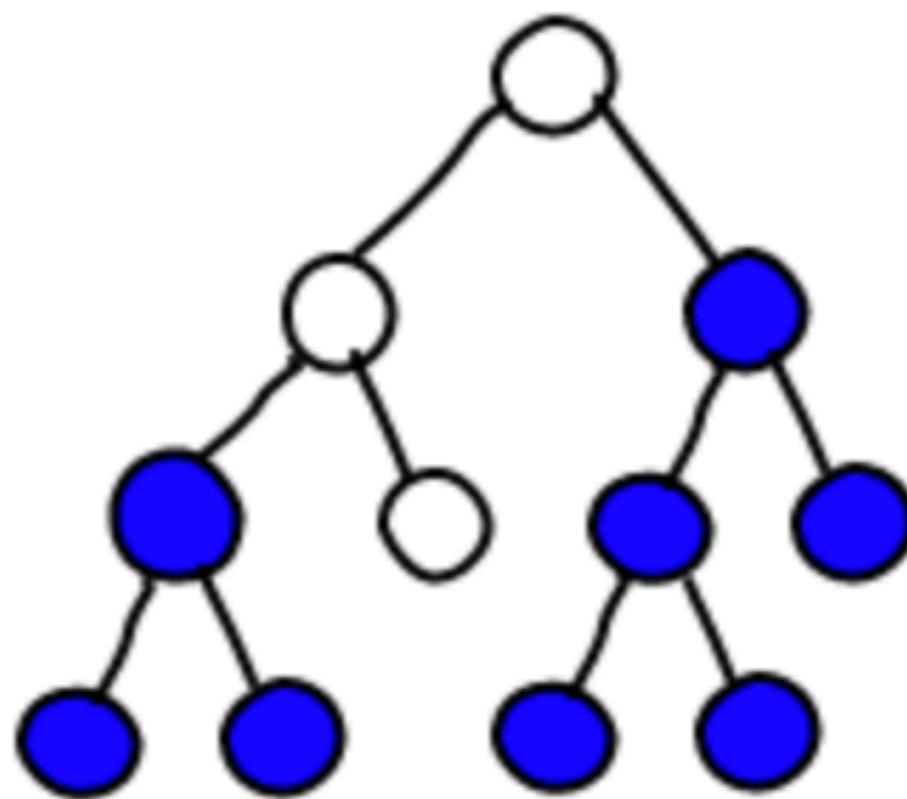
React.js



Dirty



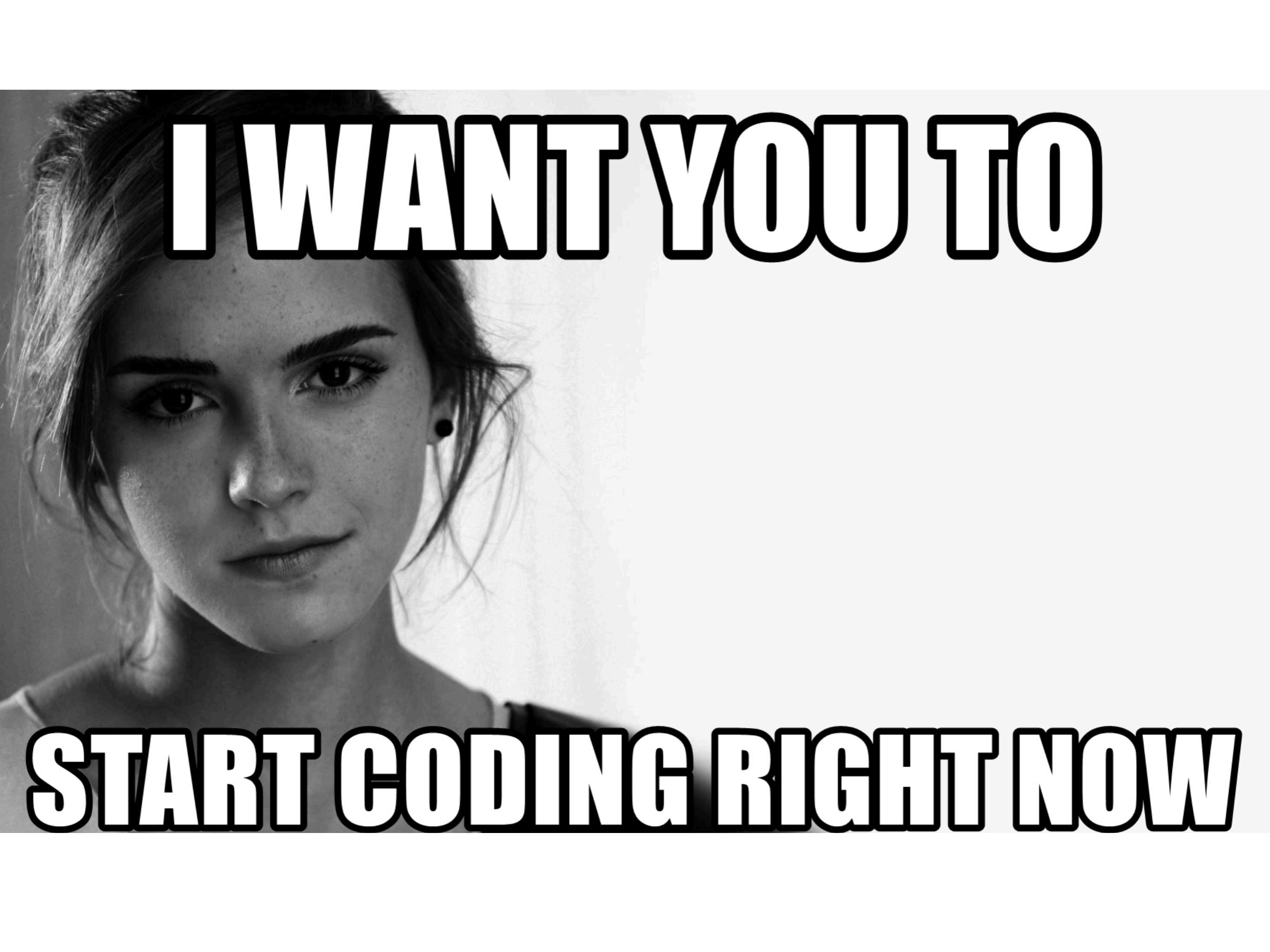
Re-rendered



Hello, world!

It is 12:26:46 PM.

```
Console  Sources  Network  Timeline
▼<div id="root">
  ▼<div data-reactroot>
    <h1>Hello, world!</h1>
    ▼<h2>
      <!-- react-text: 4 -->
      "It is "
      <!-- /react-text -->
      <!-- react-text: 5 -->
      "12:26:46 PM"
      <!-- /react-text -->
      <!-- react-text: 6 -->
      "."
      <!-- /react-text -->
    </h2>
  </div>
</div>
```



