Audit Access Expires Apr 20, 2020
You lose all access to this course, including your progress, on Apr 20, 2020.

Lists

Rendering arrays of React Elements

JSX will render an array of React Elements as long as there is at least one element enclosing all of the array elements. The array elements will be inserted into the enclosing element:

For example, it is possible to render multiple components at the same time using a for loop with JSX:

```
var elements = []
var array = [1,2,3,4,5]

for(let i = 0; i < array.length; i++){
   elements.push(<li>{array[i]}<li/>)
}


ReactDOM.render(
  <ol>{elements}</ol>,
  document.getElementById('root')
)
```

Using Map() to render arrays of React Elements

The map() method is often used to create an array of React Elements. The map() method is called on an array and returns a new array with a provided function applied to each element in the original array.

Example of using the map() method to return an array of React Elements:

```
var array =[
  {product:"Apple", price:3},
  {product:"Banana", price:1},
  {product:"Carrot", price:2},
  {product:"Donuts", price:5},
  {product:"Eggplant", price:4}
]

var elements = array.map( (item) => {
  return <li>Product: {item.product} | Price: ${item.price}  </li>
})

ReactDOM.render(
  <ol>{elements}</ol>,
  document.getElementById('root')
)
```

The map() method can also be directly used inside a JSX expression:

```
ReactDOM.render(
  <ol>{
      array.map( (item) =>
          <li>Product: {item.product} | Price: ${item.price} </li>
      )}
  </ol>,
  document.getElementById('root')
)
```

Adding Keys to List Items

React uses Keys to help render list items quickly. Keys should be a string that uniquely identifies a list item from the other items on the list, such as an ID attribute.

Example of using an ID as a key value:

```
var array =[
  {id: 100, product:"Apple", price:3},
  {id: 101, product:"Banana", price:1},
  {id: 102, product:"Carrot", price:2},
  {id: 103, product:"Donuts", price:5},
  {id: 104, product:"Eggplant", price:4}
]

var elements = array.map( (item) => {
  return <li key={item.id}>Product: {item.product} | Price:
${item.price}  </li>
})

ReactDOM.render(
  <ol>{elements}</ol>,
  document.getElementById('root')
)
```

If your array items do not have anything that can uniquely identify them, you can use the item index as a last resort for the key value. The drawback to using indexes as keys is that list item reordering is slow to rerender.

Example of using the item index as a key value:

```
var array =[
  {product:"Apple", price:3},
  {product:"Banana", price:1},
  {product:"Carrot", price:2},
  {product:"Donuts", price:5},
  {product:"Eggplant", price:4}
]

//the item index is the second argument to the map() method
var elements = array.map( (item,index) => {
  return <li key={index}>Product: {item.product} | Price:
${item.price}  </li>>
})

ReactDOM.render(
  <ol>{elements}</ol>,
  document.getElementById('root')
)
```

Building a List Component

It is useful to be able to build a React Component that can dynamically generate a list from an array property that is passed into it.

```
class ProductList extends React.Component{
  constructor(props){
    super(props)
  }
  render(){
    var elements = this.props.productArray.map( (item,index) => {
      return <li key={item.id}>Product: {item.product} | Price:
${item.price}  </li>
    })
    return <ol>{elements}</ol>
  }
}

var array =[
  {id: 100, product:"Apple", price:3},
  {id: 101, product:"Banana", price:1},
  {id: 102, product:"Carrot", price:2},
  {id: 103, product:"Donuts", price:5},
  {id: 104, product:"Eggplant", price:4}
]


ReactDOM.render(
  <ProductList productArray = {array}/>,
  document.getElementById('root')
)
```

Extracting List Items

Each list item may be extracted into its own React Component to make the code more maintainable. If the list items are extracted, the keys do not need to be passed down to the list item components. Keys are only necessary when React Elements are generated dynamically using arrays.

Example:

```
function ListItem(props){
  //don't need to add a key to
  return <li>Product: {props.product} | Price: ${props.price}  </li>

}


class ProductList extends React.Component{
  render(){
    var elements = array.map( (item,index) => {
      //need to add a key here
      return <ListItem key={item.id} product={item.product} price =
{item.price}/>
    })

    return (
      <ol>
        {elements}
      </ol>
    )

  }

}

var array =[
  {id: 100, product:"Apple", price:3},
  {id: 101, product:"Banana", price:1},
  {id: 102, product:"Carrot", price:2},
  {id: 103, product:"Donuts", price:5},
  {id: 104, product:"Eggplant", price:4}
]

ReactDOM.render(
  <ProductList productArray = {array}/>,
  document.getElementById('root')
)
```