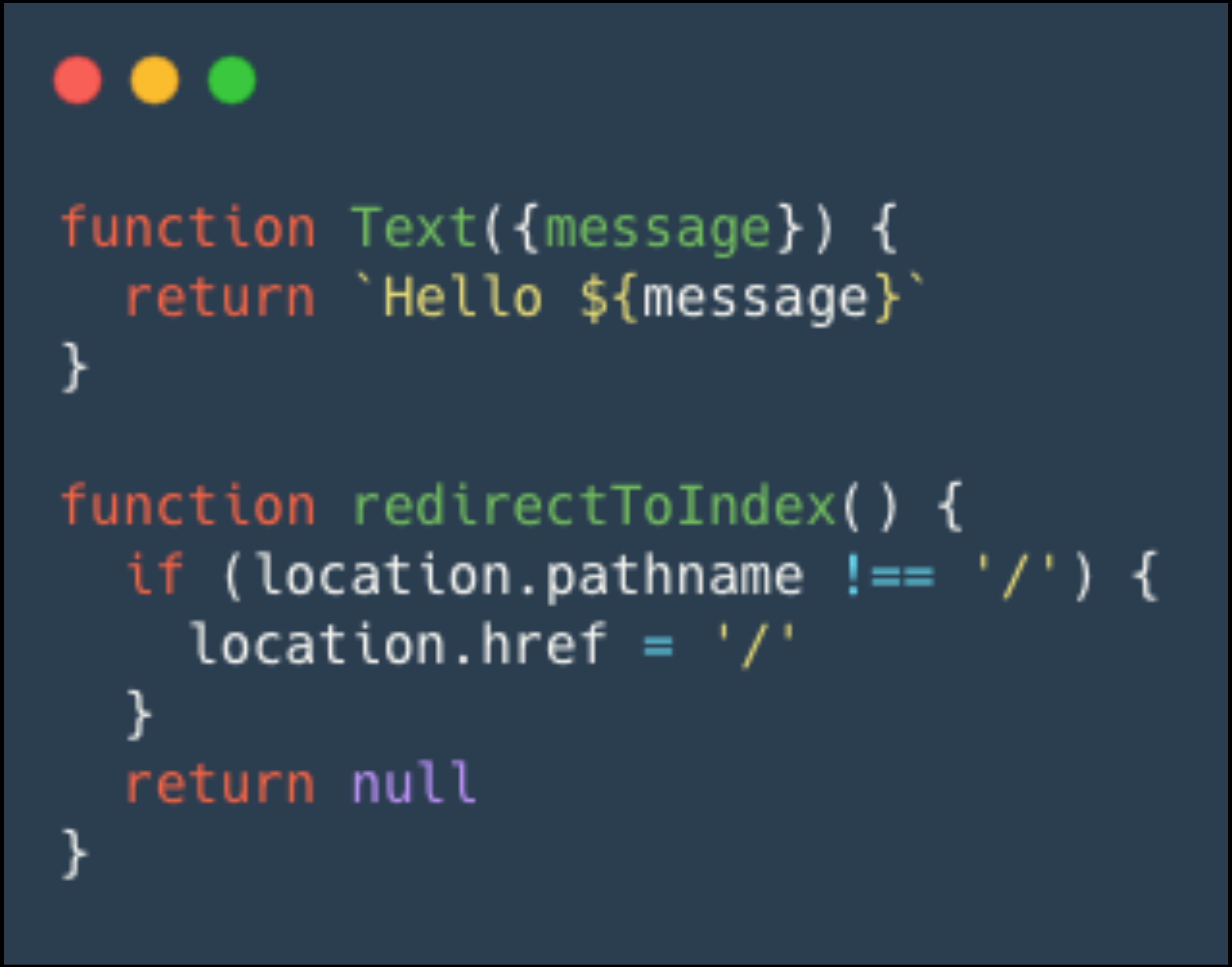
React Design Pattern

懷恩

Design Pattern

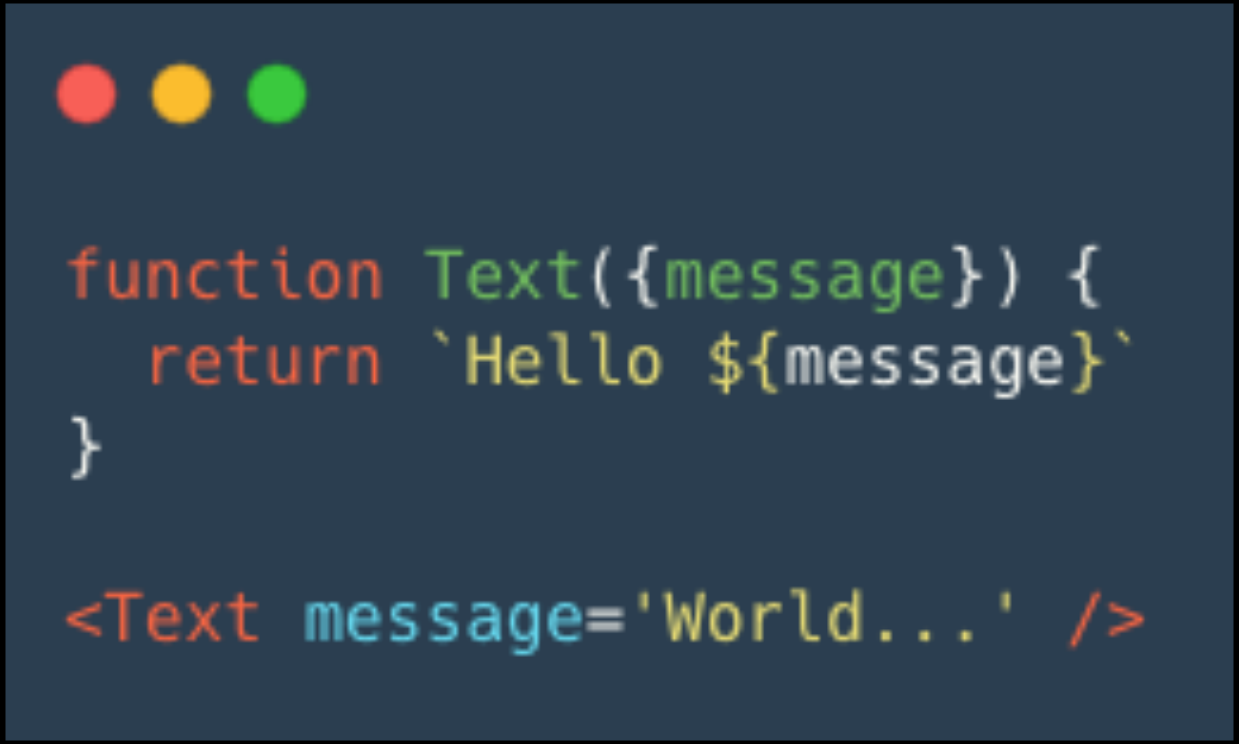
- Function Base
- JSX
- Props
- Conditional Render
- Array as Children
- *Proxy Component
- *Style Component
- Class Component
- Stateless Component
- Higher Order Component
- Render Props Component
- Function as Children Component