I WANT YOU TO

START CODING RIGHT NOW

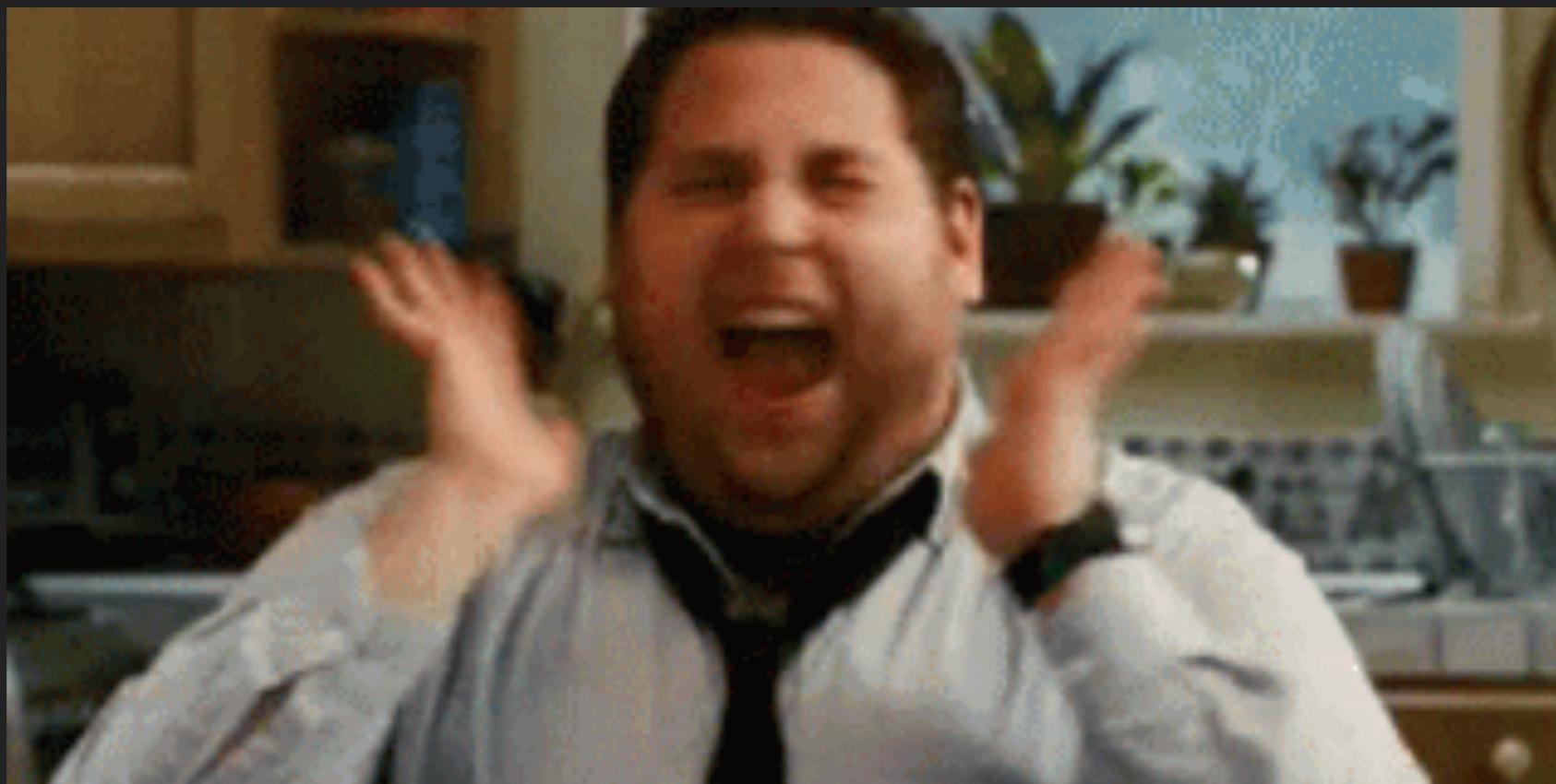
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>React App</title>
    <script src="https://unpkg.com/react@16/umd/react.production.min.js" crossorigin></script>
    <script src="https://unpkg.com/react-dom@16/umd/react-dom.production.min.js" crossorigin></script>
  </head>
  <body>
    <div id="root"></div>

    <script type="text/javascript">
      var Welcome = function Welcome() {
        return React.createElement("h1", null, "Hello, World");
      };

      ReactDOM.render(Welcome, document.getElementById("root"));
    </script>
  </body>
</html>
```

**CREATE
REACTAPP**

npx create-react-app myapp

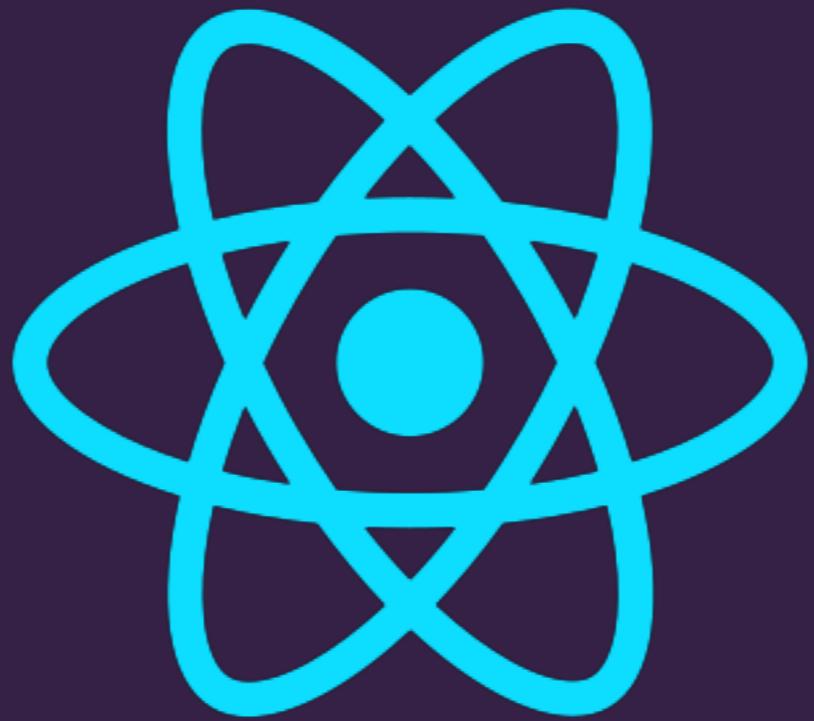


DEMO

REACT DEV TOOLS

DEMO

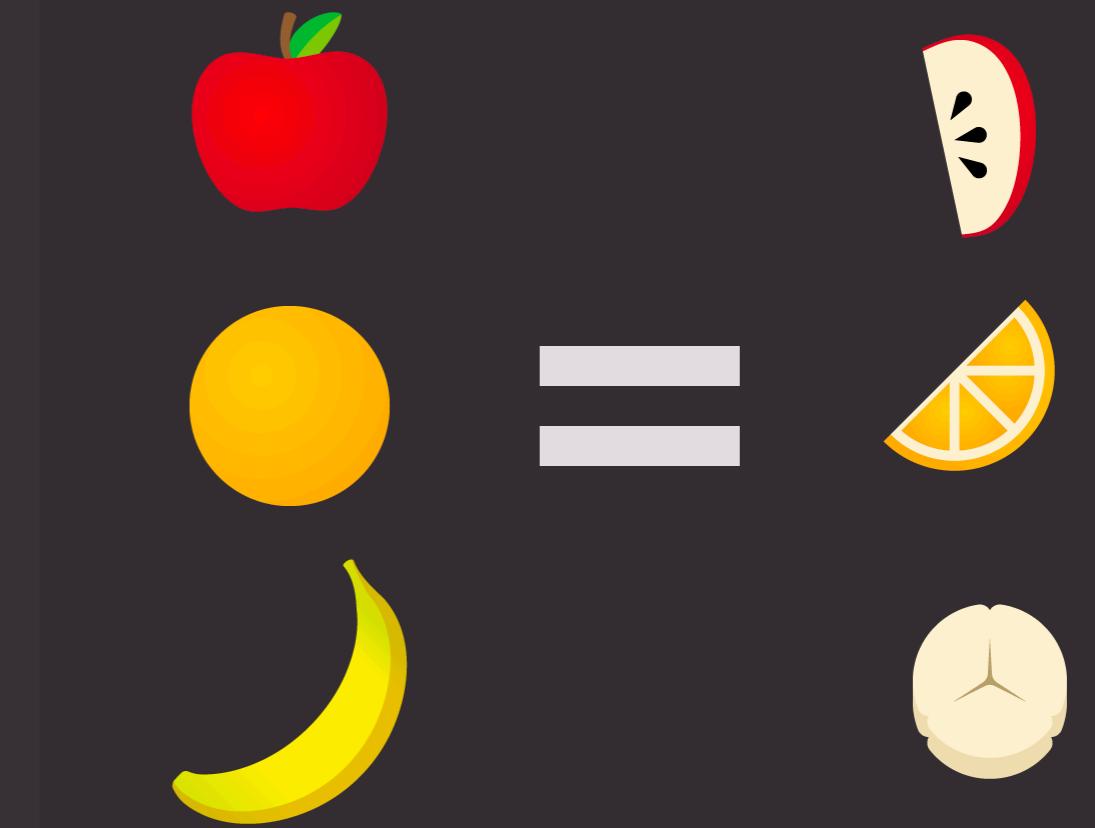
THANKS



REACT.JS

FUNCTIONAL PROGRAMMING

MAP



MAP

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

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

