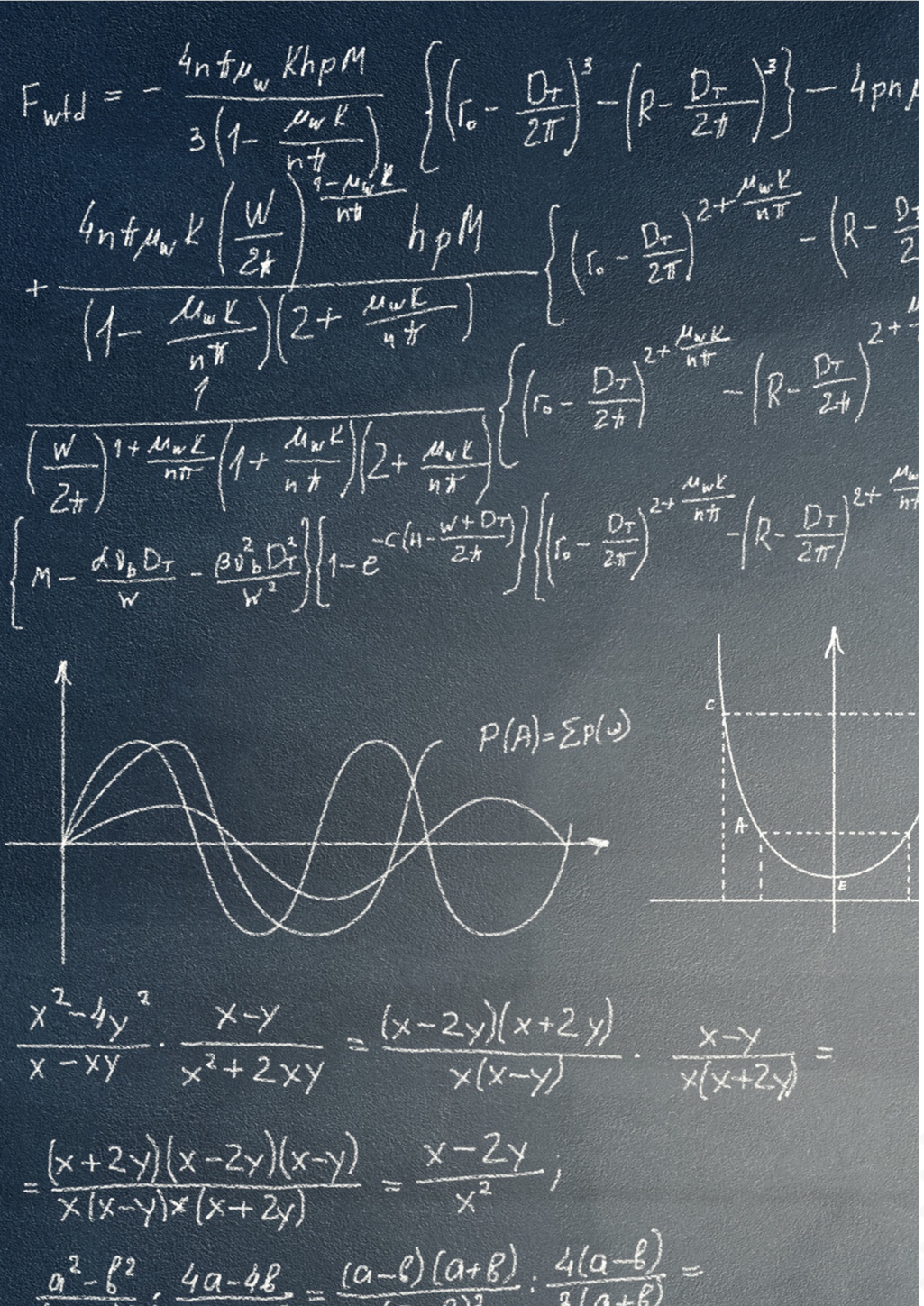
Understanding React Hooks

Peter Kellner

Developer, Consultant and Author

ReactAtScale.com @pkellner linkedin.com/in/peterkellner99

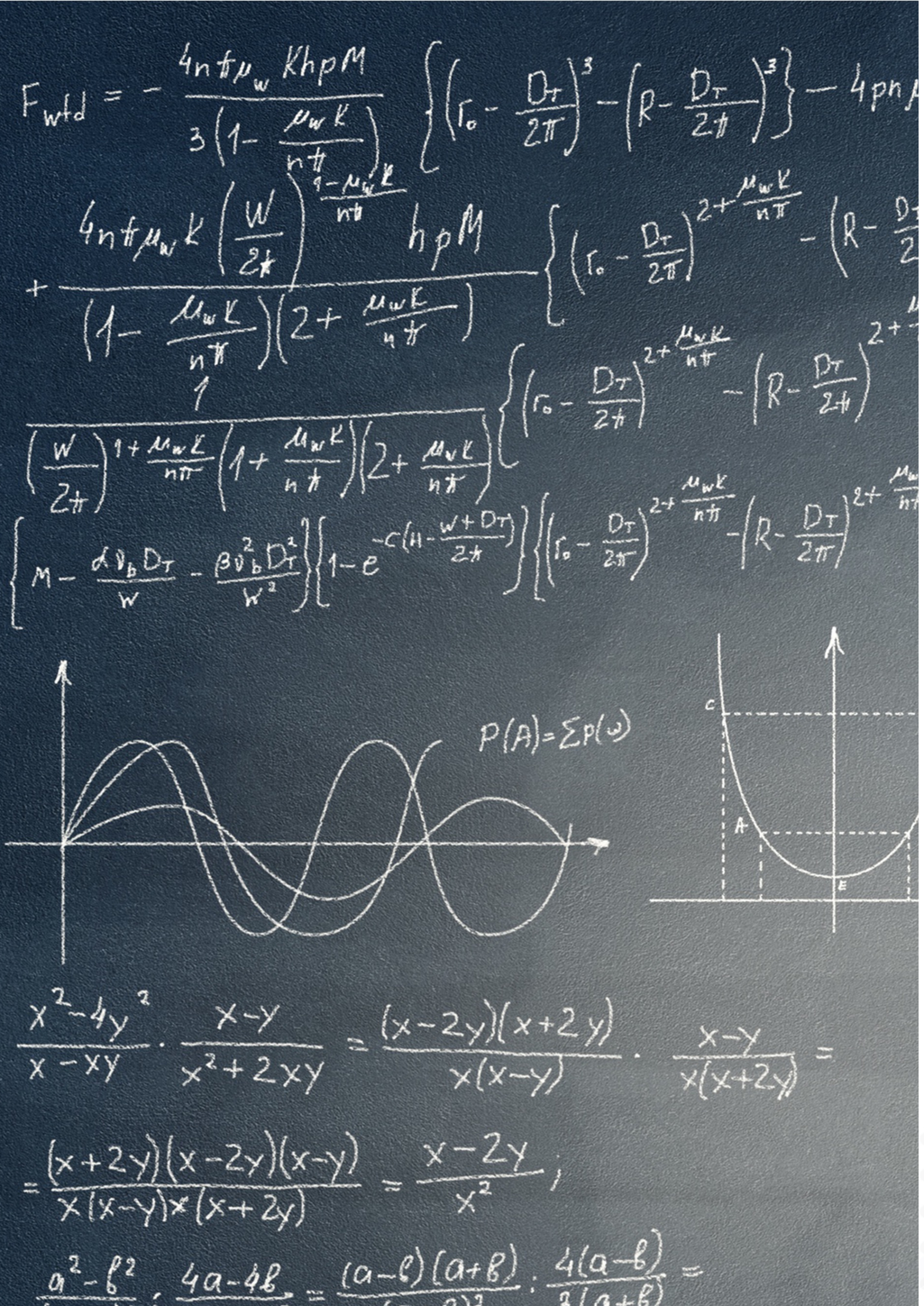


The “Theory of Operations” of React Hooks

Hooks are just JavaScript functions

Only valid when

- Used in other JavaScript functions
- Only at the top level of a function



The “Theory of Operations” of React Hooks

Hooks are just JavaScript functions

Only valid when

- Used in other JavaScript functions
- Only at the top level of a function

Two Most Commonly Used Hooks in React

useState: Manage state

useEffect: Manage lifecycle events

Two Most Commonly Used Hooks in React

useState: Manage state

```
const [value, setValue]
  = useState("Init")
```

useEffect: Manage lifecycle events

```
useEffect(() => {
  // code after render
})
```

Two Most Commonly Used Hooks in React

useState: Manage state

useEffect: Manage lifecycle events

useState()

```
useState("Init")
```

```
const []
   = useState("Init")
```

```
const [value]
   = useState("Init")
```

```
const [value, setValue]
   = useState("Init")
```

`useEffect()`

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

```
useEffect(() => {

})
```

```
useEffect(() => {
  // code after render
})
```

```
useEffect(() => {
// code after render
})
```