Function Base



```
function Text({message}) {
  return `Hello ${message}`
}

function redirectToIndex() {
  if (location.pathname !== '/') {
    location.href = '/'
  }
  return null
}
```

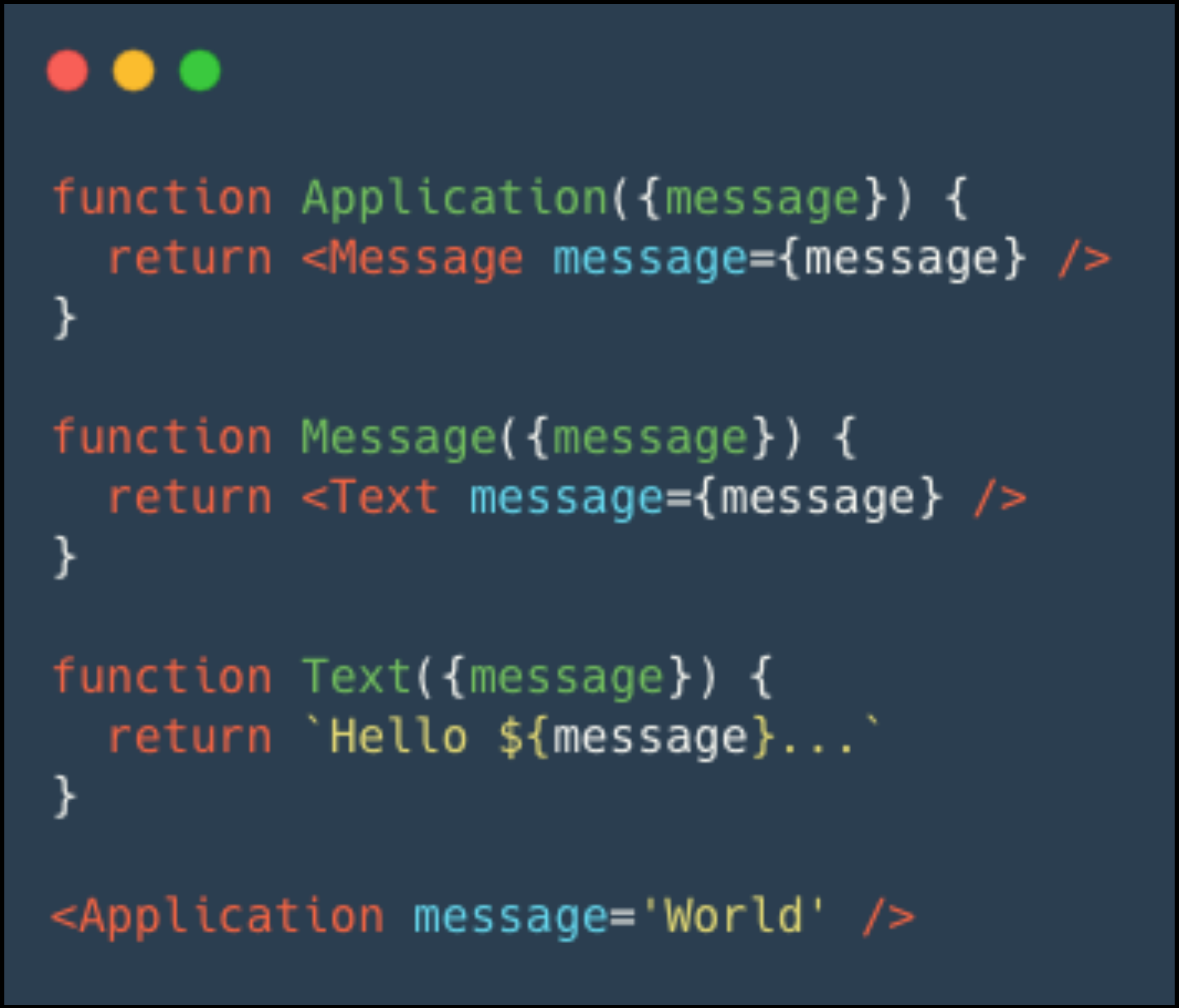
JSX



```
function Text({message}) {
  return `Hello ${message}`
}

<Text message='World...' />
```

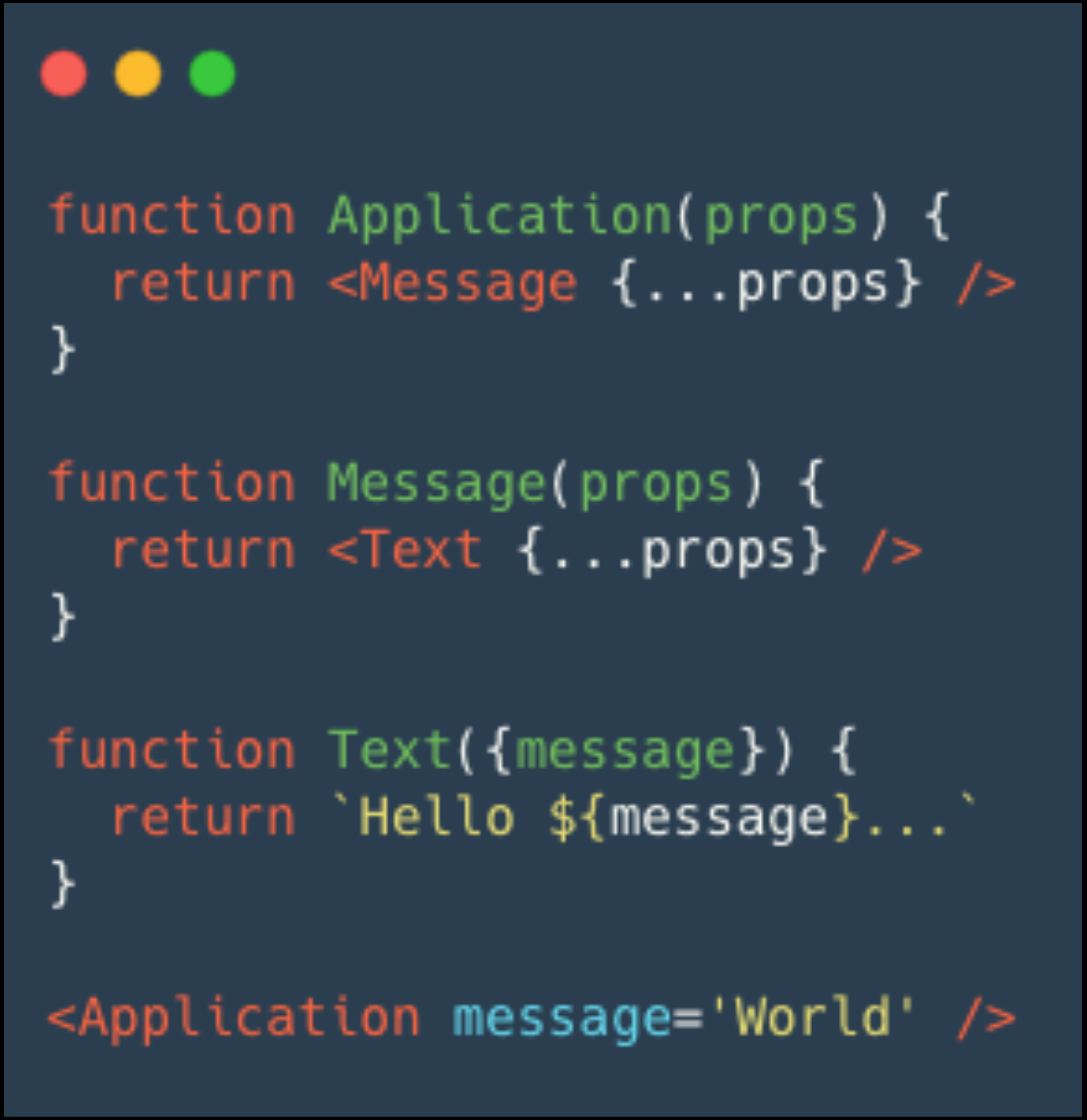
```
React.createElement(
  Text,
  { message: “World..." }
)
```

