ES6 Arrow Functions

An introduction to Arrow Functions and their importance.

JavaScript ES6 introduced arrow functions expressions, which are shorter than a function expressions.

```
// function declaration
function () { ... }

// arrow function declaration
() => { ... }
```

You can remove the parentheses in an arrow function expression if it only has one argument, but you have to keep the parentheses if it gets multiple arguments:

```
// allowed
item => { ... }

// allowed
(item) => { ... }

// not allowed
item, key => { ... }

// allowed
(item, key) => { ... }
```

You can also write `map` functions more concisely with an ES6 arrow function:

```
{list.map(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>
  );
```

```
))}
```

You can remove the *block body*, the curly braces, with the ES6 arrow function. In a *concise body*, an implicit return is attached; thus, you can remove the `return` statement. This will happen often in this course, so be sure to understand the difference between a block body and a concise body when using arrow functions.

```
{list.map(item =>
  <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>
)}
```

Q

What would the following code do?

```
render(){
   let users = [
  { 'name' : 'Elton Jhon', 'age' : 60 },
  { 'name' : 'Elvis Persley', 'age' : 50 },
  { 'name' : 'Kurt Cobain', 'age' : 24}
];
   return (<div>
     {users.map(user => <p>user.name</p>)}
   </div>)
}
```

Your JSX should look more concise and readable now, as it omits the `function` statement, the curly braces, and the return statement.

Further Reading:

- Read about ES6 arrow functions