Coming up in This Module

- Build an app that calls useState and useEffect
- Create our own version of the useState hook
- Confirm our useState hook works in our app
- Review the “Three Rules of Hooks”

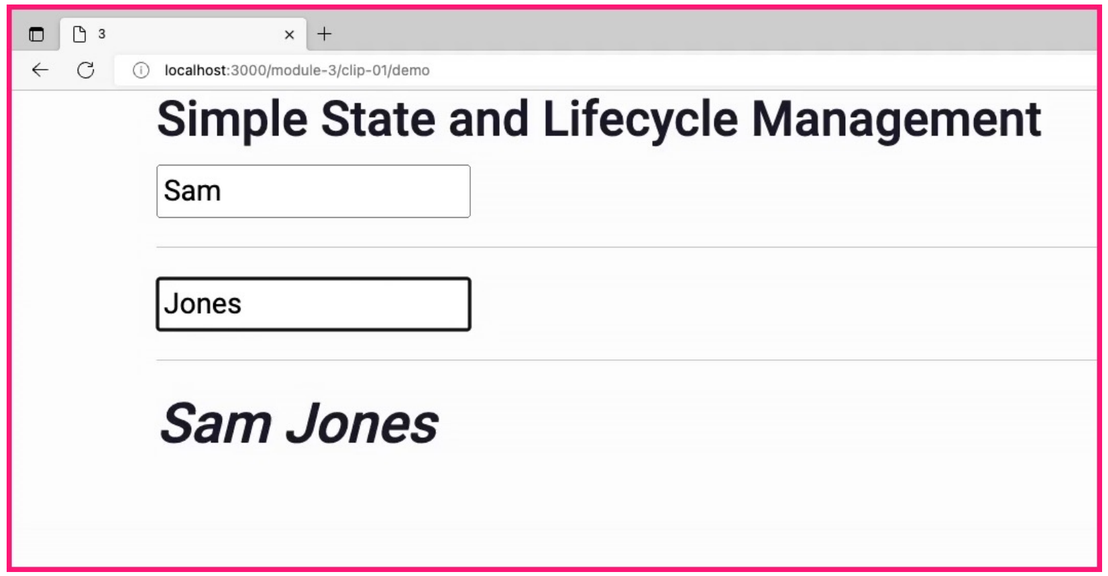
The Three Rules of React Hooks

Hooks can only be called inside of a React function component

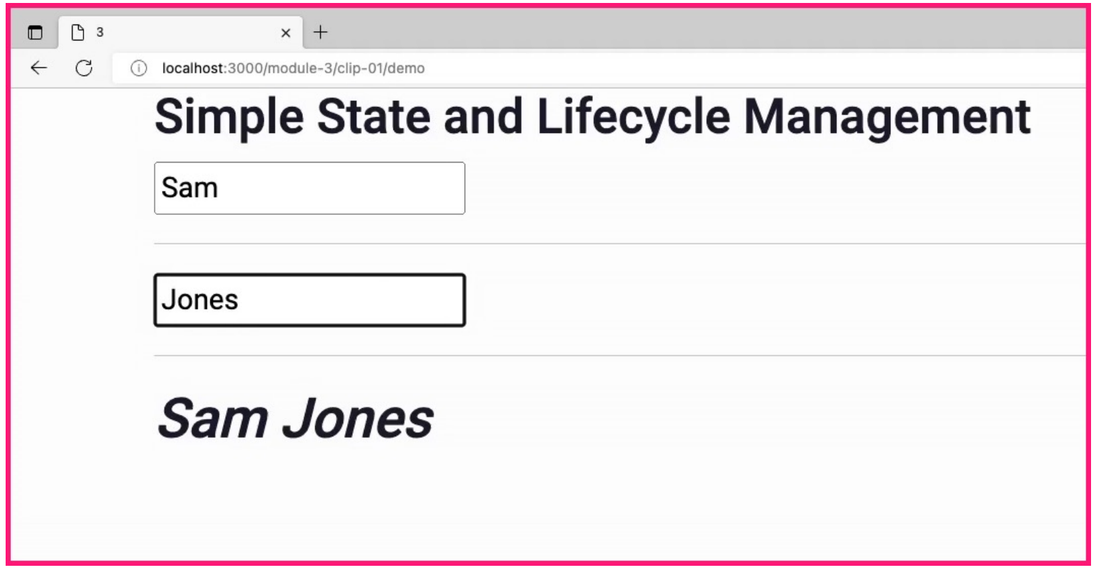
Hooks can not be called conditionally

Hooks can only be called at the top level of a function component

Demo



Implement useState for our input fields


3
localhost:3000/module-3/clip-01/demo
Simple State and Lifecycle Management
Sam
Jones
Sam Jones