Props



```
function Application({message}) {
  return <Message message={message} />
}

function Message({message}) {
  return <Text message={message} />
}

function Text({message}) {
  return `Hello ${message}...`
}

<Application message='World' />
```

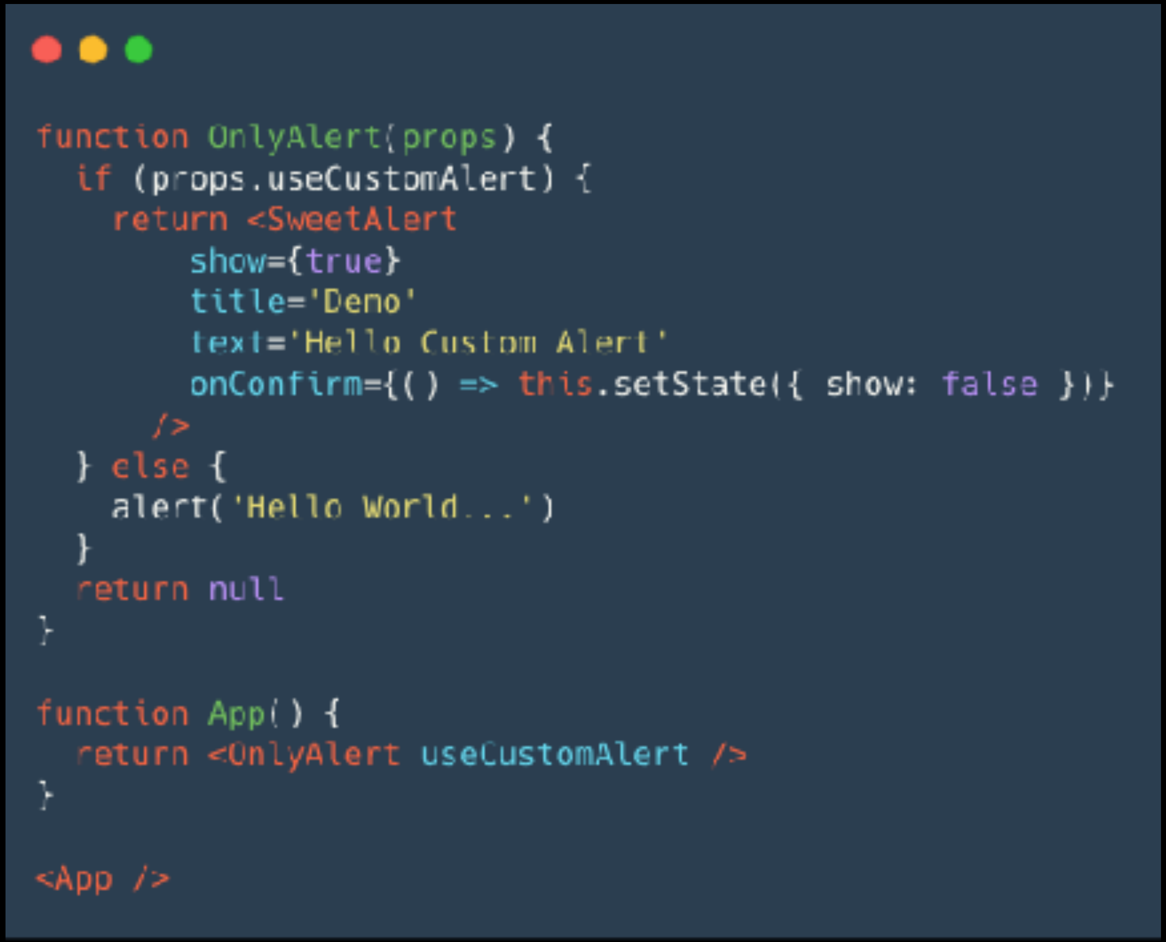
Destructured Props



```
function Application(props) {
  return <Message {...props} />
}

function Message(props) {
  return <Text {...props} />
}

function Text({message}) {
  return `Hello ${message}...`
}

<Application message='World' />
```

