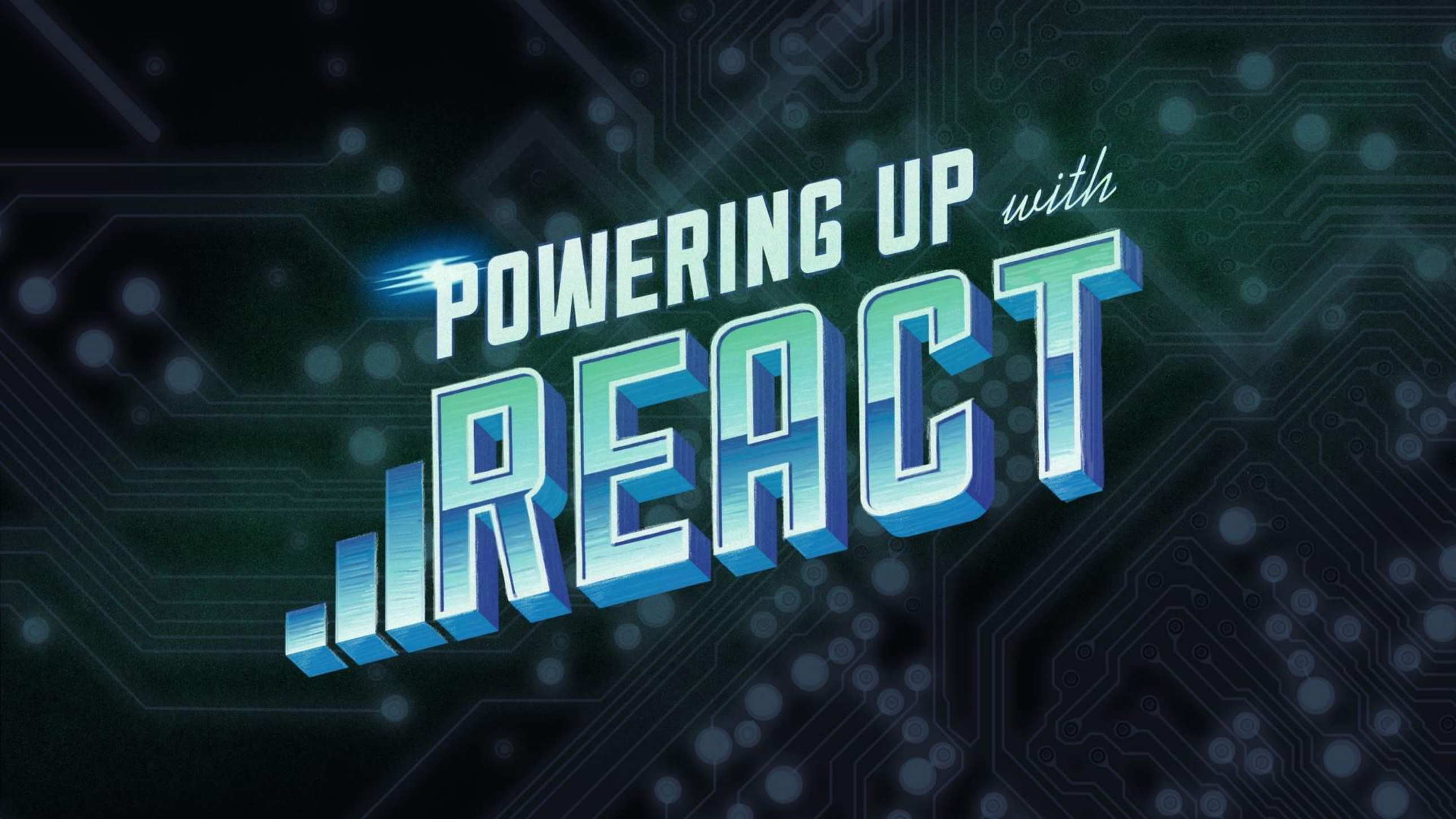

POWERING UP with
REACT

Level 2 – Section 2

Talk Through Props

Passing Dynamic Arguments

Problem: Props Aren't Dynamic Yet

We are passing literal strings as props, but what if we wanted to traverse an array of objects?

