Complex JavaScript in JSX

Let's see how to render lists and objects in your JSX!

So far, you have rendered a few primitive variables in your JSX. Now, we will render a list of items. The list will contain sample data in the beginning, but later we will learn how to fetch the data from an external API.

First, you have to define the list of items:

```
import React, { Component } from 'react';
import './App.css';

const list = [
  {
    title: 'React',
    url: 'https://reactjs.org/',
    author: 'Jordan Walke',
    num_comments: 3,
    points: 4,
    objectID: 0,
  },
  {
    title: 'Redux',
    url: 'https://redux.js.org/',
    author: 'Dan Abramov, Andrew Clark',
    num_comments: 2,
    points: 5,
    objectID: 1,
  },
];

class App extends Component {
  ...
}
```

The sample data represents information we will fetch from an API later on. Items in this list each have a title, a URL, and an author, as well an identifier, points (which indicate how popular an article is), and a count of comments.

Now you can use the built-in JavaScript map functionality in JSX, which iterates over a list of items to display them according to specific attributes. Again, we use curly braces to encapsulate the JavaScript expression in JSX:

```
import React, { Component } from 'react';
require('./App.css');

const list = [
  {
    title: 'React',
    url: 'https://reactjs.org/',
    author: 'Jordan Walke',
    num_comments: 3,
    points: 4,
    objectID: 0,
  },
  {
    title: 'Redux',
    url: 'https://redux.js.org/',
    author: 'Dan Abramov, Andrew Clark',
    num_comments: 2,
    points: 5,
    objectID: 1,
  },
];

class App extends Component {

  render() {
    return (
      <div className="App">
        {list.map(function(item) {
          return <div>{item.title}</div>;
        })}
      </div>
    );
  }
}

export default App;
```

Using JavaScript alongside HTML in JSX is very powerful. For a different task you may have used `map` to convert one list of items to another. This time, we used `map` to convert a list of items to HTML elements.

```
const array = [1, 4, 9, 16];

// pass a function to map
const newArray = array.map(function (x) { return x * 2; });

console.log(newArray);
// expected output: Array [2, 8, 18, 32]
```

So far, only the `title` is displayed for each item. Let's experiment with more of the item's properties:

```
import React, { Component } from 'react';
require('./App.css');

const list = [
  {
    title: 'React',
    url: 'https://reactjs.org/',
    author: 'Jordan Walke',
    num_comments: 3,
    points: 4,
    objectID: 0,
  },
  {
    title: 'Redux',
    url: 'https://redux.js.org/',
    author: 'Dan Abramov, Andrew Clark',
    num_comments: 2,
    points: 5,
    objectID: 1,
  },
];

class App extends Component {
  render() {
    return (
      <div className="App">
        {list.map(function(item) {
          return (
            <div>
              <span>
                <a href={item.url}>{item.title}</a>
              </span>
              <span>{item.author}</span>
              <span>{item.num_comments}</span>
              <span>{item.points}</span>
            </div>
          );
        })}
      </div>
    );
  }
}

export default App;
```

Note how the `map` function is inlined in your JSX. Each item property is displayed with a `<span>` tag, and the url property of the item is in the `href` attribute of the anchor tag.

React will display each item, but you can still do more to help React embrace its full potential. By assigning a key attribute to each list element, React can identify modified items when the list changes. These sample list items come with an identifier:

with an identifier.

```
{list.map(function(item) {
  return (
    <div key={item.objectID}>
      <span>
        <a href={item.url}>{item.title}</a>
      </span>
      <span>{item.author}</span>
      <span>{item.num_comments}</span>
      <span>{item.points}</span>
    </div>
  );
})}
```

Make sure that the key attribute is a stable identifier. Avoid using the index of the item in the array, because the array index is not stable. If the list changes its order, for example, React will not be able to identify the items properly.

```
// don't do this
{list.map(function(item, key) {
  return (
    <div key={key}>
      ...
    </div>
  );
})}
```

```
import React, { Component } from 'react';
require('./App.css');

const list = [
  {
    title: 'React',
    url: 'https://reactjs.org/',
    author: 'Jordan Walke',
    num_comments: 3,
    points: 4,
    objectID: 0,
  },
  {
    title: 'Redux',
    url: 'https://redux.js.org/',
    author: 'Dan Abramov, Andrew Clark',
    num_comments: 2,
    points: 5,
    objectID: 1,
  },
];

class App extends Component {
  render() {
    return (
      <div className="App">
```

```
        {list.map(function(item) {
          return (
                                    <div key={item.objectID}>

              <span>
                <a href={item.url}>{item.title}</a>
              </span>
              <span>{item.author}</span>
              <span>{item.num_comments}</span>
              <span>{item.points}</span>
            </div>
          );
        })}
      </div>
    );
  }
}

export default App;
```

Now you should see both items of the list displayed.

Exercises:

- Use more JavaScript expressions on your own in JSX

Further Readings:

- Read about React lists and keys
- Recap the standard built-in array functionalities in JavaScript