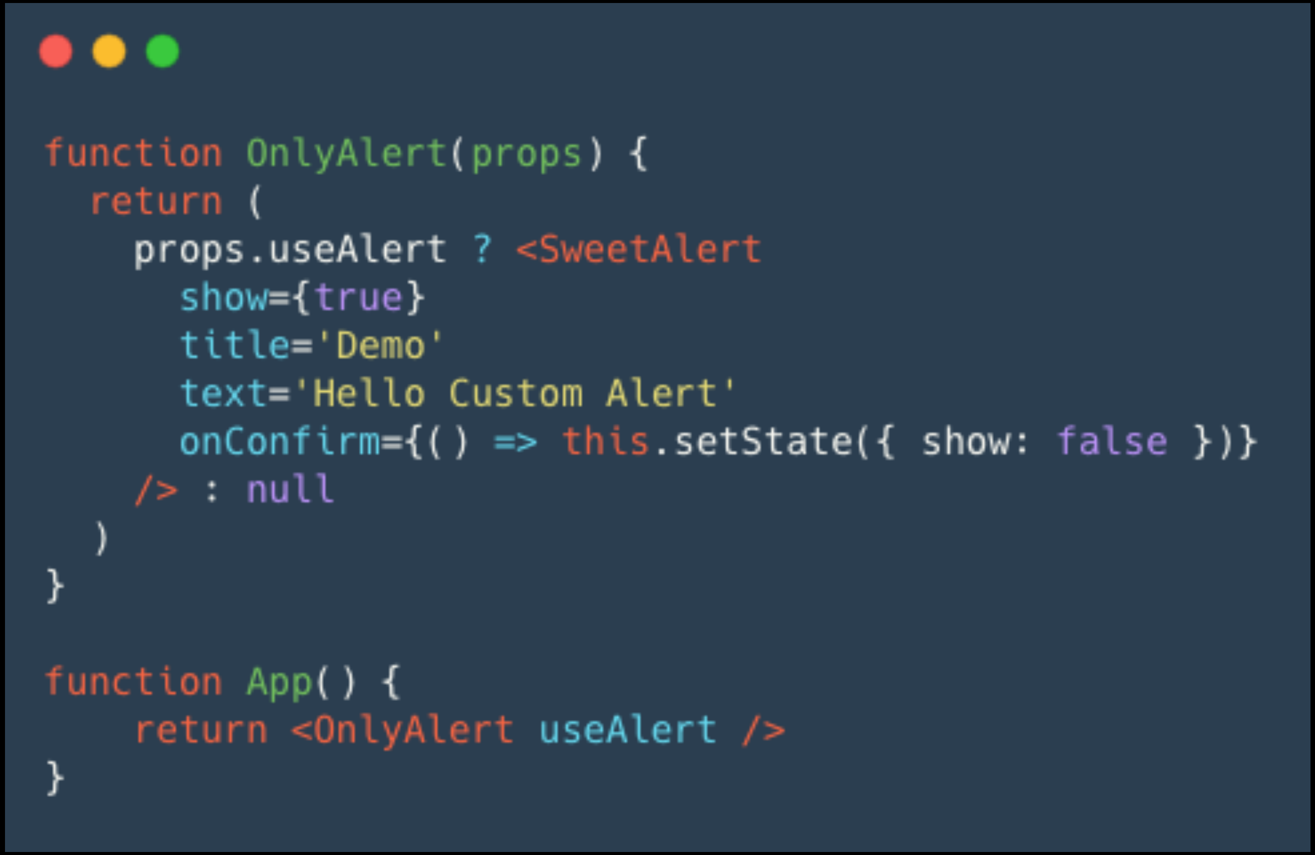
Condition Render

If else



```
function OnlyAlert(props) {
  if (props.useCustomAlert) {
    return <SweetAlert
        show={true}
        title='Demo'
        text='Hello Custom Alert'
        onConfirm={() => this.setState({ show: false })}
      />
  } else {
    alert('Hello World...')
  }
  return null
}

function App() {
  return <OnlyAlert useCustomAlert />
}

<App />
```

三元式

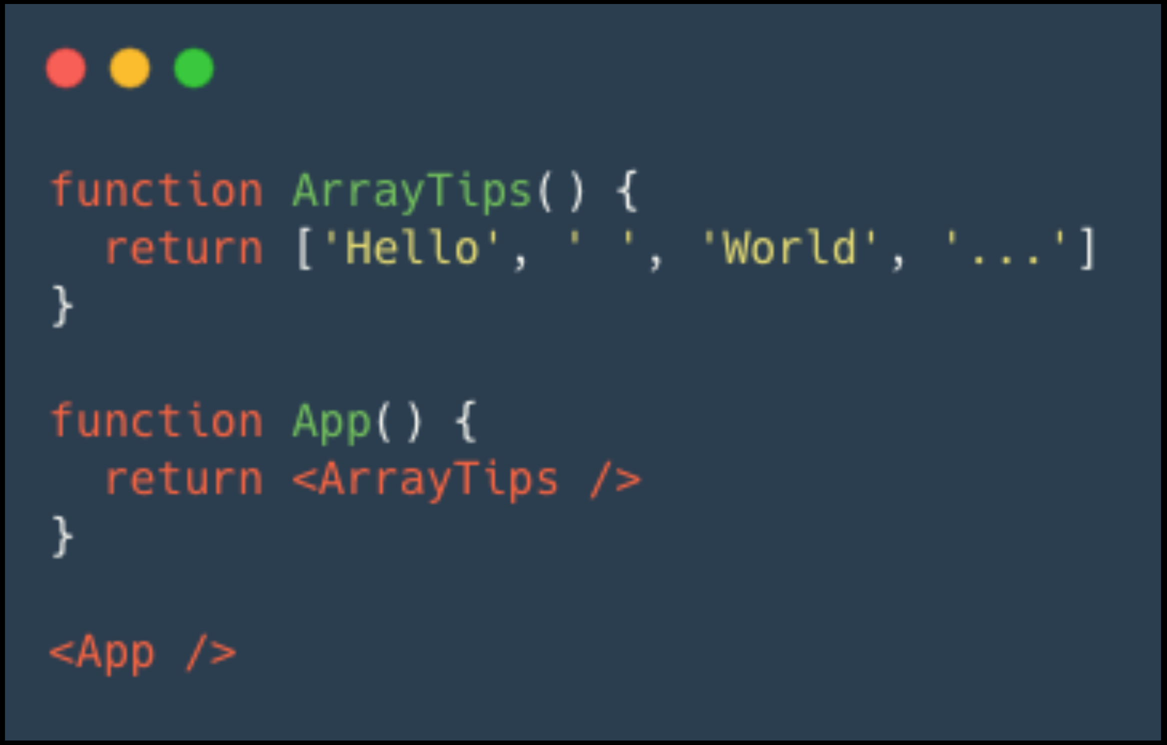


```
function OnlyAlert(props) {
  return (
    props.useAlert ? <SweetAlert
      show={true}
      title='Demo'
      text='Hello Custom Alert'
      onConfirm={() => this.setState({ show: false })}
    /> : null
  )
}

function App() {
    return <OnlyAlert useAlert />
}
```

