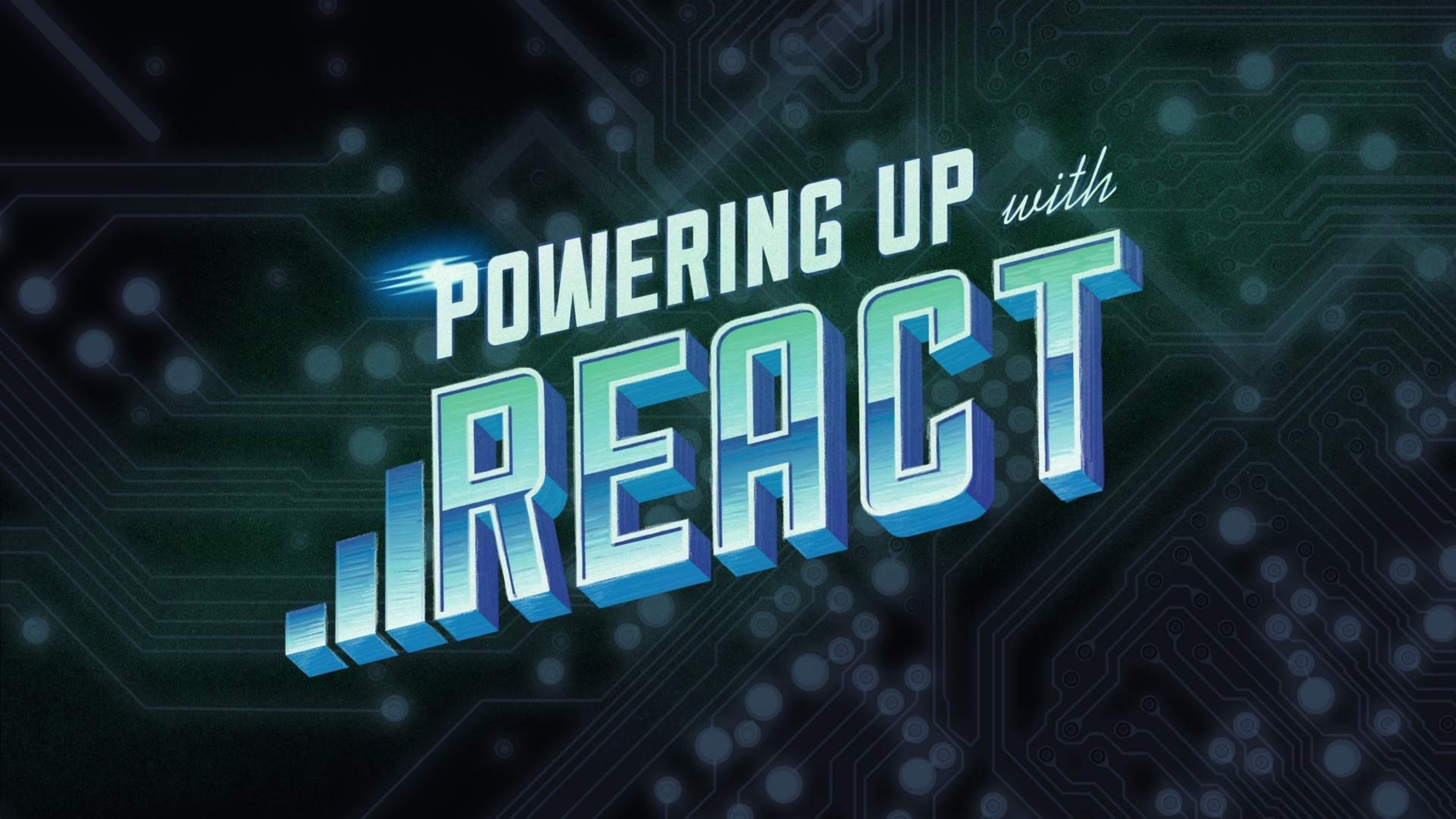

POWERING UP with
REACT

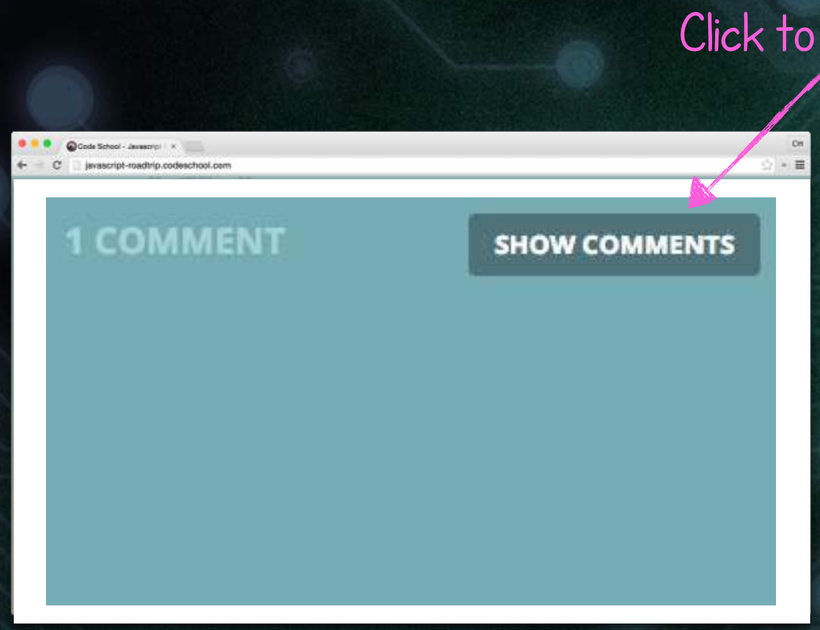
Level 3

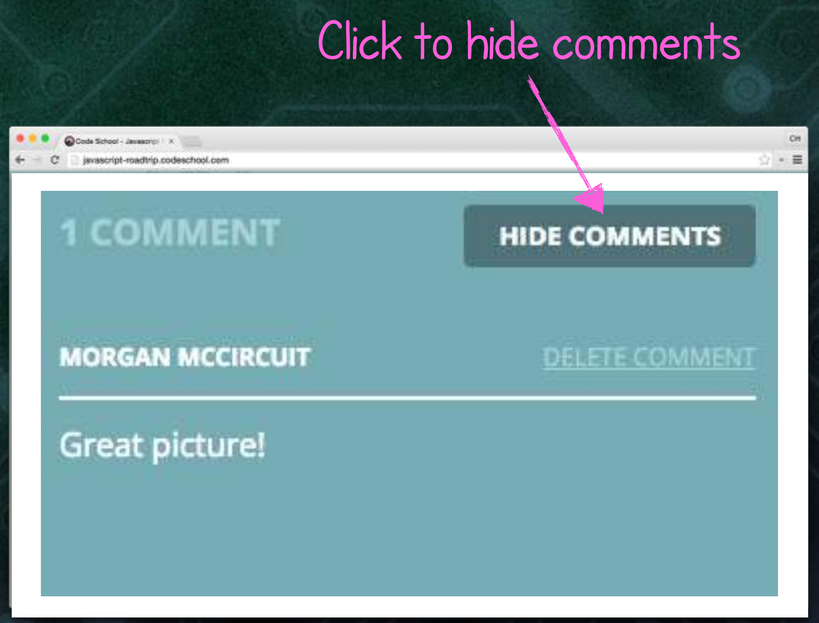
# Component State

## Handling Data Changes With State

# Show and Hide Comments

We'd like to add a button to the page that will let users toggle the comments.




How can we show and hide comments based on button clicks?

POWERING UP with REACT

# Different Ways to Manipulate the DOM

1. **Direct DOM Manipulation**

   jQuery, Backbone, etc.

2. **Indirect DOM Manipulation**

   React

# Direct DOM Manipulation

One way to manipulate the DOM API is by modifying it **directly** via JavaScript in response to browser events.


Example using **jQuery**:

```
$('.show-btn').on('click', function() {
  $('.comment-list').show();
})

$('.hide-btn').on('click', function() {
  $('.comment-list').hide();
})
```

Manually manipulating the DOM

# Indirect DOM Manipulation

In React, we **don't modify the DOM directly**. Instead, we modify a component state object in response to user events and let React handle updates to the DOM.


