QUIZ

Why can't you call this.setState from within the render method?

By the time a component renders, its state has already been set.

`this.setState` automatically calls `render`, so it would create an infinite loop.

👏 You got it!

You *can* call `this.setState` from within the render method.

Calling `this.setState` from `render` would override all `props`.

Which file or files should use `ReactDOM.render()`?

```
// Inner.js

import React from 'react';

export class Inner extends React.Component {
  render() {
    return <p>How's it going?</p>;
  }
}

// Outer.js

import React from 'react';
import { Inner } from './Inner';

class Outer extends React.Component {
  render() {
    return (
      <div>
        <h1>Hello world</h1>
        <Inner />
      </div>
    );
  }
}
```

Inner.js

Outer.js

👏 You got it!

What does `this.props.children` return?

- Everything that gets passed a prop in between a component's opening and closing tags.
- Everything that gets passed a prop in a component's render method.
- Everything between a component's opening and closing tags.

You got it!

Fill in the blank, so that clicking on a button will capitalize the button's inner text.

```
import React from 'react';

class Button extends React.Component {
  constructor(props) {
    super(props);
    this.state = { message: 'Hello world' };
    this.capitalize = this.capitalize.bind(this);
  }

  capitalize() {
    this.setState({
      message: this.state.message.toUpperCase()
    });
  }

  render() {
    return (
      <button ____________________>
        {this.state.message}
      </button>
    );
  }
}
```

- onClick={this.setState.capitalize}
- onClick={capitalize}
- onClick={this.state.capitalize}
- onClick={this.capitalize}

You got it!

Importer.js and Exporter.js share a parent folder. In Importer.js, what code evaluates to `"Hello world"`?

```
// Exporter.js
export const myObj = {
  dang: "Hello world"
};
```

```
import { myObj }.dang;
```

```
import { myObj } from './Exporter';
myObj;
```

```
import { myObj } from './Exporter';
myObj.dang;
```

Which file or files needs to use an export statement?

```
// Inner.js

import React from 'react';

class Inner extends React.Component {
  render() {
    return <p>How's it going?</p>;
  }
}

// Outer.js

import React from 'react';
import { Inner } from './Inner';

class Outer extends React.Component {
  render() {
    return (
      <div>
        <h1>Hello world</h1>
        <Inner />
      </div>
    );
  }
}
```

Both Inner.js and Outer.js

Outer.js

Inner.js

You got it!

What could you put in Exporter.js to make `h1` equal `<h1>Hello world</h1>`?

```
// Importer.js
import { greet } from './Exporter';
const h1 = greet('Hello world');
```

```
export function greet(text) {
  return <h1>text</h1>;
}
```

```
export const greet = {
  'Hello World': <h1>Hello world</h1>
};
```

```
export function greet(text) {
  return <h1>{text}</h1>;
}
```

👏 You got it!

How would you get an `<Animal />` to render `<h1>cat</h1>`?

```
import React from 'react';

class Animal extends React.Component {
  render() {
    if (this.props.type == 'cat') {
      return <h1>Meow Meow</h1>;
    } else if (this.props.type == 'dog') {
      return <h1>Arf Arf</h1>;
    }
  }
}
```

```
<Animal type="dog" />
```

You can't.

👏 You got it!

Which is the correct way to pass a prop?

```
<Headline "text"={this.props.text} />
```

```
<Headline>.prop.text = "Hello world"
```

```
<Headline text="Hello world" />
```

👏 You got it!

`this.props` evaluates to what data type?

Number

Function

Object

👏 You got it!

Array

What's wrong with this way of setting an initial state?

```
constructor(props) {
  super(props);
  this.state = 'Hello world';
}
```

Nothing.

`this.state` should be equal to an object.

👏 You got it!