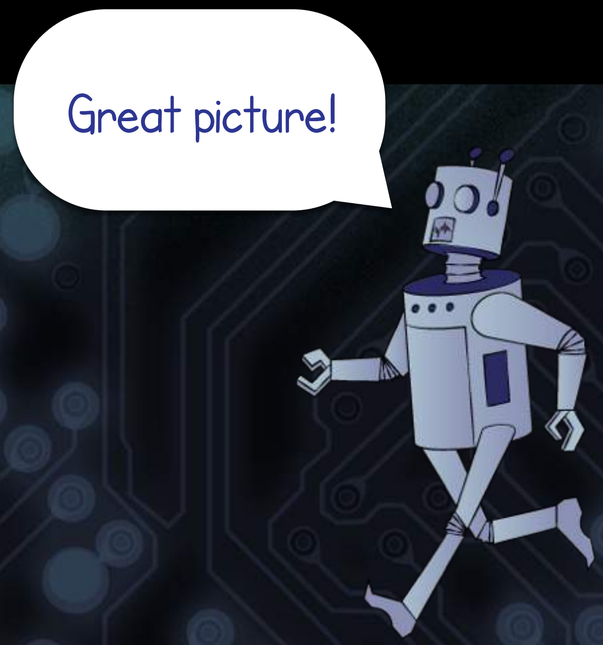
```
class CommentBox extends React.Component {
  render() {
    return(
      <div className="comment-box">
        <h3>Comments</h3>
        <h4 className="comment-count">2 comments</h4>
        <div className="comment-list">
          <Comment
            author="Morgan McCircuit" body="Great picture!" />
          <Comment
            author="Bending Bender" body="Excellent stuff" />
        </div>
      </div>
    );
  }
}
```

JavaScript Object Arrays

Typically, when we consume data from API servers, we are returned object arrays.

JavaScript

```
const commentList = [
  { id: 1, author: 'Morgan McCircuit', body: 'Great picture!' },
  { id: 2, author: 'Bending Bender', body: 'Excellent stuff' }
];
```



Mapping an Array to JSX

We can use JavaScript's *map* function to create an array with *Comment* components.

Underscore helps distinguish custom methods from React methods

```
class CommentBox extends React.Component {
  ...
  _getComments() {

    const commentList = [
      { id: 1, author: 'Morgan McCircuit', body: 'Great picture!' },
      { id: 2, author: 'Bending Bender', body: 'Excellent stuff' }
    ];

    return commentList.map(() => {
      return (<Comment                />);
    });
  }

}
```

New method that will return array of JSX elements

Returns an array...

...with a new component built for each element present in commentList.

Passing Dynamic Props

The callback to *map* takes an argument that represents each element from the calling object.

```
class CommentBox extends React.Component {
  ...
  _getComments() {

    const commentList = [
      { id: 1, author: 'Morgan McCircuit', body: 'Great picture!' },
      { id: 2, author: 'Bending Bender', body: 'Excellent stuff' }
    ];

    return commentList.map((comment) => {
      return (
        <Comment
          author={comment.author} body={comment.body} />
      );
    });
  }
}
```

Each element from commentList is passed as argument...

...which we can use to access properties and pass them as props.

Using Unique Keys on List of Components

Specifying a **unique key** when creating multiple components of the same type can **help improve performance**.

```
class CommentBox extends React.Component {
  ...
  _getComments() {

    const commentList = [
      { id: 1, author: 'Morgan McCircuit', body: 'Great picture!' },
      { id: 2, author: 'Bending Bender', body: 'Excellent stuff' }
    ];

    return commentList.map((comment) => {
      return (
        <Comment
          author={comment.author} body={comment.body} key={comment.id} />
      );
    });
  }
}
```

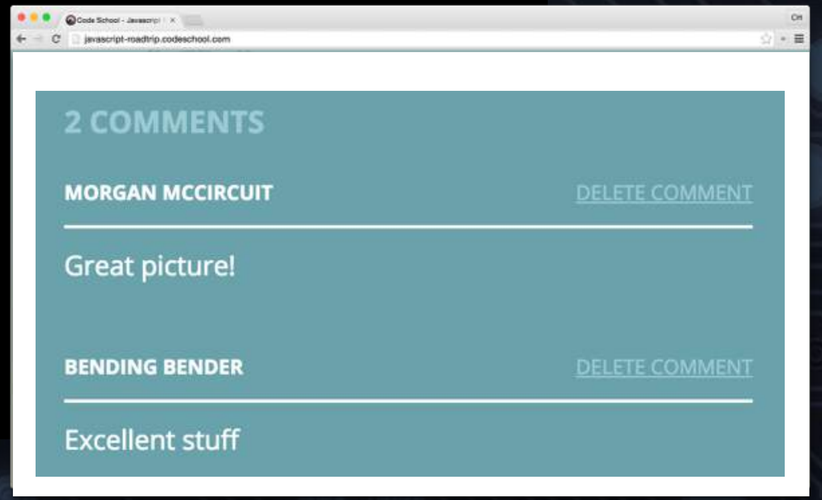
Unique key

Using the *_getComments()* method

We'll store the returned value in a variable named *comments* and use it for display purposes.

```
class CommentBox extends React.Component {
  render() {
    const comments = this._getComments();
    return(
      <div className="comment-box">
        <h3>Comments</h3>
        <h4 className="comment-count">{comments.length} comments</h4>
        <div className="comment-list">
          {comments}
        </div>
      </div>
    );
  }

  _getComments() { ... }
}
```