Example using **React:**

```
render() {
  if (this.state.showComments) {
    // code displaying comments
  } else {
    // code hiding comments
  }
}
```

Display logic based on state

# How to Use State in a Component

The **state** is a JavaScript object that lives inside each component. We can access it via *this.state*.

```
class CommentBox extends React.Component {
  ...
  render() {
    const comments = this._getComments();
    if (this.state.showComments) {
      // add code for displaying comments
    }
    return(
      <div className="comment-box">
        <h4 className="h4">{this._getCommentsTitle(comments.length)}</h4>
        <div className="comment-list">{comments}</div>
      </div>
    );
  }
  ...
}
```

Create list of comments if state is true.

We also need to move these comments into the conditional.

# Showing Comments Only if State Is *true*

```
class CommentBox extends React.Component {
  ...
  render() {
    const comments = this._getComments();
    let commentNodes;
    if (this.state.showComments) {
      commentNodes = <div className="comment-list">{comments}</div>;
    }
    return(
      <div className="comment-box">
        <h4 className="h4">{this._getCommentsTitle(comments.length)}</h4>
        {commentNodes}
      </div>
    );
  }
  ...
}
```

Now being displayed based on component's state!

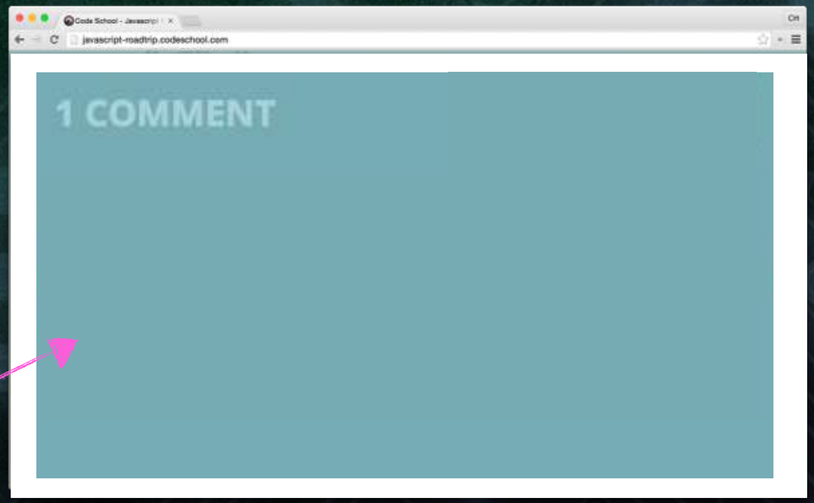
# Hiding Comments on the Initial State

We set the initial state of our component in the class constructor.

```
class CommentBox extends React.Component {

  constructor() {
    super();

    this.state = {
      showComments: false
    };
  }

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

super() must be called in our constructor.

Initial state hides comments.




# How to Update a Component's State

We don't assign to the state object directly — instead, we call *setState* by passing it an object.

Setting state this way won't work.

```
this.state.showComments = true
```

Updates the showComments property
and re-renders component

