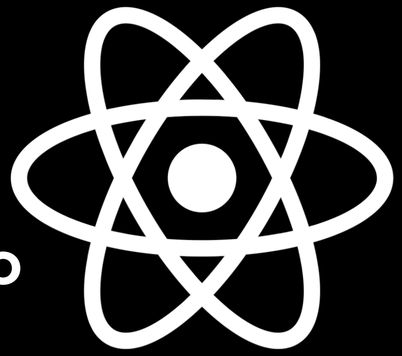
WELCOME TO



REACT 11

LESSON 3

WELCOME TO

STATE PT 1

BUT FIRST

CLASS-BASED COMPONENTS

WHAT ARE CLASS-BASED COMPONENTS

WHAT ARE CLASS-BASED COMPONENTS

Components made using the javascript class template

WHAT ARE CLASS-BASED COMPONENTS

Components made using the javascript class template

FUNCTIONAL COMPONENT

```
const MyAwecomeComponent = ( ) => {
    render ( ) {
        // render jsx here
    }
}
```

WHAT ARE CLASS-BASED COMPONENTS

Components made using the javascript class template

FUNCTIONAL COMPONENT

```
const MyAwecomeComponent = ( ) => {
    render ( ) {
        // render jsx here
    }
}
```

CLASS-BASED COMPONENT

```
class MyAwecomeComponent extends React.Component {
    render ( ) {
        // render jsx here
    }
}
```

WHAT ARE CLASS-BASED COMPONENTS

```
class MyAwecomeComponent extends React.Component {
    render ( ) {
        // render jsx here
    }
}
```

USUAL CLASS-BASED COMPONENT SECTIONS

```
class MyAwecomeComponent extends React.Component {
    // constructor (props and state init)
    // life cycle methods go here
     render ( ) {
         // render jsx here
     }
}
```

WHAT'S NEW?

```
const MyAwecomeComponent = (props) => {
    render ( ) {
        // render jsx here
    }
}
```

WHAT'S NEW?

```
class MyAwecomeComponent extends React.Component {
    constructor (props) {
      super(props)

    }
    render (
        // render jsx here
    )
}
```

WHAT'S NEW?

How do we access the props is it just <props>?

```
class MyAwecomeComponent extends React.Component {
    constructor (props) {
      super(props)

    }
    render (
        // render jsx here
    )
}
```

WHAT'S NEW?

How do we access the props is it just <props>?

No but actually Yes

```
class MyAwecomeComponent extends React.Component {
    constructor (props) {
      super(props)

    }
    render (
        // render jsx here
    )
}
```

WHAT'S NEW?

How do we access the props is it just <props>?

No but actually Yes

```
class MyAwecomeComponent extends React.Component {
    constructor (props) {
      super(props)
      console.log(this.props)
    }
    render (
        // render jsx here
    )
}
```

WHAT'S NEW?

How do we access the props is it just <props>?

No but actually Yes

What is super( )

```
class MyAwecomeComponent extends React.Component {
    constructor (props) {
      super(props)
      console.log(this.props)
    }
    render (
        // render jsx here
    )
}
```

WHAT'S NEW?

How do we access the props is it just <props>?

No but actually Yes

What is super( )

Why do we need to pass props to super( )

```
class MyAwecomeComponent extends React.Component {
    constructor (props) {
      super(props)
      console.log(this.props)
    }
    render (
        // render jsx here
    )
}
```

STATE

STATE

a javascript object that represents the parts of an app that can change

HOW DO WE USE STATE

```
class MyAwecomeComponent extends React.Component {
    constructor (props) {
      super(props)

    }
    render (

        // render jsx here
    )
}
```

HOW DO WE USE STATE

```
class MyAwecomeComponent extends React.Component {
    constructor (props) {
      super(props)
      this.state = {}
    }
    render (

        // render jsx here
    )
}
```

HOW DO WE USE STATE

```
class MyAwecomeComponent extends React.Component {
    constructor (props) {
      super(props)
      this.state = {}
    }
    render (
        // this.state.users
        // render jsx here
    )
}
```

HOW DO WE USE STATE

```
class MyAwecomeComponent extends React.Component {

    state = {}

    render (

        // render jsx here

    )
}
```

HOW DO WE USE STATE

```
class MyAwecomeComponent extends React.Component {

    state = {}

    render (
        // this.state.users
        // render jsx here
    )
}
```

HOW DO WE USE STATE

```
class MyAwecomeComponent extends React.Component {
    constructor (props) {
      super(props)
      this.state = {
        users: [],
        isLoading: true
        // etc
      }
    }
    render (

        // render jsx here
    )
}
```

HOW DO WE USE STATE

```
class MyAwecomeComponent extends React.Component {
    constructor (props) {
      super(props)
      this.state = {
        users: [],
        isLoading: true
        // etc
      }
    }
    render (
        // this.state.users
        // render jsx here
    )
}
```

HOW DO WE USE STATE

```
class MyAwecomeComponent extends React.Component {

  state = {
    users: [],
    isLoading: true
    // etc
  }
  render (

    // render jsx here
  )
}
```

HOW DO WE USE STATE

```
class MyAwecomeComponent extends React.Component {

  state = {
    users: [],
    isLoading: true
    // etc
  }
  render (
    // this.state.users
    // render jsx here
  )
}
```

PROPS VS STATE

PROPS VS STATE

both are javascript objects

PROPS VS STATE

both are javascript objects

both influences render

PROPS VS STATE

both are javascript objects

both influences render

props get passed to the component

PROPS VS STATE

both are javascript objects

both influences render

props get passed to the component

state is managed within the component

HOW DO WE CHANGE THE STATE?

HOW DO WE CHANGE THE STATE?

stay tuned!

HOW DO WE CHANGE THE STATE?

stay tuned!

but yeah it’s actually setState()

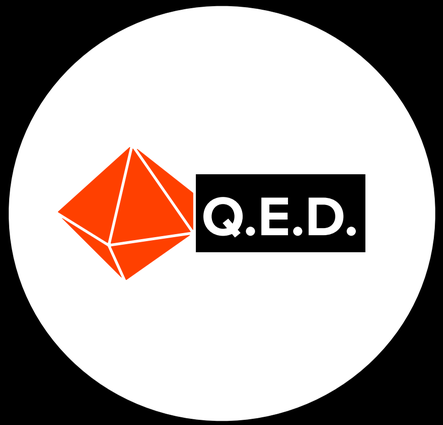
NEXT UP ON REACT 11

COMPONENT LIFE CYCLE AND LIFE CYCLE METHODS

EXERCISE

HOT AND COLD

BEFORE OUR LAST TOPIC
WHICH IS HTTP


Q.E.D.