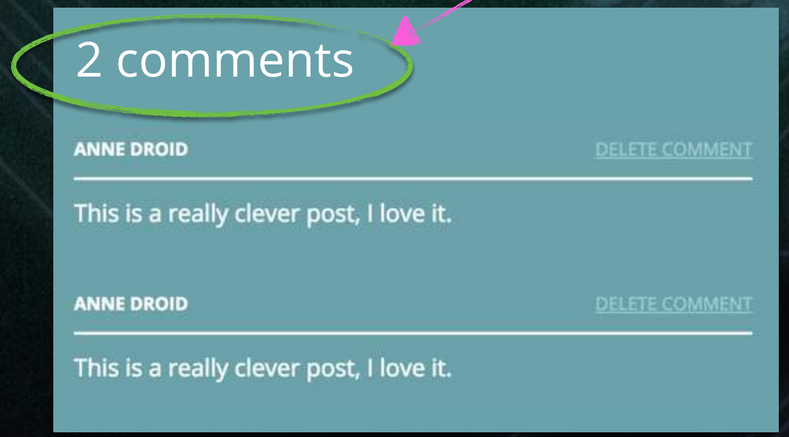
JSX knows how to render arrays



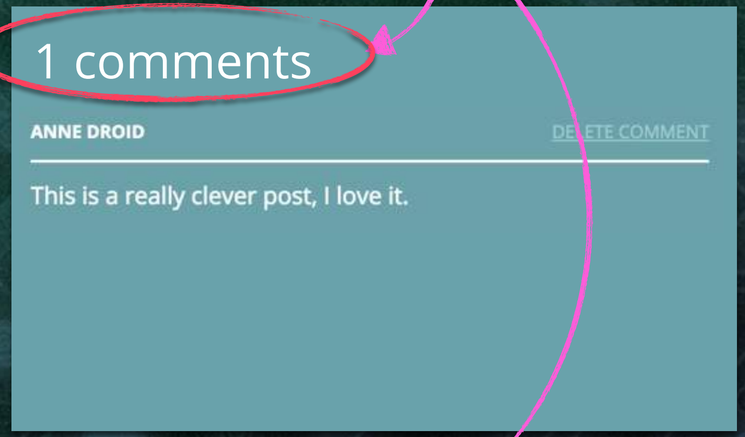
Incorrect Grammar on the Comments Title

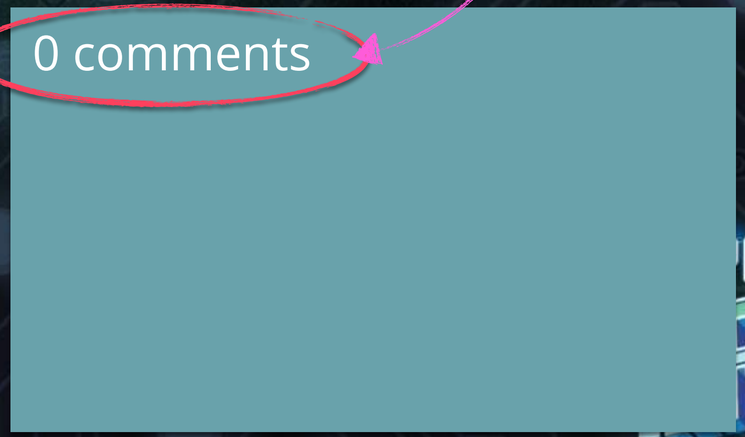
The title has incorrect grammar in some cases.

This is correct...



...but this is wrong!





