

Working with a List

Wednesday, March 23, 2022 4:38 PM


Working with list challenges

- ▶ **JSX interpolation syntax `{ }` can work with**
 - ▶ Simple Expressions
 - ▶ Function calls
- ▶ **JSX interpolation syntax `{ }` can't include statements like**
 - ▶ For loop
 - ▶ If Statements

```
<div> { 2 + 2 } </div>

<div> { getDate() } </div>

<div> {
  for( let x : values)
    if(x)
      doSomething()
}
</div>
```



We Render a List in React Using array.map function

- We map each item of the array with a JSX Element

```
{ values.map ( value => <p>{value}</p> ) }
```

- Now we get one JSX per value in the list.
- If the value rendered is a complex item it is recommended to delegate the details to separate component and render that component

```
{ books.map ( book => <BookListInfo book={book} /> ) }
```

```
const BookListInfo = ( {book} ) =>{
```

```
  return (
    <div>
      {book.title}
      ...
    </div>
  );
```

```
}
```

Key 'prop'

- Whenever we render a List with several items, React needs to identify each item uniquely.
- This helps react modify the list in an efficient manner when we edit or delete list items
 - It can re-render only the item that has changed
- If key prop is missing, react gives a warning
- It can still edit or delete the item, but to do that
 - It re-renders entire list.
 - It becomes in-efficient
- Key prop should be specified to the item generate in the map function
- Key should be a unique value
 - Most business object has some kind of id
 - Book has isbn
- Key prop is not accessible in the Child component.

- It is only for react's internal use
- If you need that info for your own purpose add it to some other props

With missing key prop

The screenshot shows a web browser at localhost:3001 displaying a book list. The first item is 'Harry Potter and the Philosopher's Stone' by JK Rowling. Below the browser, the VS Code editor shows the `BookListScreen.js` file. The code defines a `BookListScreen` component that renders a list of books. The `books.map` function call does not include a `key` prop for the `BookListItem` component.

```
const BookListScreen = () => {
  // TODO: Initialize Here

  let books = [
    // ...
  ];

  return (
    <div className="BookListScreen">
      <h1>Book List Screen</h1>
      <div className="list">
        {books.map( book => <BookListItem book={book} /> )}
      </div>
    </div>
  );
};

export default BookListScreen;
```

The browser's console shows a warning: "Warning: Each child in a list should have a unique 'key' prop." The warning points to the `BookListItem` component in the `BookListScreen` component.

- It is a warning not an error
- IT SHOULD BE TAKEN SERIOUSLY

- Items generated using `map()` should have a key prop

Find a Unique Key and Supply

The screenshot shows the same web browser and VS Code editor. In the VS Code editor, the `BookListScreen.js` file is updated to include a `key` prop for the `BookListItem` component. The `key` prop is set to `book.isbn`, which is a unique value for each book.

```
const BookListScreen = () => {
  // TODO: Initialize Here

  let books = [
    // ...
  ];

  return (
    <div className="BookListScreen">
      <h1>Book List Screen</h1>
      <div className="list">
        {books.map( book => <BookListItem key={book.isbn} book={book} /> )}
      </div>
    </div>
  );
};

export default BookListScreen;
```

The browser's console now shows no warnings, indicating that the `key` prop has been successfully added to the list items.

Remember

- This key is not accessible as props of `BookListItem`

What If you don't have a unique key in your list item

- It is highly unlikely in any business scenario
 - You should have a key
- We may use some unique value to generate a key
 - Common value used
 - Index of the current item
 - Available as second parameter in `map` function call back
 - Random number generated from a large pull
 - Current date in millisecond
 - Most of these can suppress the warning but may not be very useful from app perspective
 - You can use "index" if you are sure you are not going to delete the content and index will not change.