```
> newArray[2,,,]
```

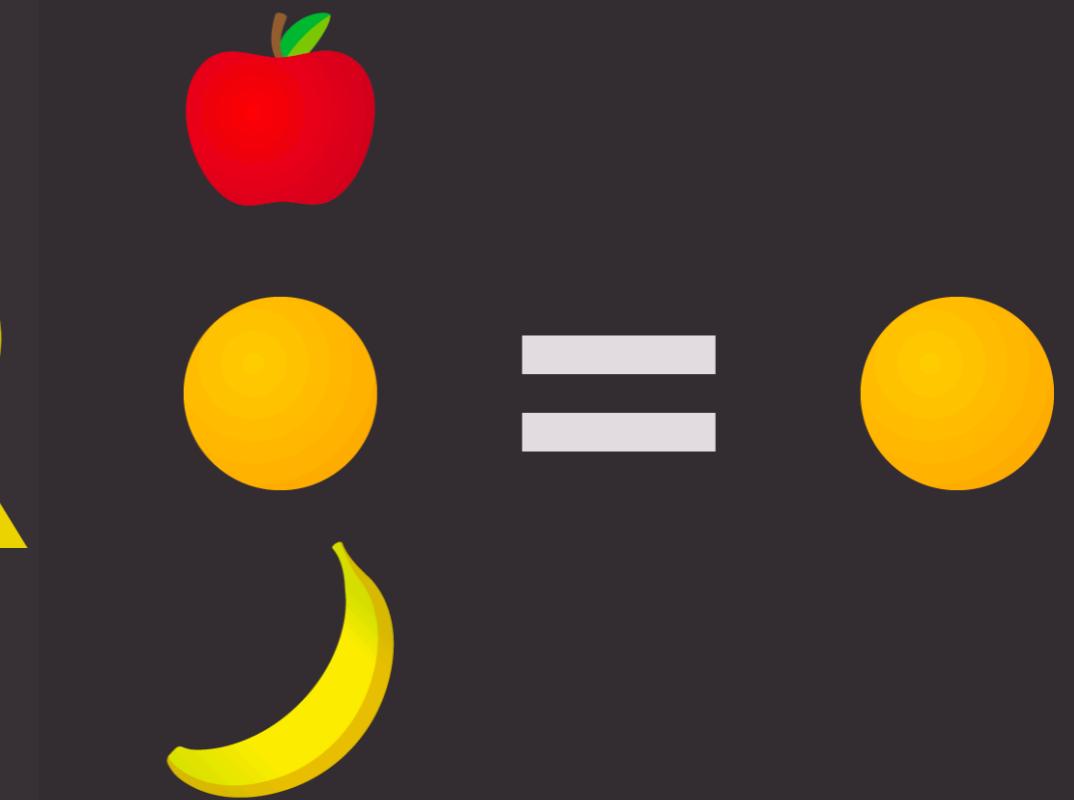
```
> newArray[2,3,,,]
```

```
> 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.map(element => element > 3);
```

```
> newArray[]
```

```
> newArray[]
```

```
> newArray[]
```

```
> newArray[4]
```

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

RENDER MULTIPLE COMPONENT

RENDER MULTIPLE COMPONENT

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

```
const App = () => (
  <div>
    {[{"name": "Asia"}, {"name": "Paweł"}, {"name": "Gosia"}, {"name": "Darek"}].map(item =>
      <h4>{item}</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**

PROPS

**PROPS =
PRIMITIVE VALUES,
REACT ELEMENTS
FUNCTIONS**

PROPS DEFAULT IS TRUE

```
<MyComponent dark />
```

```
<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."  
/>
```

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 <Post {...post} />;  
  } else {  
    return null;  
  }  
};
```

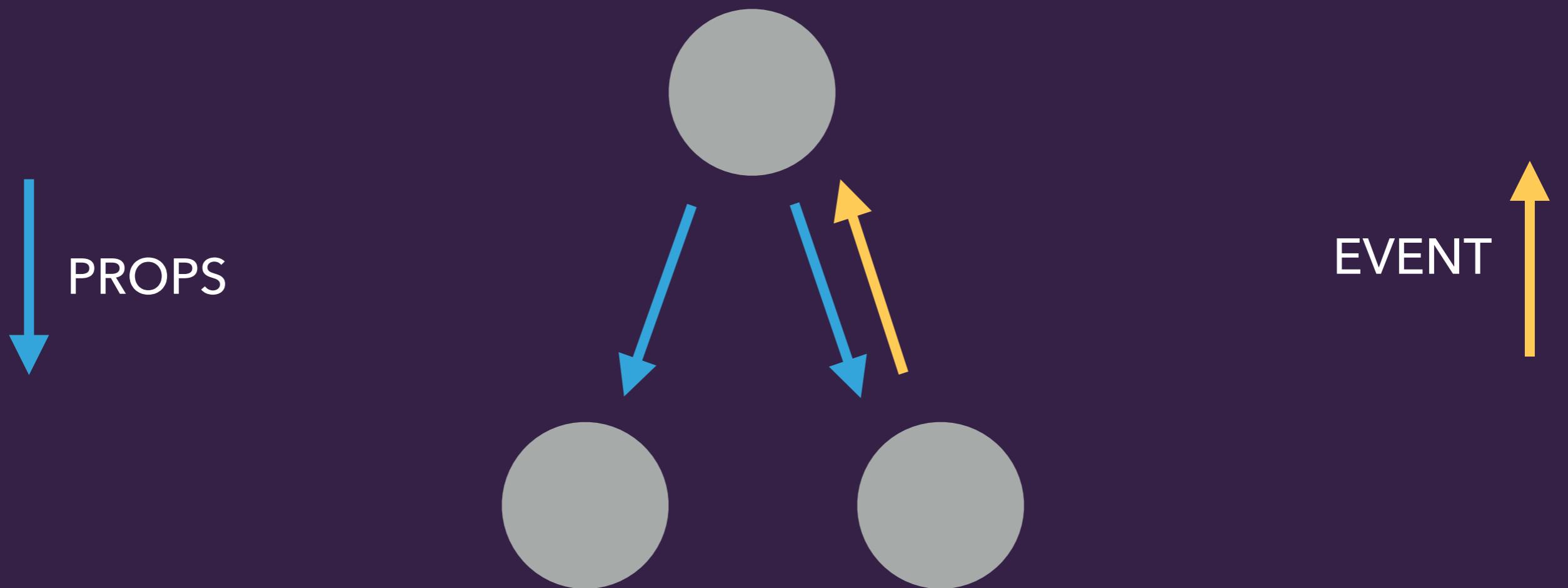
```
const App = () => {  
  return post ? <Post {...post} /> : null;  
};
```

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 = () => {  
  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

- ✖ 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

```
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 />
```

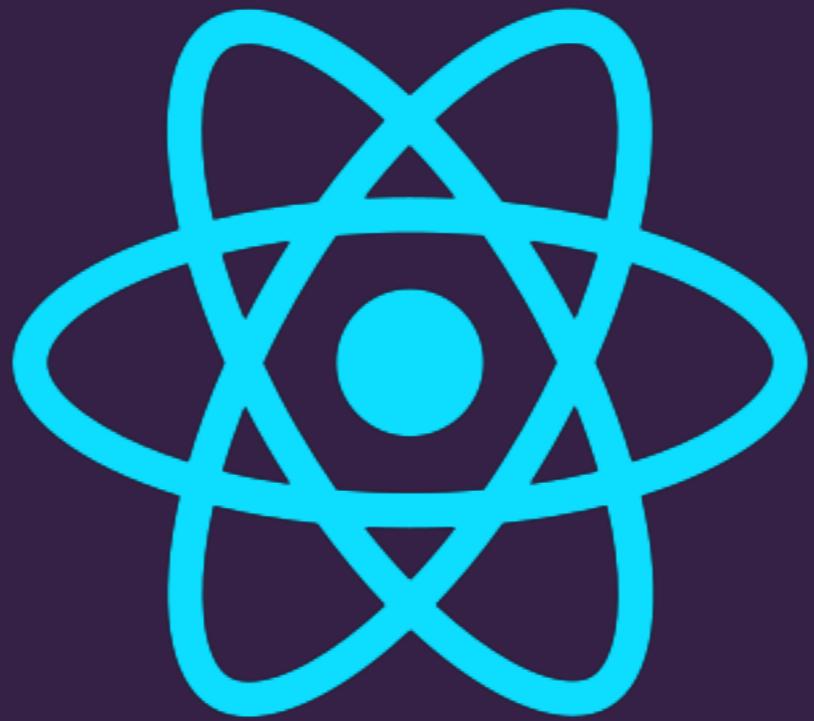
<PROVIDER>

<CONSUMER>

```
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>
);
```

THANKS



REACT.JS

HOC
**HIGH ORDER
COMPONENT**

HOC

```
component => componentOnSteroids;
```

HOC

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

HOC

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

```
const HelloWithLang = withLanguage(Hello);
```

```
const App = () => (
  <>
    <Hello name="Kasia" />
    <HelloWithLang name="Kasia" />
  </>
);
```

HOC

```
▼ <App> == $r
  ► <Hello name="Kasia">...</Hello>
  ▼ <Unknown name="Kasia">
    ► <Hello name="Kasia" language="pl">...</Hello>
    </Unknown>
  </App>
```

HOC



HOC