Fixing the Title With Comment Count

Let's write a new method called *_getCommentsTitle()* that handles the plural case in our title.

```
class CommentBox extends React.Component {
  ...
  _getCommentsTitle(commentCount) {
    if (commentCount === 0) {
      return 'No comments yet';
    } else if (commentCount === 1) {
      return '1 comment';
    } else {
      return `${commentCount} comments`;
    }
  }
```

Uses same convention with starting underscore

Getting the Correct Comments Title

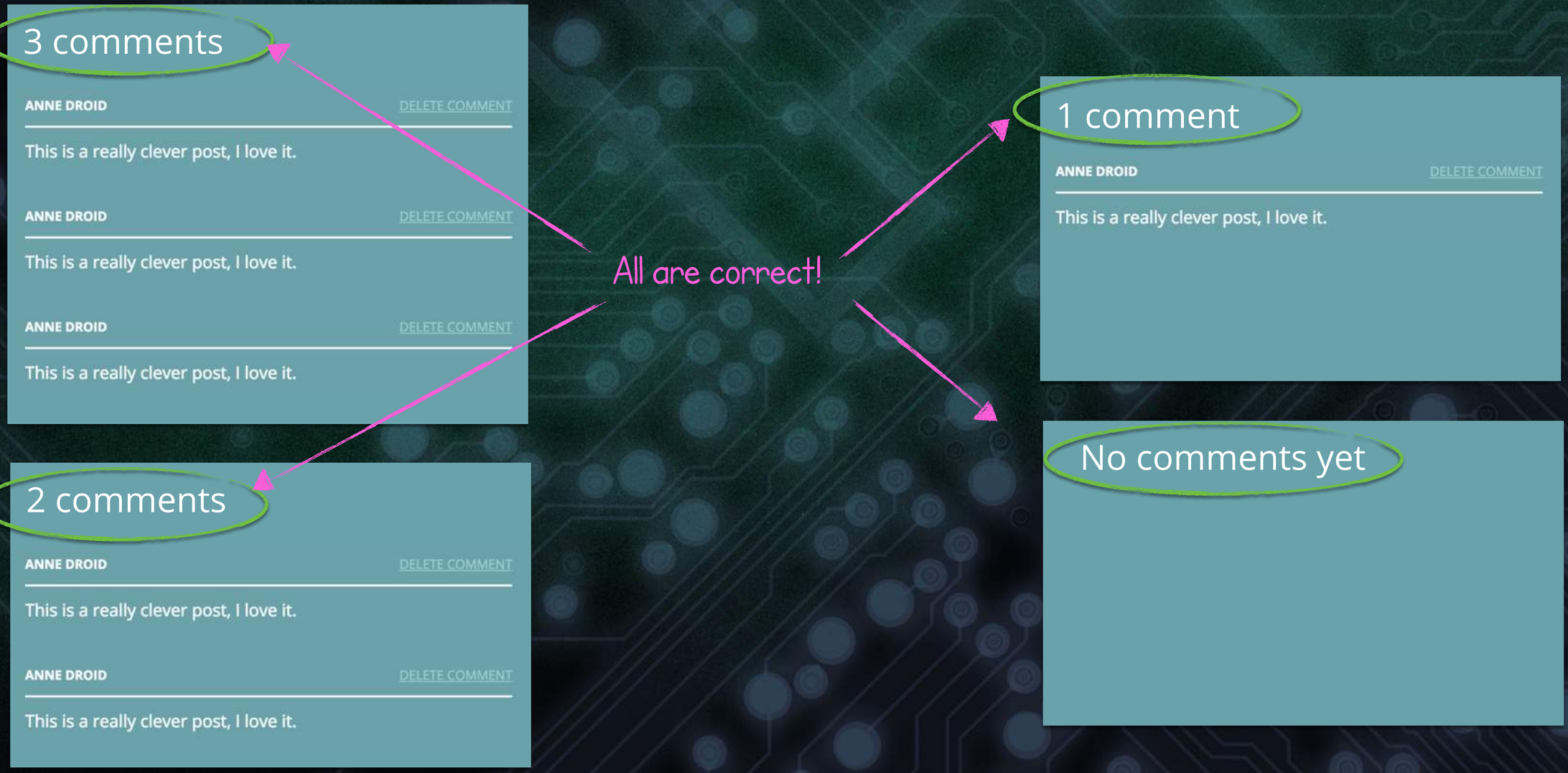
Let's call the method we just created from our component's *render* function.

```
class CommentBox extends React.Component {
  render() {
    const comments = this._getComments();
    return(
        ...
        <h4 className="comment-count">
          {this._getCommentsTitle(comments.length)}
        </h4>
        ...
    );
  }

  _getCommentsTitle(commentCount) { ... }

  ...
}
```

Get proper title for our component

Title Issue Is Fixed

The title now handles different quantities of comments accordingly.