Switch…

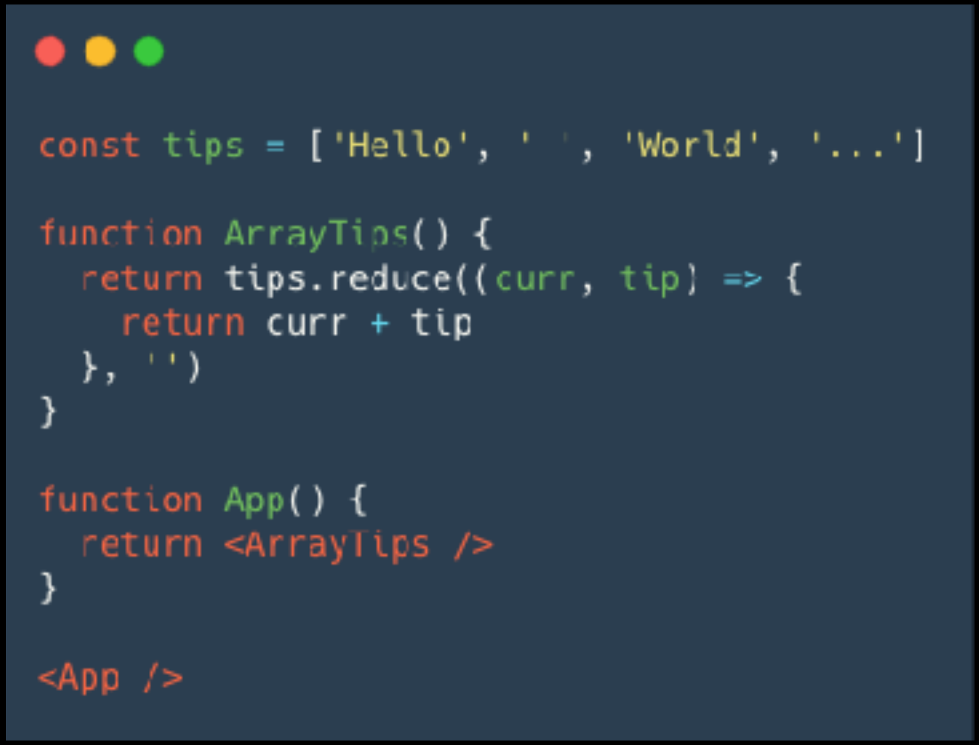
Array as Children

Basic



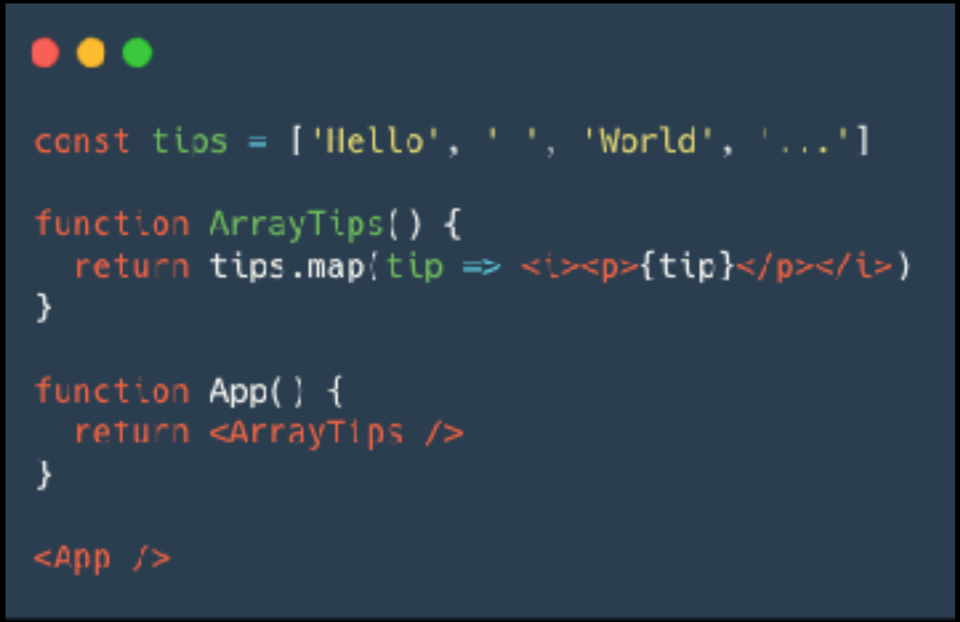
```
function ArrayTips() {
  return ['Hello', ' ', 'World', '...']
}

function App() {
  return <ArrayTips />
}

<App />
```

Reduce



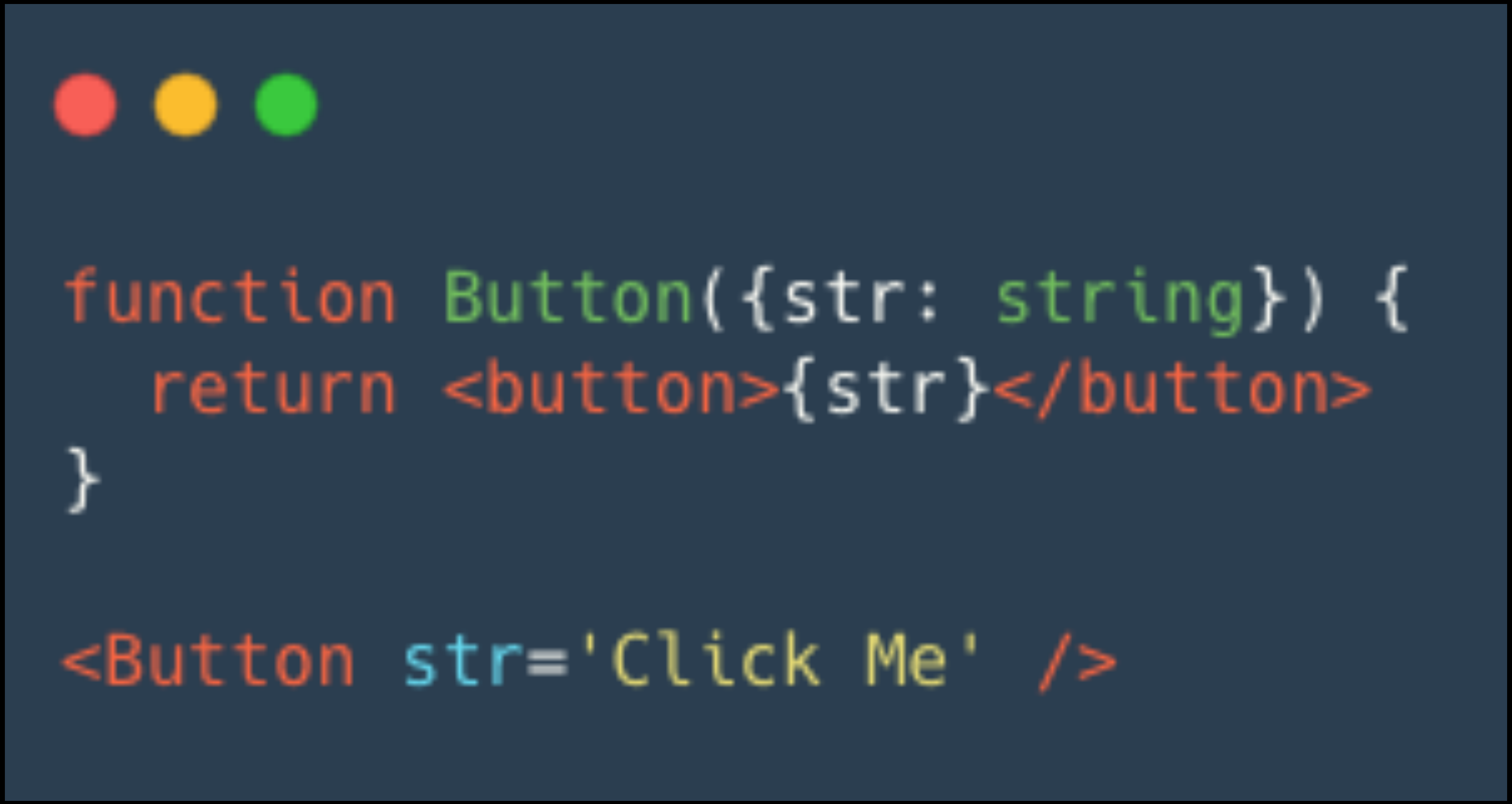
```
const tips = ['Hello', ' ', 'World', '...']

function ArrayTips() {
  return tips.reduce((curr, tip) => {
    return curr + tip
  }, '')
}

function App() {
  return <ArrayTips />
}

<App />
```