```
this.setState({showComments: true })
```

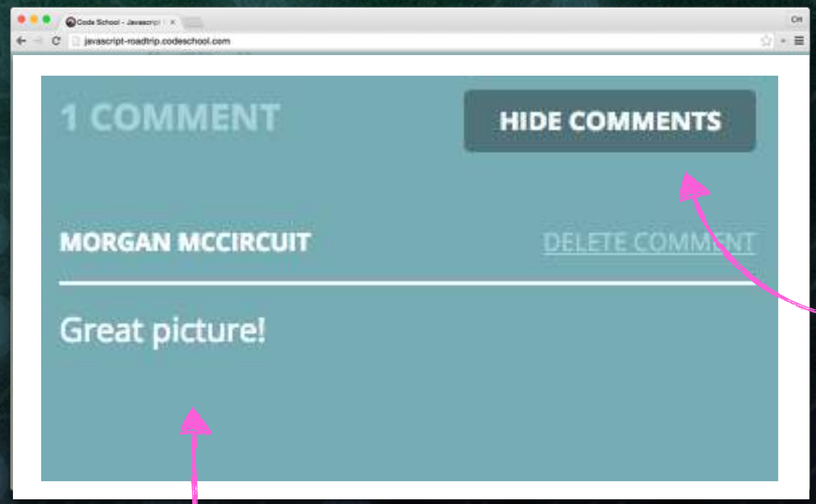
Calling *setState* **will only update** the properties passed as an argument, not replace the entire *state* object.

# Causing State Change

State changes are usually triggered by user interactions with our app.

## Things that could cause state change:

- Button clicks
- Link clicks
- Form submissions
- AJAX requests
- And more!




Button clicks can cause a change of state.

Loading comments from a remote server can also cause a change of state.

# Handling Click Events

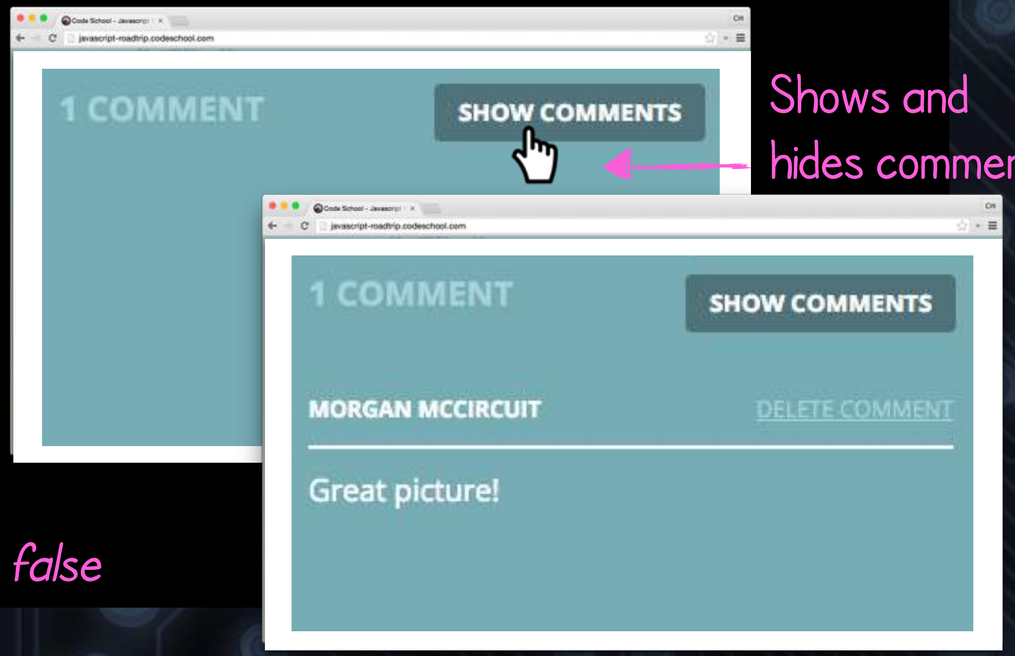
Let's add a button that will toggle the *showComments* state when a click event is fired.

```
class CommentBox extends React.Component { ...
  render() {
    ...
    return(
      ...
        <button onClick={this._handleClick.bind(this)}>Show comments</button>
      ...
    );
  }