```
function logProps(InputComponent) {  
  InputComponent.prototype.componentWillReceiveProps = function(nextProps) {  
    console.log('Current props: ', this.props);  
    console.log('Next props: ', nextProps);  
  };  
  return InputComponent;  
}  
  
const EnhancedComponent = logProps(InputComponent);
```

HOC

```
function logProps(WrappedComponent) {  
  return class extends React.Component {  
    componentWillMount(nextProps) {  
      console.log('Current props: ', this.props);  
      console.log('Next props: ', nextProps);  
    }  
    render() {  
      return <WrappedComponent {...this.props} />;  
    }  
  }  
}
```

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

```
@withRouter
@connect(param)
export default class MyFancyComponent extends React.Component {
}
```

HOC

react-fns

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

CLASS COMPONENT

CLASS COMPONENT

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

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

CLASS COMPONENT

„When should I use a function and when a class?”

~undefined

**CLASS
COMPONENT**

DIFFERENCES

CLASS COMPONENT

SYNTAX

CLASS COMPONENT DIFFERENCES

```
var Welcome = function Welcome(props) {  
  return React.createElement(  
    "h1",  
    null,  
    "Hello, ",  
    props.name);  
};
```

CLASS COMPONENT

```
var Welcome = (function(_React$Component) {
  _inherits(Welcome, _React$Component);

  function Welcome() {
    _classCallCheck(this, Welcome);

    return _possibleConstructorReturn(
      this,
      (Welcome.__proto__ || Object.getPrototypeOf(Welcome)).apply(
        this,
        arguments
      )
    );
  }

  _createClass(Welcome, [
    {
      key: "render",
      value: function render() {
        return React.createElement("h1", null, "Hello, ", this.props.name);
      }
    }
  ]);

  return Welcome;
}
```

**CLASS
COMPONENT**

STATE

**CLASS
COMPONENT
STATE**

STATE VS PROPS

CLASS COMPONENT STATE

COMMON

plain JS objects

render update

deterministic

**CLASS
COMPONENT
STATE**

PROPS

Component's configuration

Received from above

Immutable

CLASS COMPONENT STATE

STATE

Component's information

Created in component

Changeable

CLASS COMPONENT STATE

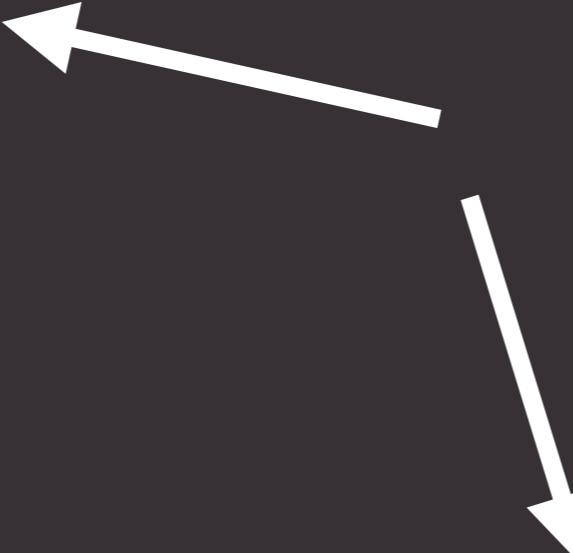
```
class Button extends React.Component {  
  constructor() {  
    super();  
    this.state = {  
      count: 0,  
    };  
  }  
}
```



```
  render() {  
    .....  
  }  
}
```

CLASS COMPONENT STATE

```
class Button extends React.Component {  
  constructor() {  
    super();  
    this.state = {  
      count: 0  
    };  
  }  
  
  render() {  
    return (  
      <button>  
        Clicked {this.state.count} times  
      </button>  
    );  
  }  
}
```

A diagram illustrating the flow of state from the constructor to the render method. A white arrow points from the 'count' variable declaration in the constructor up to the same 'count' variable declaration in the state object. Another white arrow points from the 'count' variable in the state object down to its usage in the 'Clicked' text within the render method's return statement.

```
class Button extends React.Component {  
  constructor() {  
    super();  
    this.state = {  
      count: 0  
    };  
    this.updateCount = this.updateCount.bind(this)  
  }  
}
```

```
updateCount() {  
  this.setState((prevState, props) => {  
    return { count: prevState.count + 1 };  
  });  
}
```

```
render() {  
  return (  
    <button onClick={this.updateCount}>  
      Clicked {this.state.count} times  
    </button>  
  );  
}  
}
```

DEMO

INITIALIZE STATE

```
this.state = {}
```

CHANGE STATE

```
this.setState()
```

setState()

```
this.setState({ count: 2 });
```

```
this.setState((state) => {  
  return {count: state.count + 1};  
});
```

setState()



```
constructor() {  
  super();  
  this.state = {  
    count: 0;  
  }  
}
```

```
incrementCount() {  
  this.setState({count: this.state.count + 1});  
}
```

```
handleSomething() {  
  this.incrementCount();  
  this.incrementCount();  
  this.incrementCount();  
}
```

```
constructor() {  
  super();  
  this.state = {  
    count: 0;  
  }  
}
```

```
incrementCount() {  
  this.setState((state) => {  
    return {count: state.count + 1}  
  });  
}
```

```
handleSomething() {  
  this.incrementCount();  
  this.incrementCount();  
  this.incrementCount();  
}
```

setState()



STATE ANTI-PATTERN



```
class MyComponent extends Component {
  constructor(props) {
    super(props);
    this.state = {
      someValue: props.someValue,
    };
  }
}
```

„When should I use a function and when a class?”

~undefined

REACT HOOKS

COMING SOON!

```
import { useState } from 'react';

const Example = () => {
  // Declare a new state variable, which we'll call "count"
  const [count, setCount] = useState(0);

  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>
        Click me
      </button>
    </div>
  );
}
```

COMING SOON!

CLASS COMPONENT DIFFERENCES

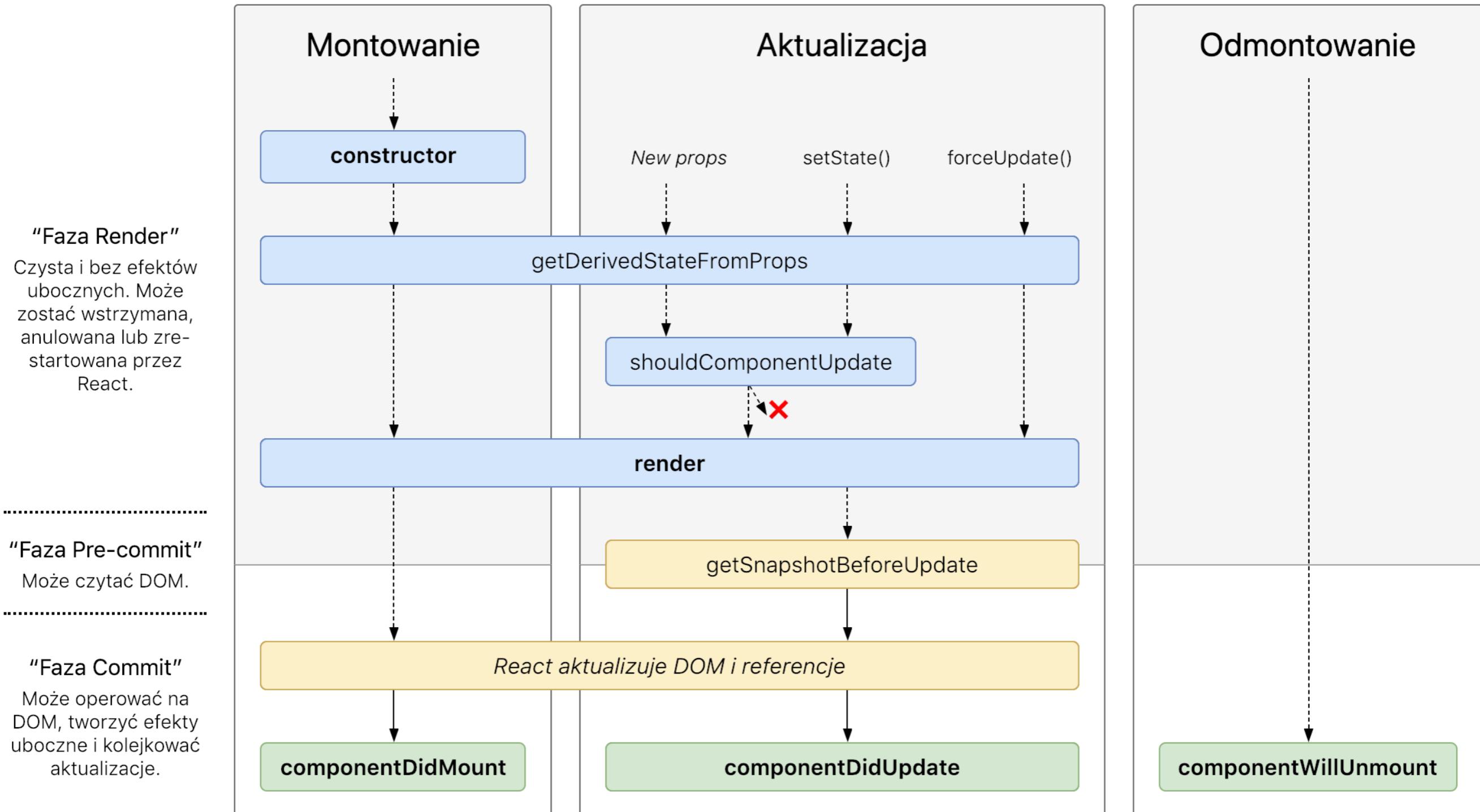
LIFECYCLE HOOKS

LIFECYCLE HOOKS

MOUNTING

UPDATING

UNMOUNTING



constructor()

initialize state (`this.state`)

bind methods

static getDerivedStateFromProps()

return an object to update
the state

fired on every render

render()

required

componentDidMount()

load data from a remote endpoint

side-effect

DOM

shouldComponentUpdate()

performance optimization

getSnapshotBeforeUpdate()

componentDidUpdate()

invoked immediately after
updating

good to do network requests

componentWillUnmount()

invoked immediately before a
component is unmounted and destroyed
cleaning up

BIND