Map



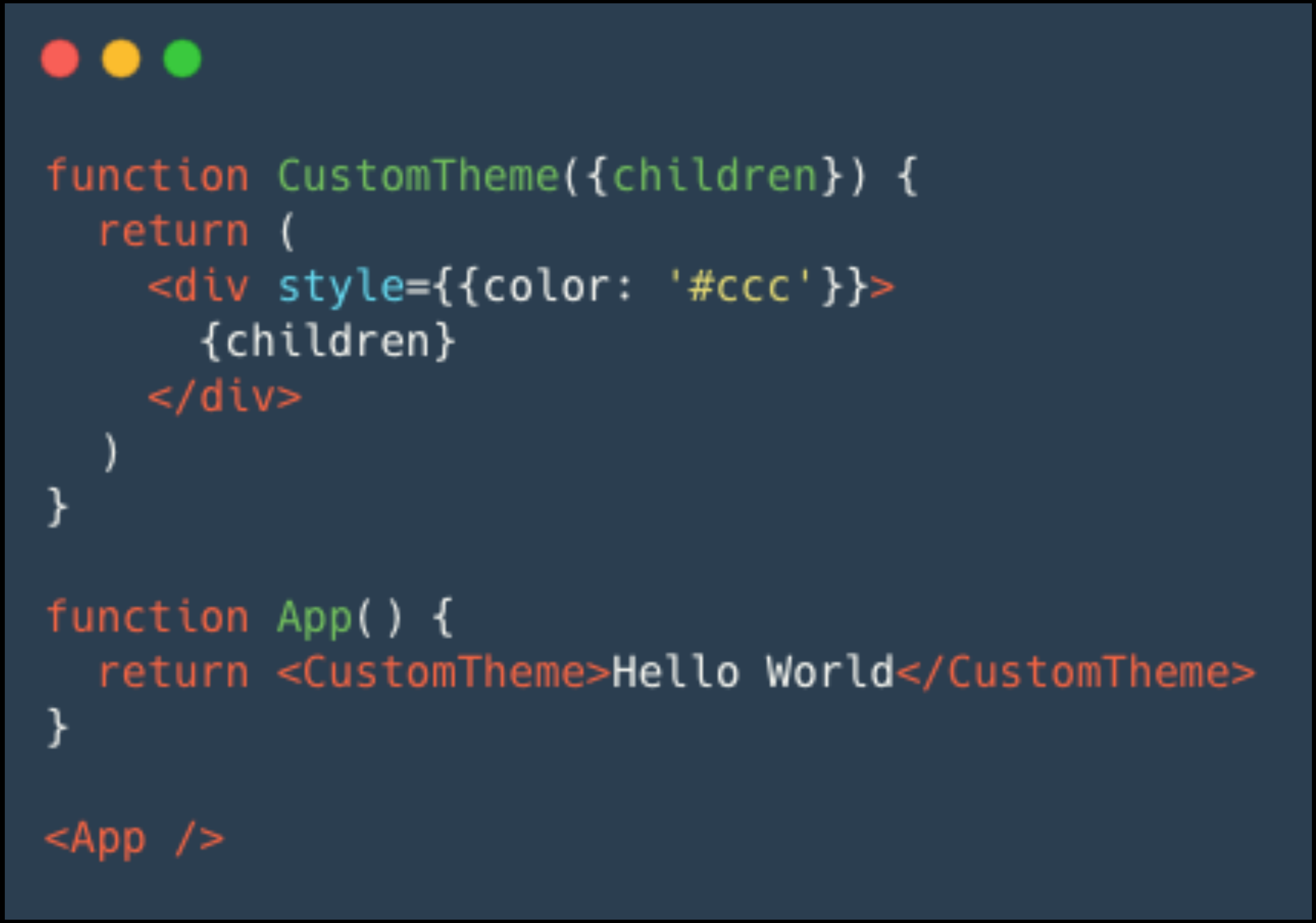
```
const tips = ['Hello', ' ', 'World', '...']

function ArrayTips() {
  return tips.map(tip => <i><p>{tip}</p></i>)
}

function App() {
  return <ArrayTips />
}

<App />
```

*Proxy Component



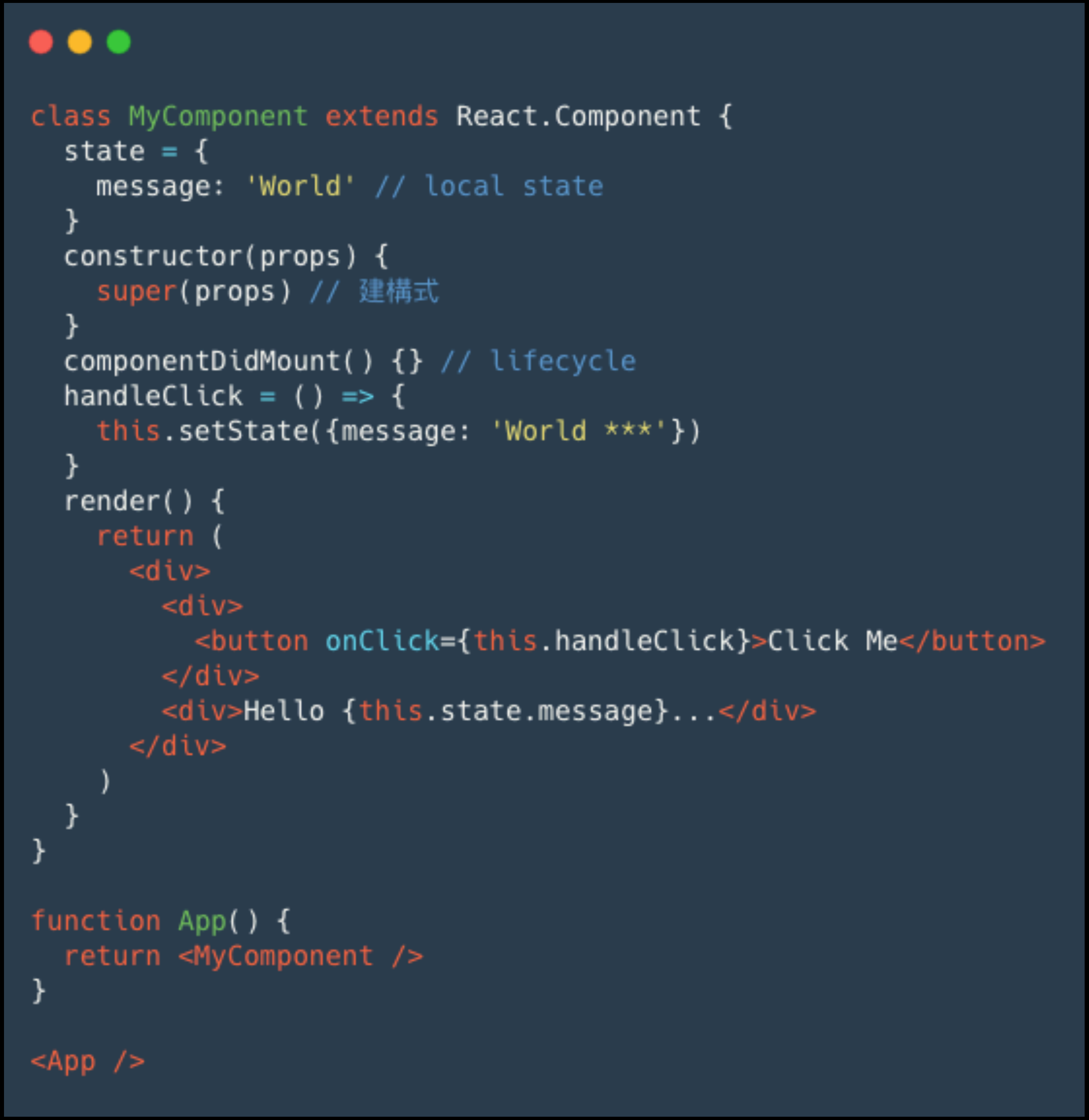
```
function Button({str: string}) {
  return <button>{str}</button>
}

<Button str='Click Me' />
```