  _handleClick() {
    this.setState({
      showComments: !this.state.showComments
    });
  }
}
```

Button that will toggle state on click event

Button click calls _handleClick()

Toggles state of showComments between *true* and *false*




Shows and hides comments

# Button Text Logic Based on State

We can switch the button text based on the component's state.

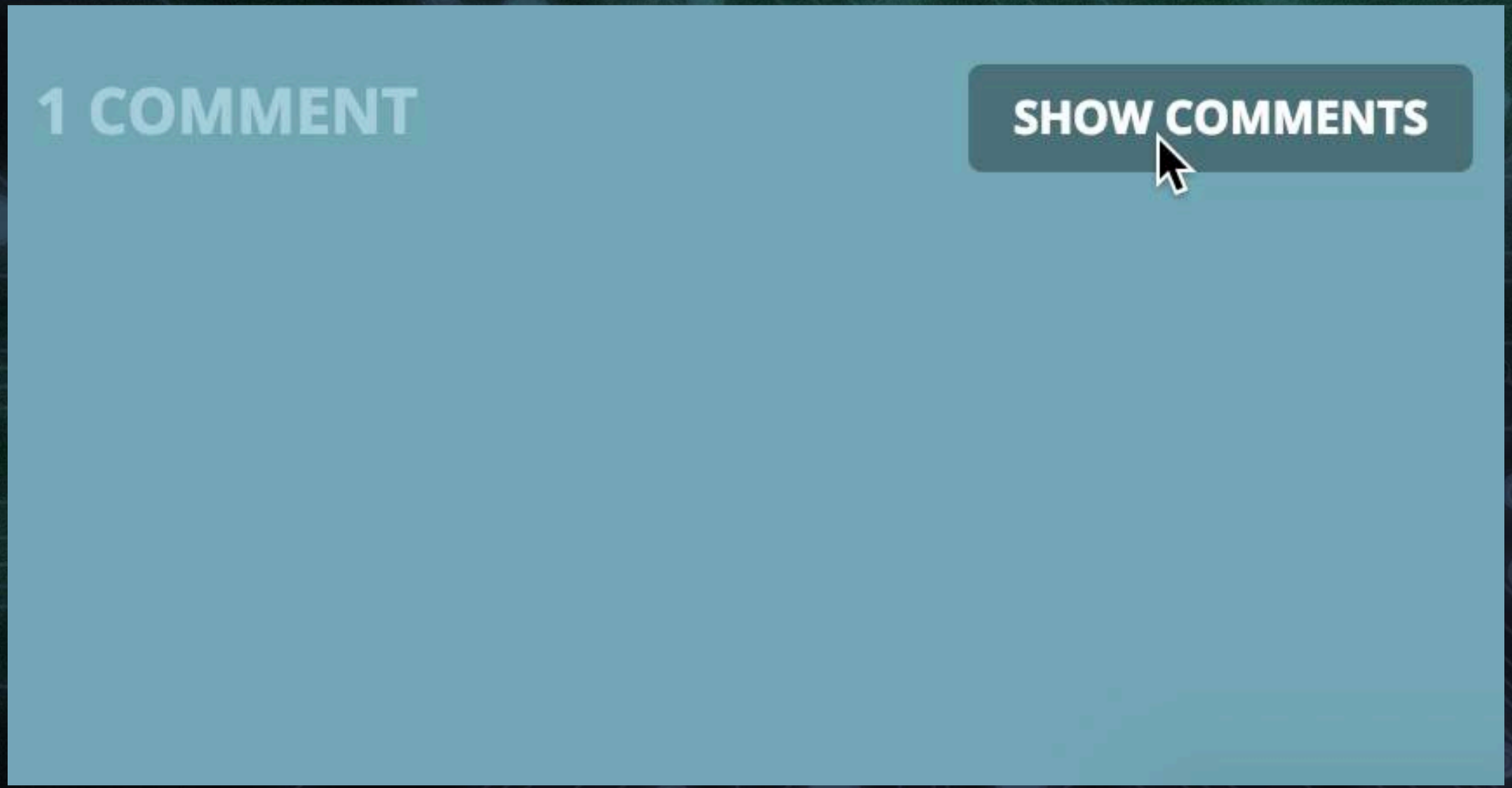
```
class CommentBox extends React.Component { ...
  render() {
    ...
    let buttonText = 'Show comments';

    if (this.state.showComments) {
      buttonText = 'Hide comments';
      ...
    }

