How to create a React Table Component

MAY 17, 2021 BY ROBIN WIERUCH - EDIT THIS POST

In this tutorial, I want to show you how to use React Table Library for **creating a table component in React**. After this tutorial, there will be many other examples which you can continue to build, such as searching, sorting, selecting, or pagination features for your React table by using the library's documentation. But let's start with the basics.

Let's begin by installing React Table Library via your command line:

```
npm install @table-library/react-table-library
```

The task is to present the following list of items in a React table component:

```
const list = [
```

```
  {
    id: '1',
    name: 'VSCode',
    deadline: new Date(2020, 1, 17),
    type: 'SETUP',
    isComplete: true,
  },
  {
    id: '2',
    name: 'JavaScript',
    deadline: new Date(2020, 2, 28),
    type: 'LEARN',
    isComplete: true,
  },
  {
    id: '3',
    name: 'React',
    deadline: new Date(2020, 3, 8),
    type: 'LEARN',
    isComplete: false,
  }
];
```



We will start with structuring the list in an object which can be consumed by the Table component. The component itself gets imported from the library:

```
import * as React from 'react';
import { Table } from '@table-library/react-table-library/table';

const list = [ ... ];

const App = () => {
  const data = { nodes: list };

  return <Table data={data}>{(tableList) => null}</Table>;
};
```

The Table component accepts a data object as prop with a `nodes` property. These nodes are the items in our lists, however, the table keeps it more generic to the naming of `nodes`, because the table can not only display list structures but also tree structures.

Moreover, the Table component uses a function as a children which gives us access to our list within the table as `tableList`. Internally the Table component applies all sorts of modifications onto our list -- e.g. sorting, pagination and so on, if these plugins are enabled -- and thus the `tableList` (and not `data` or `list`) should be used to render the items within the table.

React Table Library uses composition over configuration. Therefore, you get all the necessary building blocks as components from the library itself. Let's begin with the header of our table:

```
import * as React from 'react';
import {
  Table,
  Header,
  HeaderRow,
  HeaderCell,
} from '@table-library/react-table-library/table';

const list = [ ... ];

const App = () => {
  const data = { nodes: list };

  return (
    <Table data={data}>
      {(tableList) => (
        <Header>
          <HeaderRow>
            <HeaderCell>Task</HeaderCell>
            <HeaderCell>Deadline</HeaderCell>
            <HeaderCell>Type</HeaderCell>
            <HeaderCell>Complete</HeaderCell>
          </HeaderRow>
        </Header>
      )}
    </Table>
  );
};
```

By using these components, you can create a table as a composition from components whereas each component has its own responsibility. For example, instead of having only one Table component which accepts one large configuration object, we have composable components -- such as Header, HeaderRow, and HeaderCell, which can receive dedicated props.

Next, let's display our items like we are used to when rendering lists in React by rendering Row components with a key for each item within a Body component:

```
import * as React from 'react';
import {
  Table,
  Header,
  HeaderRow,
```

```
  HeaderCell,
  Body,
  Row,
  Cell,
} from '@table-library/react-table-library/table';

const list = [ ... ];

const App = () => {
  const data = { nodes: list };

  return (
    <Table data={data}>
      {(tableList) => (
        <>
          <Header>
            ...
          </Header>

          <Body>
            {tableList.map((item) => (
              <Row key={item.id} item={item}>
                {(tableItem) => (
                  <>
                    <Cell>{tableItem.name}</Cell>
                    <Cell>
                      {tableItem.deadline.toLocaleDateString(
                        'en-US',
                        {
                          year: 'numeric',
                          month: '2-digit',
                          day: '2-digit',
                        }
                      )}
                    </Cell>
                    <Cell>{tableItem.type}</Cell>
                    <Cell>{tableItem.isComplete.toString()}</Cell>
                  </>
                )}
              </Row>
            ))}
          </Body>
        </>
      )}
    </Table>
  );
};
```

The Row component is similar to the Table component, because it utilizes the children as a function feature from React as well. Therefore, you can access the `tableItem` to display its properties within the provided Cell components.

Important: Here again, do not use the outer `item`, because the `tableItem` has all the applied modifications which comes from plugins such as sorting, searching, and pagination.

Since you have full control over what to render in the Cell components, you can format the data as you need to. A boolean can be translated into a string and a date can be formatted to a readable version. There are no special props for the Cell components to get the rendering done.

. . .

Using React Table Library makes it straightforward to render table components in React. Head over to the library's documentation to find out more about it and its features.

Show Comments

KEEP READING ABOUT REACT›

REACT INTERNATIONALIZATION WITH I18N

When my last client asked me about internationalization in React, I went through all the hoops to prepare a presentation for them. In this React tutorial, I want to show you the gist of what I have…

HOW TO USE REACT TESTING LIBRARY TUTORIAL

React Testing Library (RTL) by Kent C. Dodds got released as alternative to Airbnb's Enzyme . While Enzyme gives React developers utilities to test internals of React components, React Testing…

THE ROAD TO REACT

Learn React by building real world applications. No setup configuration. No tooling. Plain React in 200+ pages of learning material. Learn React like **50.000+ readers**.

GET THE BOOK

Get it on Amazon.

TAKE PART

NEVER MISS AN ARTICLE ABOUT WEB DEVELOPMENT AND JAVASCRIPT.

- ✔ Join 50.000+ Developers
- ✔ Learn Web Development
- ✔ Learn JavaScript
- ✔ Access Tutorials, eBooks and Courses

✔ Access Tutorials, eBooks and Courses

✔ Personal Development as a Software Engineer

SUBSCRIBE >

View our Privacy Policy.



PORTFOLIO

Online Courses

Open Source

Tutorials

ABOUT

About me

What I use

How to work with me

How to support me