*Style Component



```
function CustomTheme({children}) {
  return (
    <div style={{color: '#ccc'}}>
      {children}
    </div>
  )
}

function App() {
  return <CustomTheme>Hello World</CustomTheme>
}

<App />
```

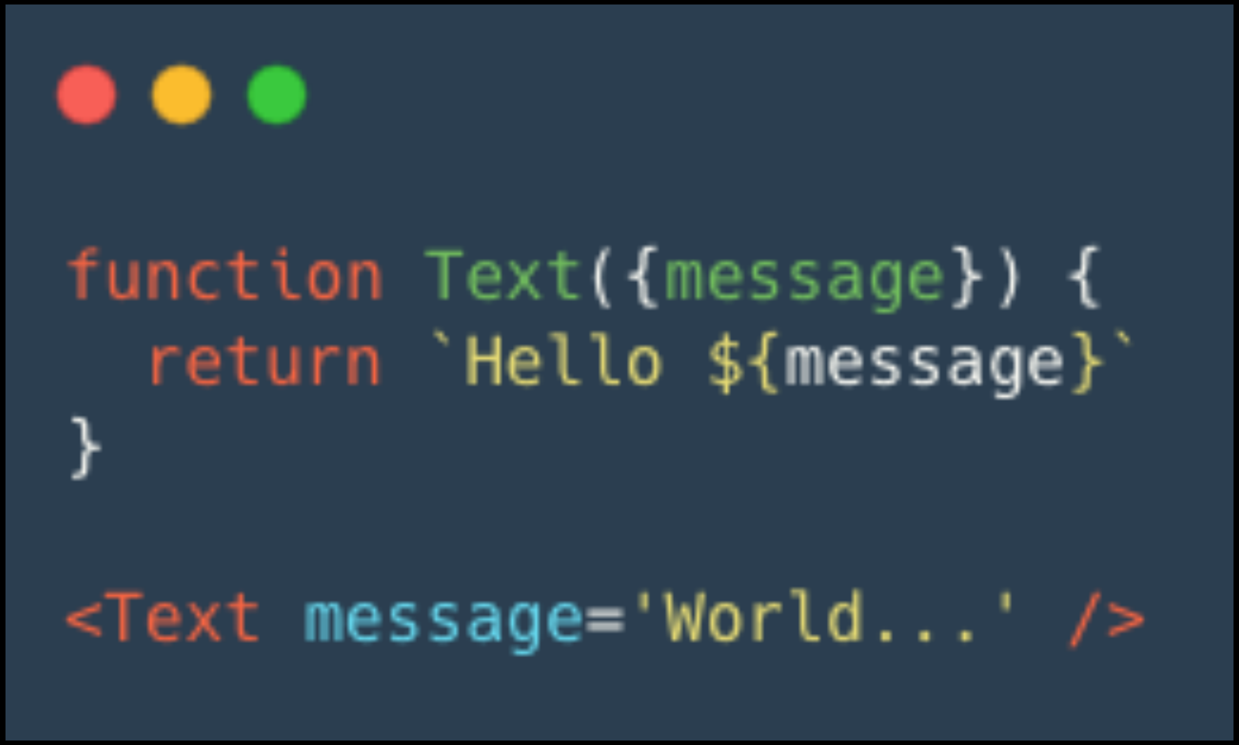
Class Component



```
class MyComponent extends React.Component {
  state = {
    message: 'World' // local state
  }
  constructor(props) {
    super(props) // 建構式
  }
  componentDidMount() {} // lifecycle
  handleClick = () => {
    this.setState({message: 'World ***'})
  }
  render() {
    return (
      <div>
        <div>
          <button onClick={this.handleClick}>Click Me</button>
        </div>
        <div>Hello {this.state.message}...</div>
      </div>
    )
  }
}

function App() {
  return <MyComponent />
}

<App />
```

- Constructor
- LocalState
- LifeCycle
- this.setState

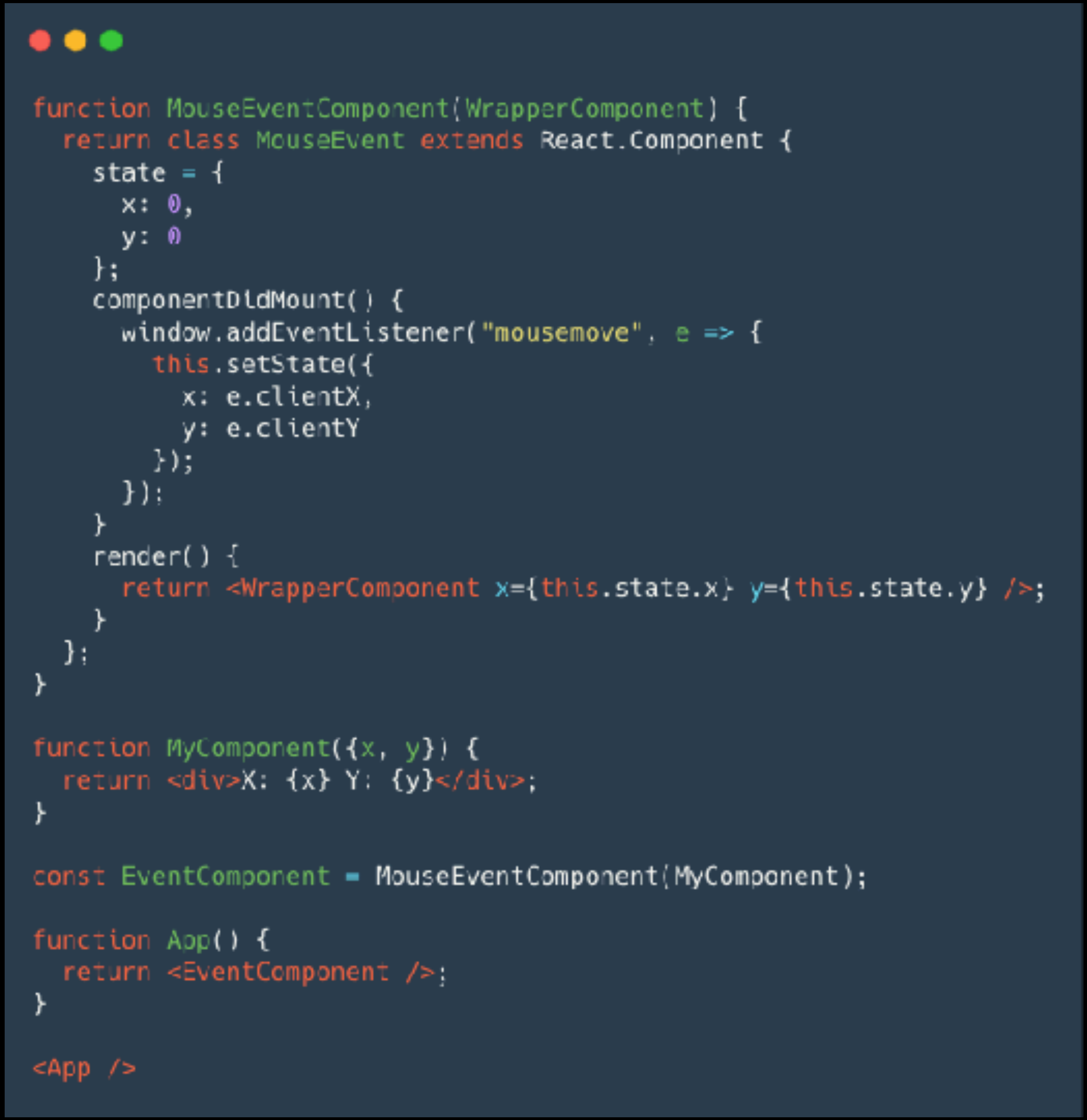
Stateless Component



```
function Text({message}) {
  return `Hello ${message}`
}

<Text message='World...' />
```