    return(
      ...
        <button onClick={this._handleClick.bind(this)}>{buttonText}</button>
      ...
    );
  }
  ...
}
```

Switch button text based on current state

Renders button with according text

# Demo: Hide and Show Comments

Our app shows and hides comments when the button is clicked.



# Quick Recap on State

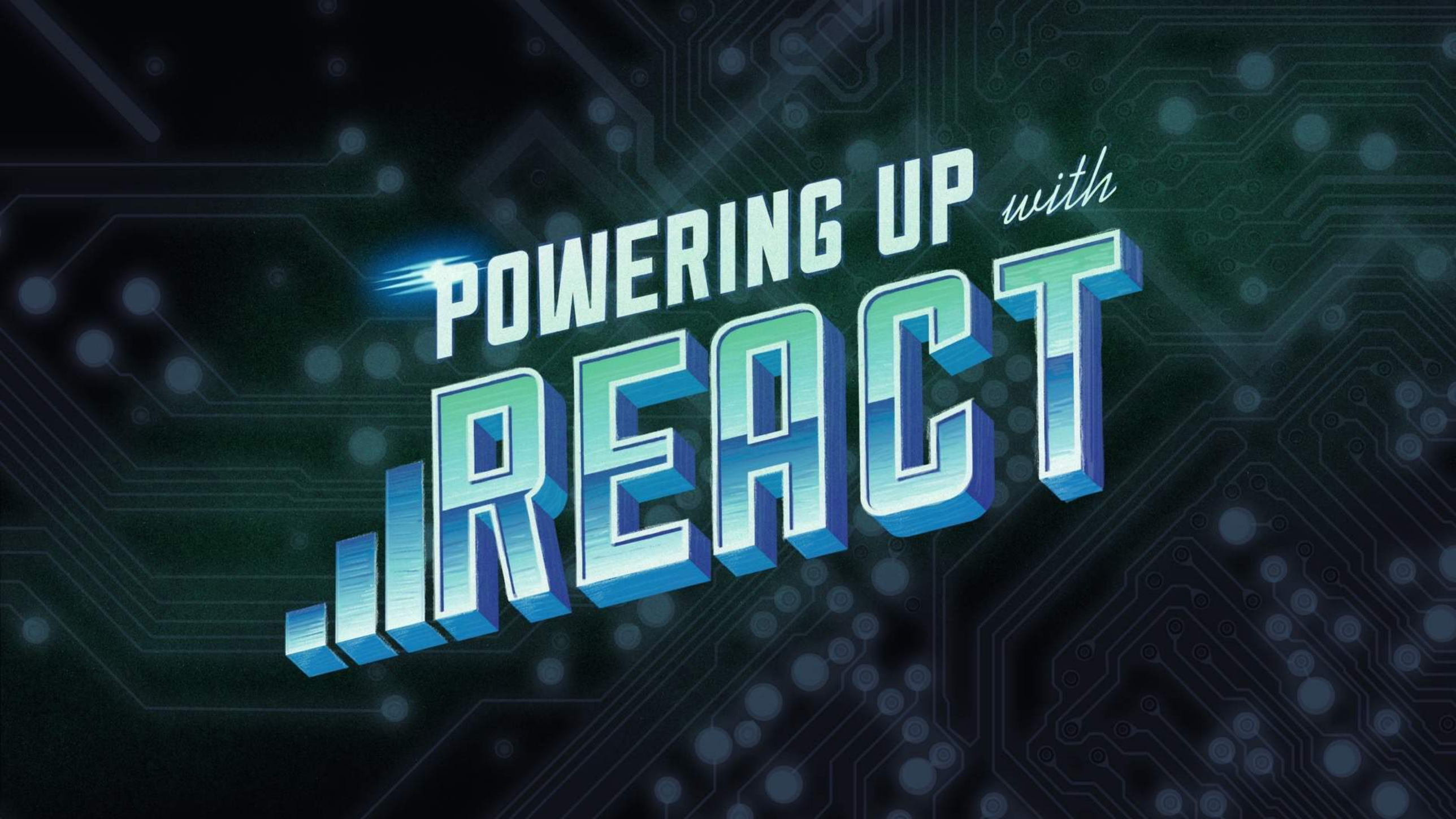
The state is a vital part of React apps, making user interfaces interactive.

State represents data that changes over time.

We update state by calling *this.setState()*.

We declare an **initial state** in the component's constructor.

Calling *this.setState()* causes our component to re-render.


POWERING UP with
REACT