POWERING UP *with*

REACT

Level 1 – Section 2

First Component

Understanding JSX

POWERING UP with REACT

No Quotes Around Markup

The markup we use when writing React apps is not a string. This markup is called **JSX** (JavaScript XML).

```
class StoryBox extends React.Component {
  render() {
    return( <div>Story Box</div> );
  }
}

ReactDOM.render(
  <StoryBox />, document.getElementById('story-app')
);
```

HTML elements are written in lowercase.

React components are written in upper camel case.

POWERING UP with REACT

A New Way to Write JavaScript

JSX is just another way of writing JavaScript with a transpile step.

```
class StoryBox extends React.Component {
  render() {
    return( <div>Story Box</div> );
  }
}

ReactDOM.render(
  <StoryBox />, document.getElementById('story-app')
);
```

This JSX becomes…

Transpiled JSX Code

```
React.createElement('div', null, 'Story Box')
```

This JSX becomes…

Transpiled JSX Code

```
React.createElement(StoryBox, null)
```

POWERING UP with REACT

Getting Used to the JSX Syntax

JSX looks similar to HTML, and it is ultimately transformed into JavaScript.

```
class StoryBox extends React.Component {
  render() {
    return(
      <div>
        <h3>Stories App</h3>
        <p className="lead">Sample paragraph</p>
      </div>
    );
  }
}
```

Notice we are using className and not class,
which is a JavaScript-reserved keyword.

Resulting JavaScript code

Transpiled JSX code

```
React.createElement("div", null,
  React.createElement("h3", null, "Stories App"),
  React.createElement("p", {className:"lead"}, "Sample paragraph")
);
```
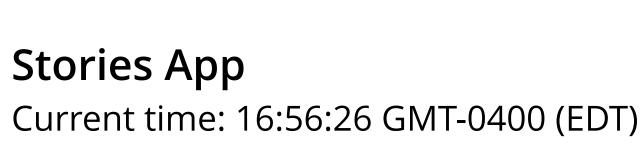
From JSX to HTML

JSX

```
...
  <div>
    <h3>Stories App</h3>
    <p className="lead">Sample paragraph</p>
  </div>
...
```

All JSX gets transformed to JavaScript.

JavaScript

```
React.createElement("div", null,
  React.createElement("h3", null, "Stories App"),
  React.createElement("p", {className:"lead"}, "Sample paragraph")
);
```

Rendered by the browser

Stories App
Sample paragraph

Elements   Console   Network   Timeline

```
<div data-reactroot>
  <div>
    <h3>Stories App</h3>
    <p class="lead">Sample paragraph</p>
  </div>
</div>
```

Generated HTML

Using the *Date* Object in JSX

Here, we're displaying the current time using JavaScript's native *Date* object and JSX.

```jsx
class StoryBox extends React.Component {
  render() {

    const now = new Date();

    return (
      <div>
        <h3>Stories</h3>
        <p className="lead">
          Current time: {now.toTimeString()}
        </p>
      </div>
    );
  }
}
```

Stories App
Current time: 16:56:26 GMT-0400 (EDT)

Code written within curly braces gets interpreted as literal JavaScript.

Iterating Arrays in JSX

Here, we're displaying a list of elements using JSX and JavaScript's native *map* function.

```
class StoryBox extends React.Component {
  render() {

    ...
    const topicsList = ['HTML', 'JavaScript', 'React'];


    return (
      <div>

        ...
        <ul>

          {topicsList.map( topic => <li>{topic}</li> )}
        </ul>
      </div>
    );
  }
}
```

Stories App
Current time: 16:56:26 GMT-0400 (EDT)

· HTML
· JavaScript
· React

```
<li>HTML</li>
<li>JavaScript</li>
<li>React</li>
```

This function returns this JSX array.

Quick Recap on JSX

JSX stands for **JavaScript XML.**

JSX markup looks similar to HTML, but ultimately gets transpiled to **JavaScript function calls,** which React will know how to render to the page.

Code written within curly braces is interpreted as literal JavaScript.

It is a common pattern to map arrays to JSX elements.

POWERING UP with REACT