- 沒有 This
- 沒有 LocalState
- 只能接收 Props
- 通常用來渲染結果

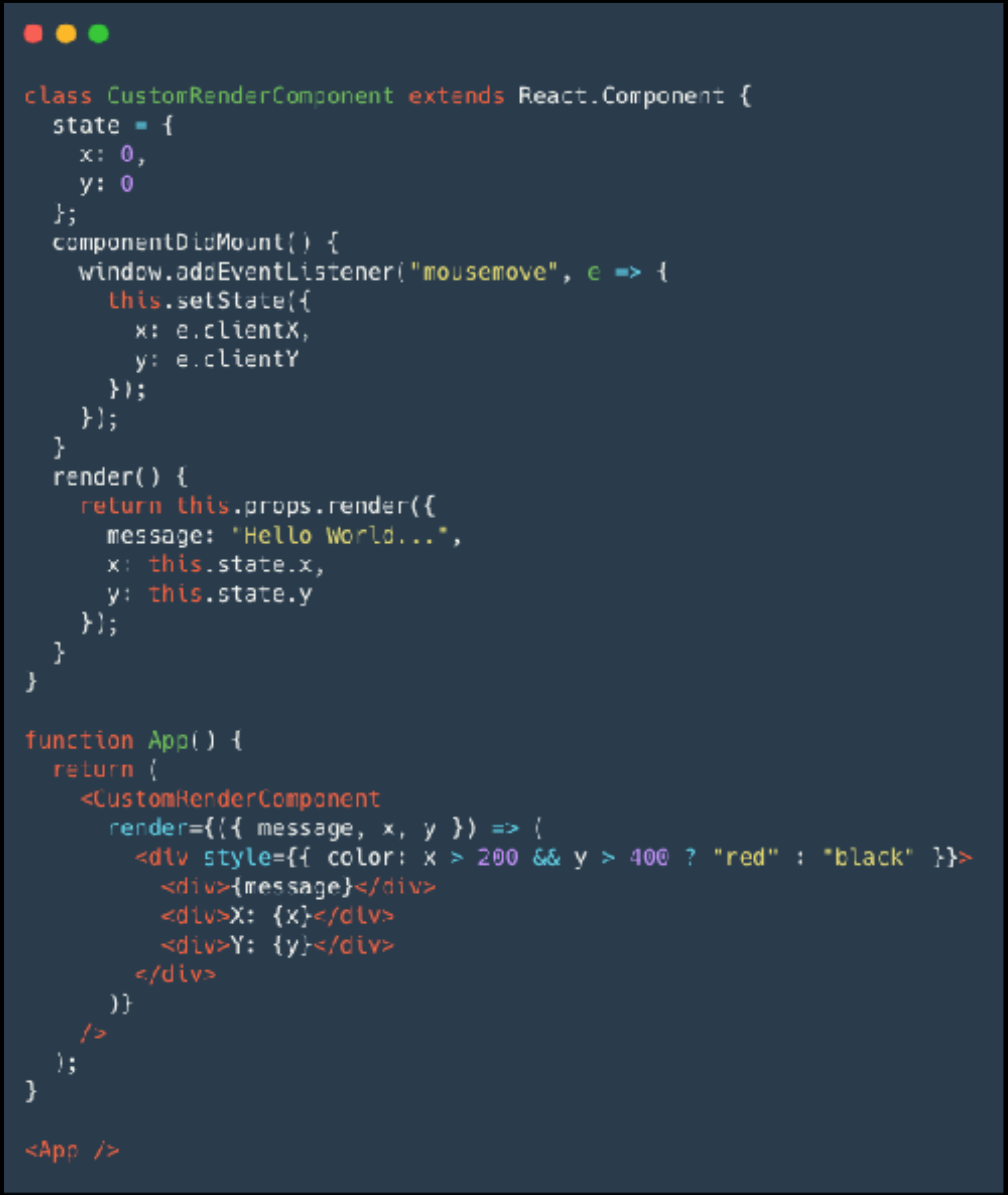
Higher Order Component



```
function MouseEventComponent(WrapperComponent) {
  return class MouseEvent extends React.Component {
    state = {
      x: 0,
      y: 0
    };
    componentDidMount() {
      window.addEventListener("mousemove", e => {
        this.setState({
          x: e.clientX,
          y: e.clientY
        });
      });
    }
    render() {
      return <WrapperComponent x={this.state.x} y={this.state.y} />;
    }
  };
}

function MyComponent({x, y}) {
  return <div>X: {x} Y: {y}</div>;
}

const EventComponent = MouseEventComponent(MyComponent);

function App() {
  return <EventComponent />;
}

<App />
```

CodeSandbox

Render Props Component



```
class CustomRenderComponent extends React.Component {
  state = {
    x: 0,
    y: 0
  };
  componentDidMount() {
    window.addEventListener("mousemove", e => {
      this.setState({
        x: e.clientX,
        y: e.clientY
      });
    });
  }
  render() {
    return this.props.render({
      message: "Hello World...",
      x: this.state.x,
      y: this.state.y
    });
  }
}

function App() {
  return (
    <CustomRenderComponent
      render={({ message, x, y }) => (
        <div style={{ color: x > 200 && y > 400 ? "red" : "black" }}>
          <div>{message}</div>
          <div>X: {x}</div>
          <div>Y: {y}</div>
        </div>
      )}
    />
  );
}

<App />
```

React Hooks

（挖坑）希望下回有能來聽聽或有機會與大家分享

Q&A