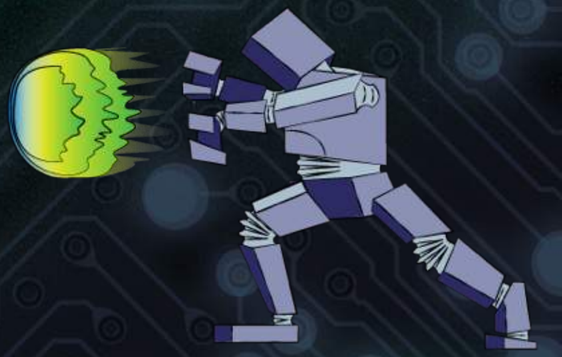
Quick Recap on Dynamic Props

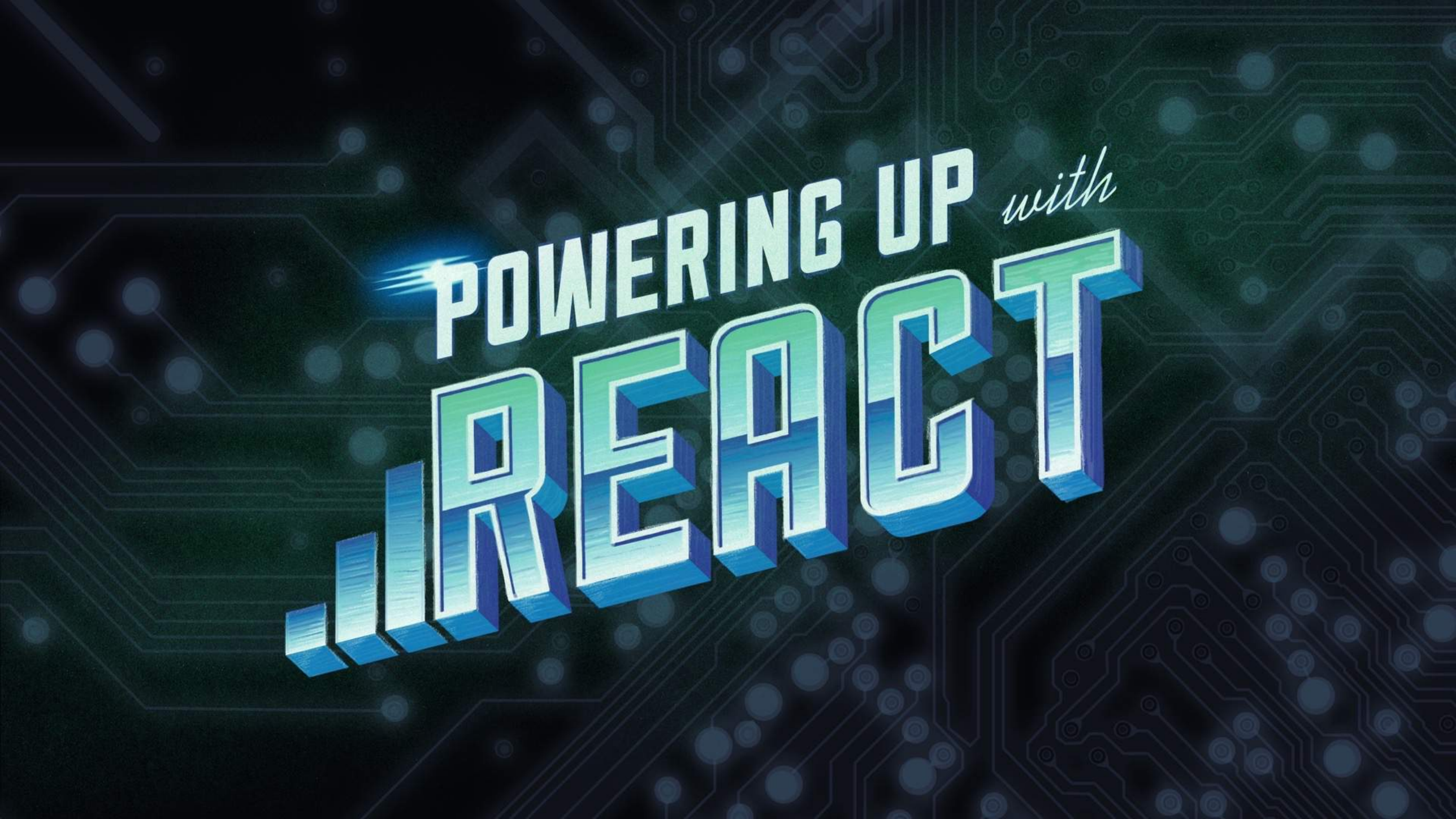
Dynamic props can be a bit mind boggling. Here's a summary of what we learned.

How to pass dynamic props using variables

Used JavaScript to handle plural case on the title

How to map object arrays to JSX arrays for display purposes




POWERING UP with
REACT