```
class Foo extends Component {  
  handleClick() {  
    console.log("Click happened", this.state);  
  }  
  render() {  
    return <button onClick={this.handleClick}>Click Me</button>;  
  }  
}
```

BIND



```
class Foo extends Component {  
  constructor(props) {  
    super(props);  
    this.handleClick = this.handleClick.bind(this);  
  }  
  handleClick() {  
    console.log('Click happened', this.state);  
  }  
  render() {  
    return <button onClick={this.handleClick}>Click Me</button>;  
  }  
}
```

BIND



```
class Foo extends Component {  
  handleClick = () => {  
    console.log("Click happened", this.state);  
  }  
  render() {  
    return <button onClick={this.handleClick}>Click Me</button>;  
  }  
}
```

BIND



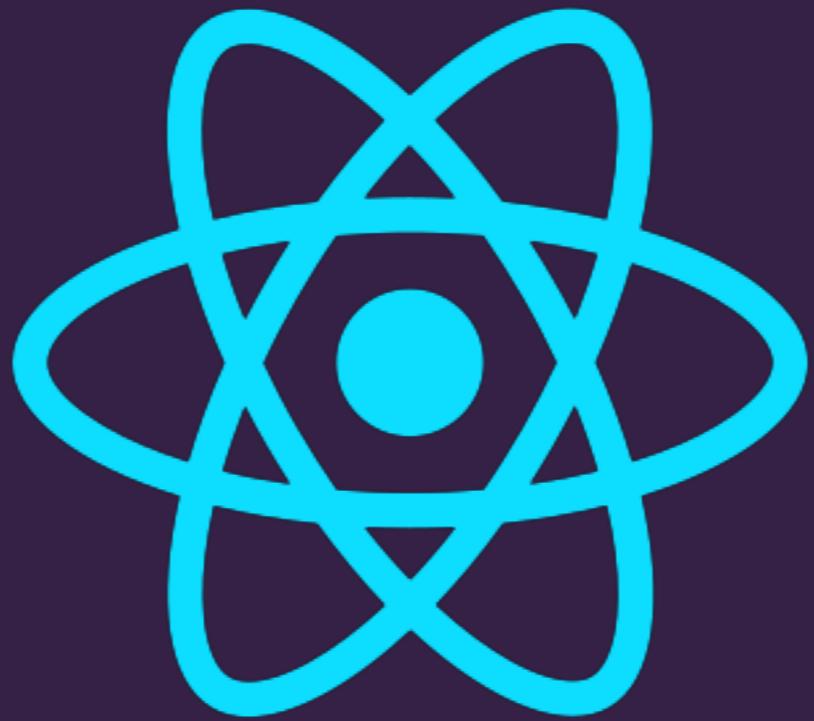
```
class Foo extends Component {  
  handleClick() {  
    console.log("Click happened", this.state);  
  };  
  render() {  
    return (  
      <button onClick={this.handleClick.bind(this)}>  
        Click Me  
      </button>  
    );  
  }  
}
```

BIND