What’s Happening Inside React

React App

setText1 called

A new value of text1 is stored internally in React

React notifies component tree to re-render

Any component that uses new state gets re-rendered

An App With Two-way Data Binding (Not React)

Two-way data binding is when the is a mechanism around input element that binds the input value to the onChange event

An App With Two-way Data Binding (Not React)

Two-way data binding is when the is a mechanism around input element that binds the input value to the onChange event

In React, no such mechanism exists

useEffect Allows for Side Effects in Function Components

Function components without side effects are pure

Pure means that you are guaranteed that passing same values in will get you same result

How Does useEffect Work?

Working with useEffect

```
useEffect()
```

Working with useEffect

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

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
})
```

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
})
```

Function passed into useEffects first parameter

- **Changes environment around the component**
- **Updates the component state**

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
})
```

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
  return () => {
    // code that executes before component leaves screen
  }
})
```

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
  return () => {
    // code that executes before component leaves screen
  }
})
```

Component cleanup

- **Unsubscribing to events**
- **Cancelling timers**

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
  return () => {
    // code that executes before component leaves screen
  }
})
```

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
  return () => {
    // code that executes before component leaves screen
  }
})
```

How often does useEffect run?

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
  return () => {
    // code that executes before component leaves screen
  }
})
```

How often does useEffect run?

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
  return () => {
    // code that executes before component leaves screen
  }
}, [])
```

How often does useEffect run?

The second parameter to useEffect, called the dependency array determines how often useEffect is run

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
  return () => {
    // code that executes before component leaves screen
  }
}, [])
```

How often does useEffect run?

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
  return () => {
    // code that executes before component leaves screen
  }
}, [])
```

How often does useEffect run?

If second parameter is undefined or left out, useEffect renders every execution

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
  return () => {
    // code that executes before component leaves screen
  }
}, [])
```

How often does useEffect run?

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
  return () => {
    // code that executes before component leaves screen
  }
}, [])
```

How often does useEffect run?

If second parameter an empty array “[]”, useEffect renders every time the component executes

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
  return () => {
    // code that executes before component leaves screen
  }
}, [])
```

How often does useEffect run?

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
  return () => {
    // code that executes before component leaves screen
  }
}, [])
```

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
  return () => {
    // code that executes before component leaves screen
  }
}, [])
```

How often does useEffect run?

Array items passed into the dependency array (typically state) will cause the component to rerender on any of those values changing

Working with useEffect

```
useEffect(() => {
  // Code that executes after component render
  return () => {
    // code that executes before component leaves screen
  }
}, [])
```

React Hooks Usage

Being a React Hook is about beginning a function name with “usr”

It’s easy to build hooks that don’t behave as expected

The React team has provided us with eslint rules for using React Hooks called “the rules of hooks”

The Most Common Hooks, useEffect and useState

useEffect

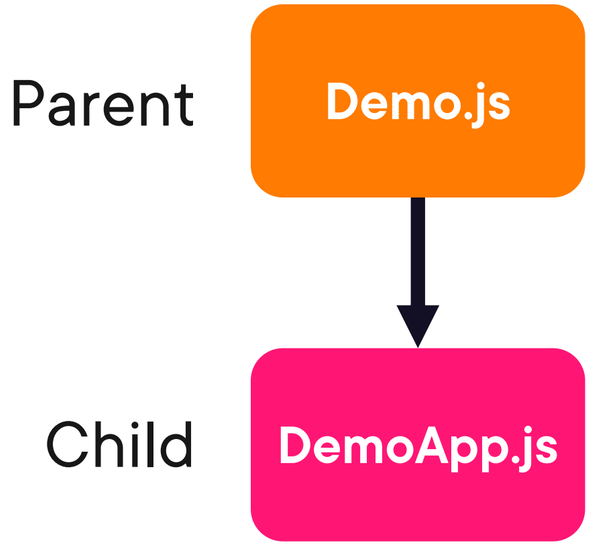
Easy to understand and use. Very straight forward

useState

Clear what it does, but how it works feels magical

Creating your own useState hook is advanced

OK not to follow all the details


Parent
Demo.js
Child
DemoApp.js

Takeaways

What React hooks are

React hooks usage in apps

Built our own useState hook

Coming up, a real-world conference app