```
class Foo extends Component {  
  handleClick() {  
    console.log("Click happened", this.state);  
  }  
  render() {  
    return (  
      <button onClick={() => this.handleClick()}>  
        Click Me  
      </button>  
    );  
  }  
}
```

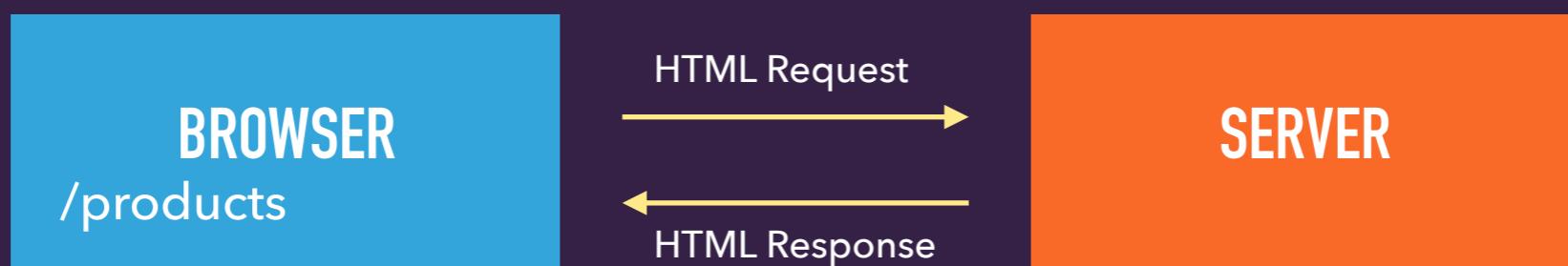
THANKS



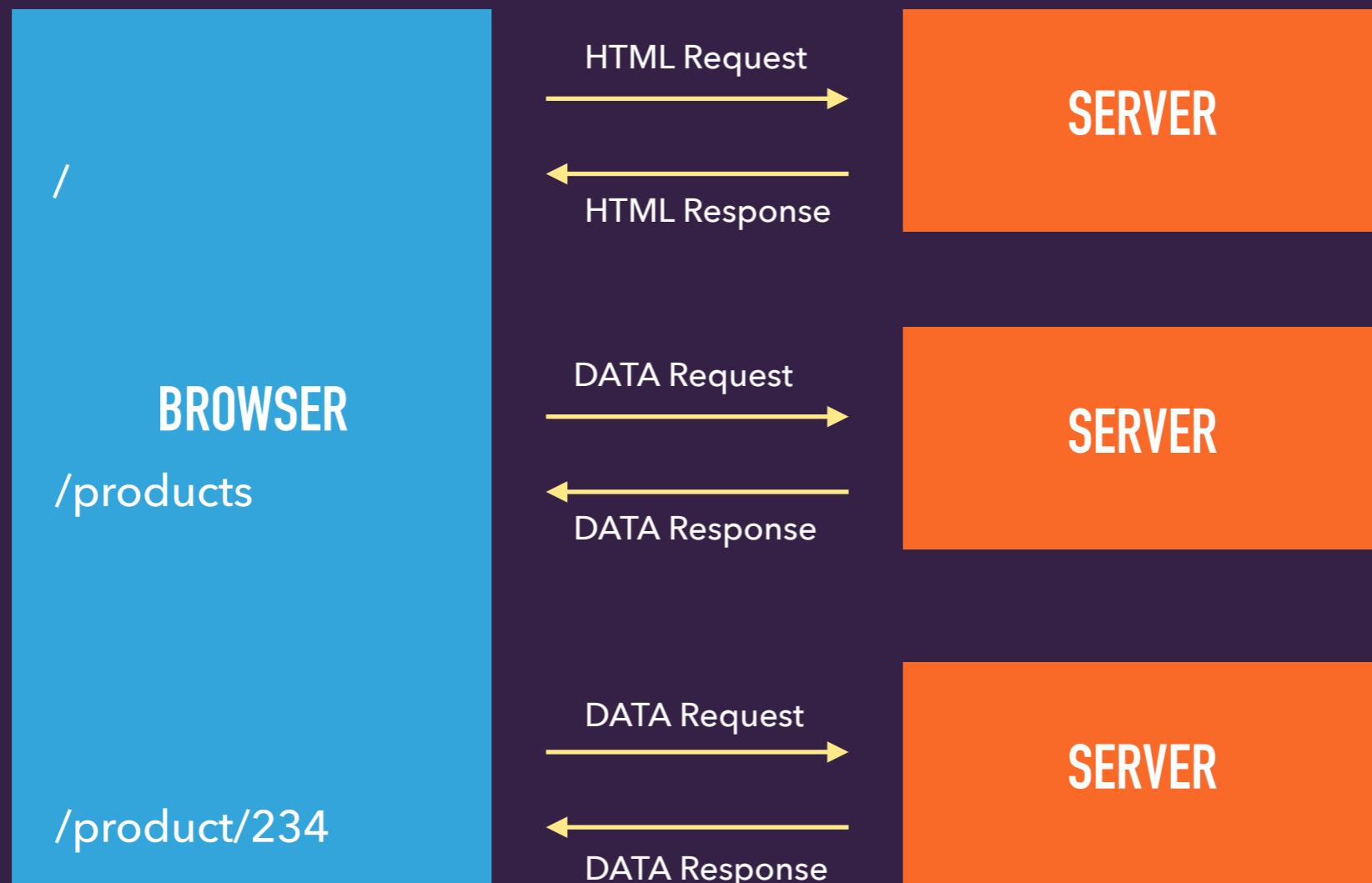
REACT.JS

ROUTING

TRADITIONAL WEBSITE



SINGLE PAGE APPLICATION

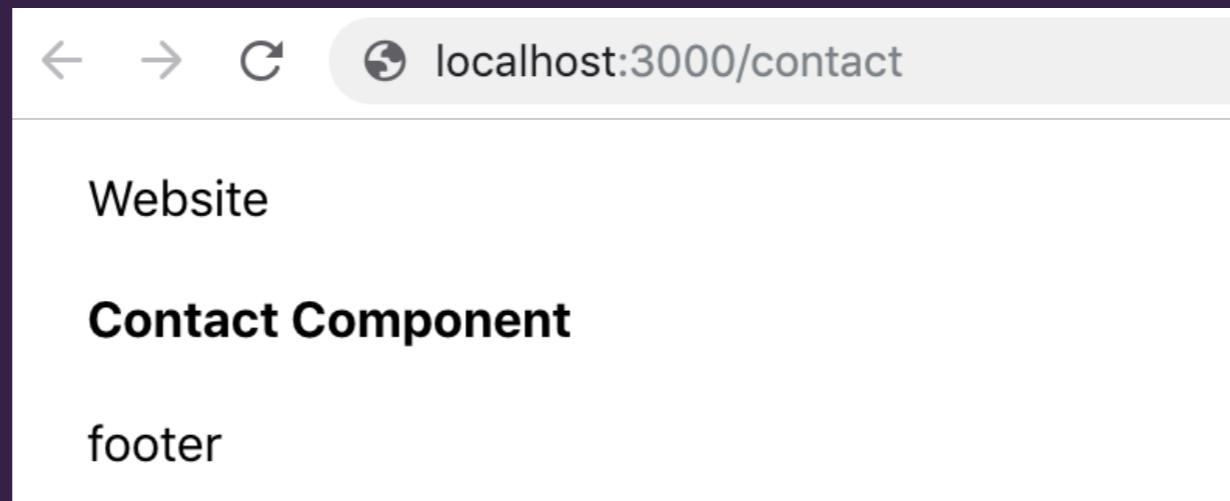
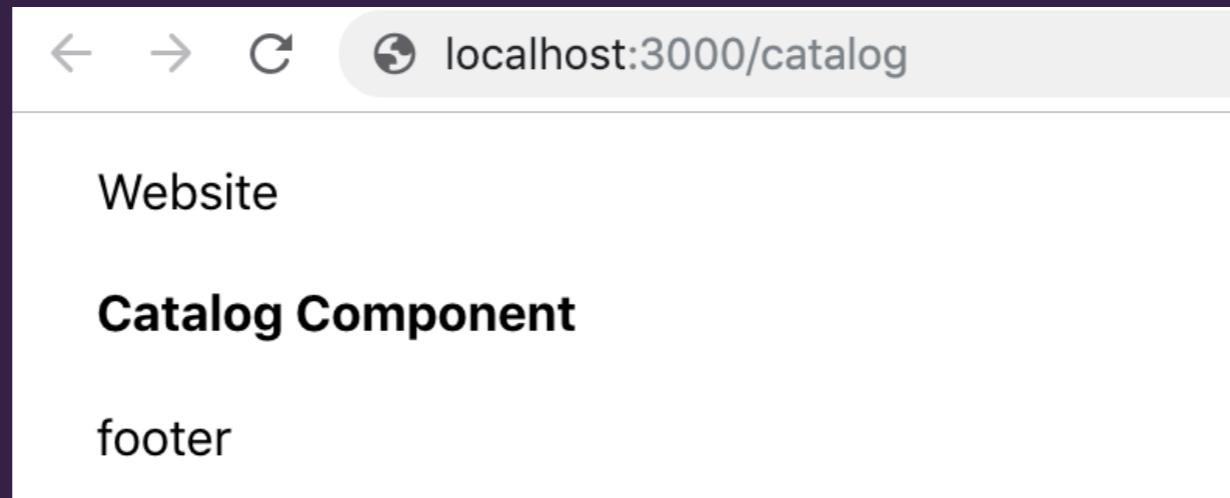
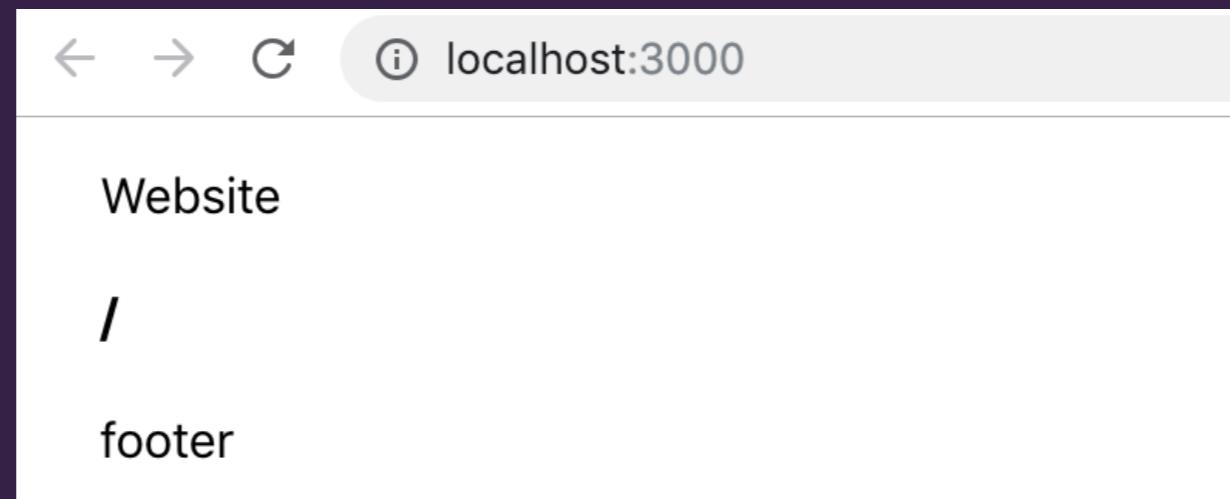


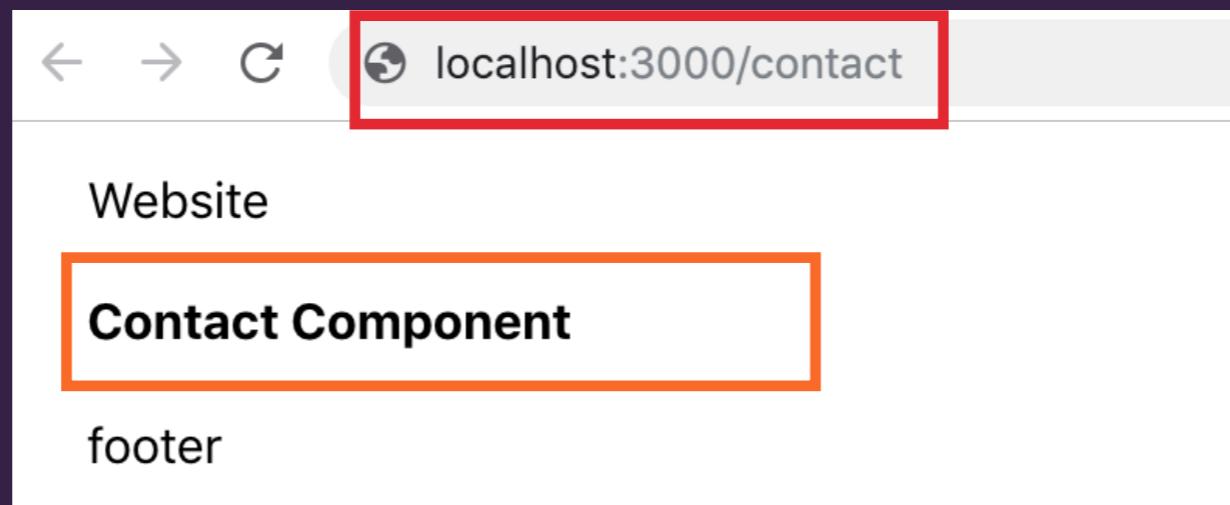
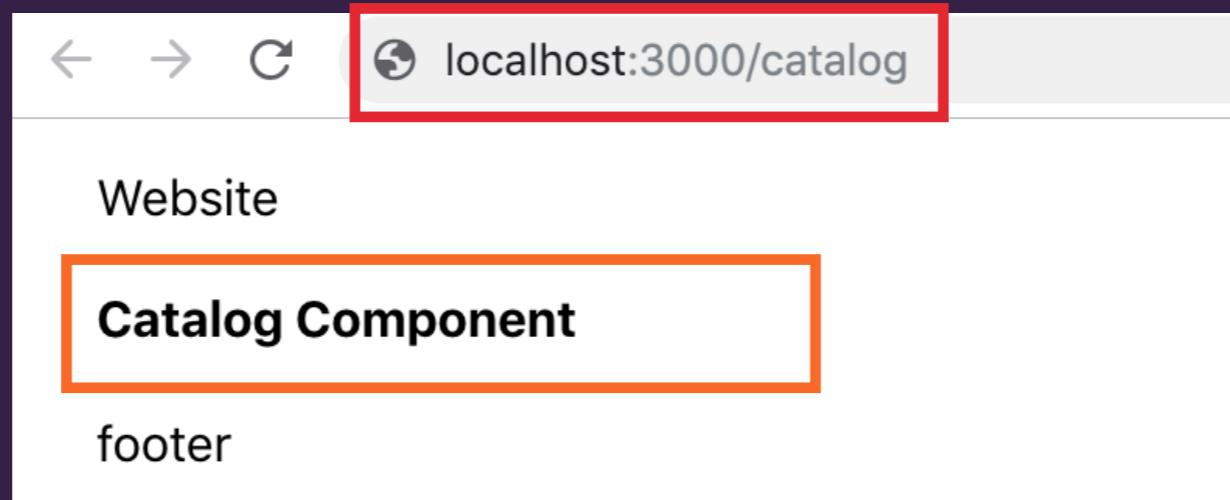
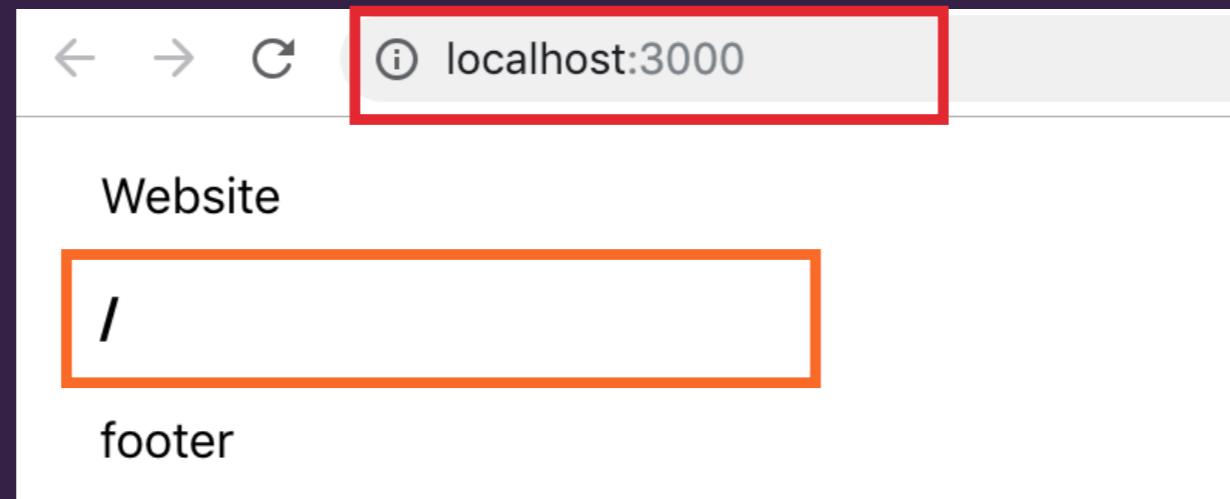
STATIC ROUTING

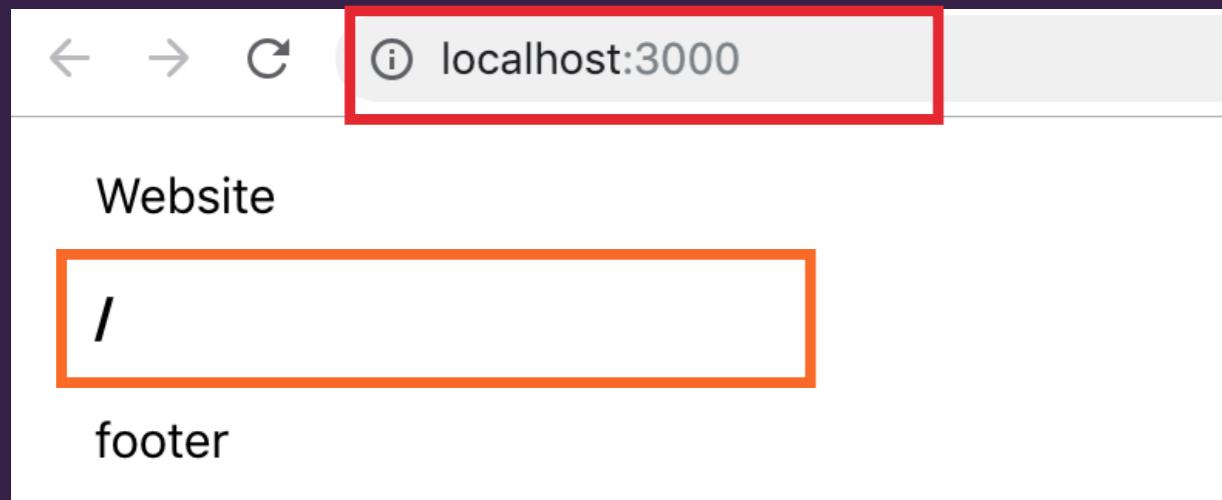
DYNAMIC ROUTING

„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.”

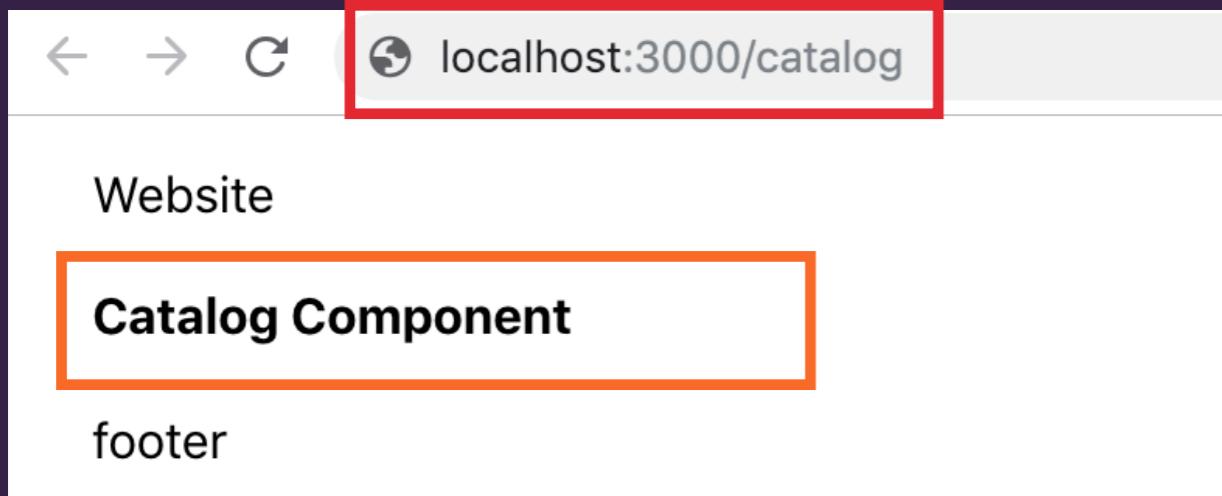
– React Training



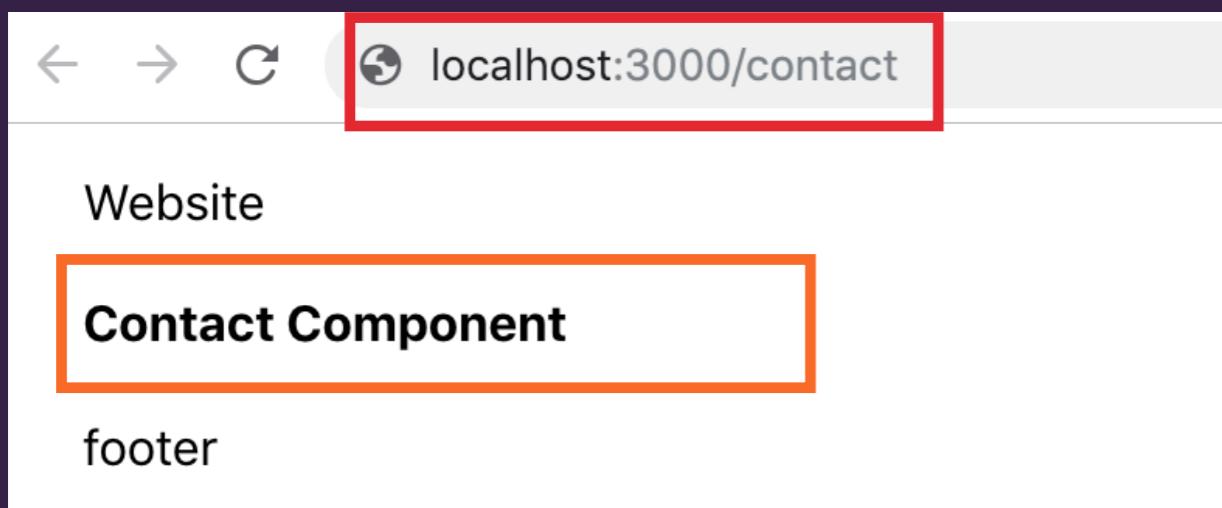




path: /
component: Home



path: /catalog
component: Catalog



path: /contact
component: Contact

REACT ROUTER V4

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-CONFIG

Static route config helpers

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

<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 />} />
```

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 }  
  }}>  
</Link>
```

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

withRouter

```
class MyComponent extends Component {  
  render() {  
    const { match, location, history } = this.props;  
  
    return <div>You are now at {location.pathname}</div>;  
  }  
}  
  
withRouter(MyComponent);
```

DOCUMENTATION

reacttraining.com/react-router/

```
/catalog => <Catalog />
```

```
/contact => <Contact />
```

```
class App extends Component {  
  render() {  
    return (  
      <div>  
        <h2>Website 2019</h2>  
        <Catalog />  
        <Contact />  
      </div>  
    );  
  }  
}
```

`npm i react-router-dom`

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

```
class App extends Component {  
  render() {  
    return (  
      <div>  
        <h2>Website 2019</h2>  
        <Catalog />  
        <Contact />  
      </div>  
    );  
  }  
}
```

```
import {  
  BrowserRouter as Router,  
  Route  
} from "react-router-dom";  
  
class App extends Component {  
  render() {  
    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

```
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>
);
```

url: http://localhost:3000/catalog
match.url: /catalog

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

path=/catalog/a
path=/catalog/b

PARAMETERS ROUTERS

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



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



**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

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

App.js

```
import "./Style.css";  
  
const Hello = () => <h1 className="red">Hello</h1>;
```

CSS MODULES

CSS MODULES

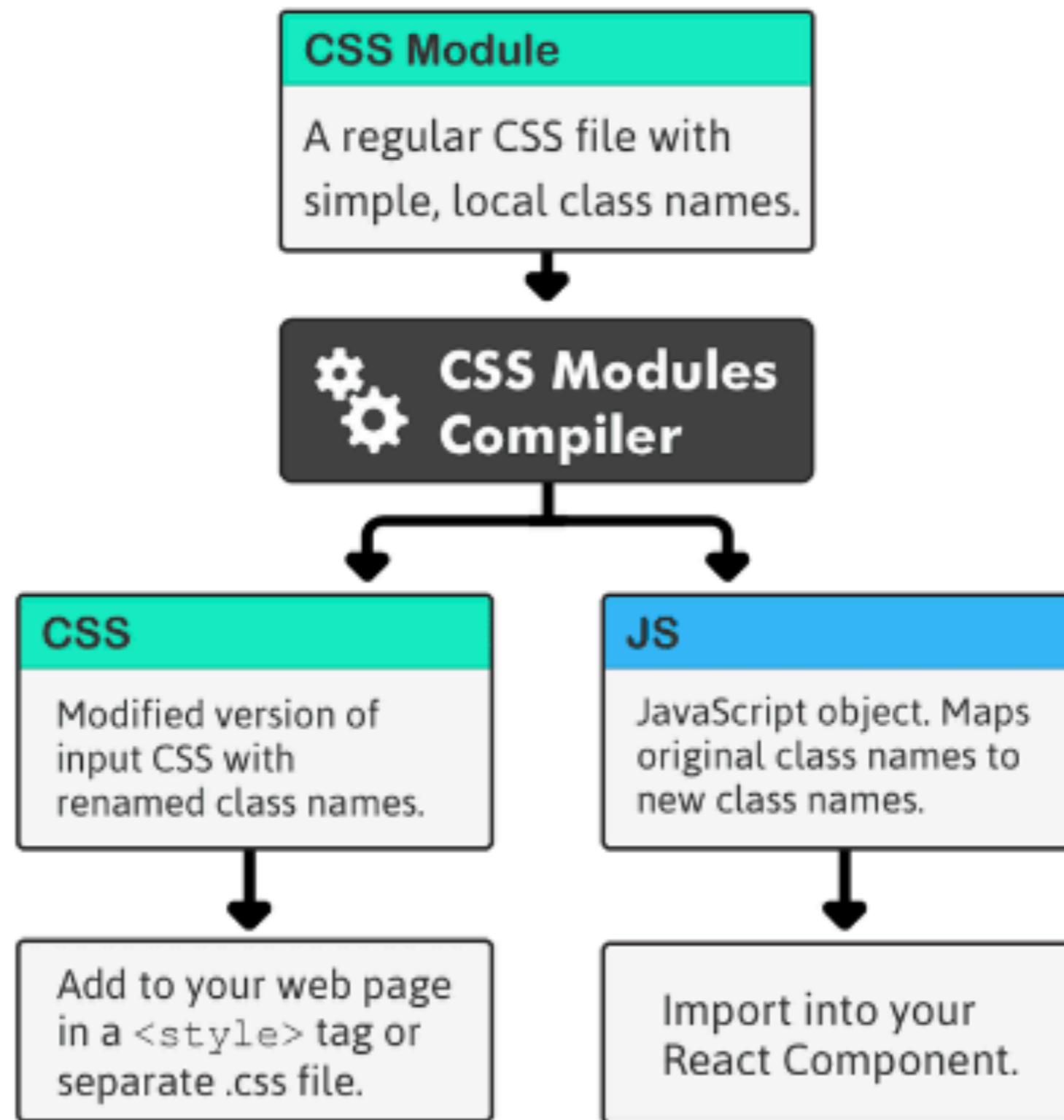
Style.module.css

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

App.js

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

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



CSS 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  
`;
```

STYLED COMPONENTS

```
<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>
```

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

```
class App extends Component {  
  render() {  
    return <Hello>Hello world</Hello>;  
  }  
}
```

STYLED COMPONENTS

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

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

Other libraries...

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

REFS

```
class App extends Component {  
  constructor() {  
    super();  
    this.inputRef = React.createRef();  
  }  
}
```

```
render() {  
  return (  
    <div>  
      <input type="text" ref={this.inputRef} />  
    </div>  
  );  
}  
}
```

REFS

```
class App extends Component {  
  constructor() {  
    super();  
    this.inputRef = React.createRef();  
  }  
  
  componentDidMount() {  
    this.inputRef.current.focus();  
  }  
  
  render() {  
    return (  
      <div>  
        <input type="text" ref={this.inputRef} />  
      </div>  
    );  
  }  
}
```

SUMMARY

THANKS