**How to make Reusable React Components ?**

Last Updated : 22 Mar, 2024

ReactJS is aJavaScript **library**used to build **single-page applications (SPAs)**. React makes it easy to create an interactive user interface by using a **component-based** approach. In this article, we will learn about **Reusable React Components** with **examples**.

**Table of Content**

* [What are Reusable React Components ?](https://www.geeksforgeeks.org/how-to-make-reusable-react-components/#what-are-reusable-react-components-)
* [How to make Reusable React Components ?](https://www.geeksforgeeks.org/how-to-make-reusable-react-components/#how-to-make-reusable-react-components-)

**What are Reusable React Components ?**

**Reusable**[**React Components**](https://www.geeksforgeeks.org/reactjs-components/) are pieces of code that can be **shared** and **reused**across different files in your application. It is a modular piece of UI that takes data from **props**to build complex and interactive user interfaces. It encapsulates the login that can be reused again.

**Features of reusable components:**

* It can accept props data to customize according to data.
* It can be used to encapsulate the same logic and reuse it again.
* It helps us to break the code into smaller units that can be reused again.
* It makes it easy to maintain the application.

**How to make Reusable React Components ?**

When creating reusable React components, it’s important to keep in mind two key factors:

**1. Avoid side Effects:**

It is not recommended to include logic that interacts with external data, such as making API calls, directly inside a reusable component. Instead, you should pass this logic as props to the component.

For example, if a button not only has a visual function but also fetches data from the internet, it may not be reusable in other contexts.

In this case, we have a reusable button component that lacks best practices. In the example section, I will demonstrate why this is the case.

// It is a Reusable ButtonReusable component and this is a bad practices  
const ButtonReusable = () => {  
 return (  
 <button> Click Me </button>  
 );  
}

**2. Use Props:**

[Props](https://www.geeksforgeeks.org/reactjs-props-set-1/) are parameters passed to a component to customize its behavior and appearance, making it reusable for different purposes.

// It is a buttonReusable component that can change its color  
const ButtonReusable = ({ color }) => {  
 return (  
 <button style={{ backgroundColor: blue }}> Click Here </button>  
 );  
}

This is considered bad practice because the label on the button is fixed as “**Click Here**“. If you wish to change the text on your button to, for example, “**Sign Up**“, you would have to go back to the button component and make that change. This means that every time you want to use a different text, you would have to edit the code again. In other words, the button would no longer be reusable.

**Steps to Create a React Application:**

**Step 1:**Create a React application using the following command:

npx create-react-app gfg

**Step 2:** After creating your project folder(i.e. gfg), move to it by using the following command:

cd gfg

**Example:** This below example demonstrate the Reusable React Components.

In this example, You can see **ProductList**and **ProductItem**component can be composed together and accept **props**for customization and are reusable with different data sets.

*//File path: src/App.js*

**import** React **from** 'react';

**import** ProductList **from** './ProductList.js';

**const** App = () => {

**const** products = [

{ id: 1, name: 'Product 1', price: 10 },

{ id: 2, name: 'Product 2', price: 20 },

{ id: 3, name: 'Product 3', price: 30 },

];

**return** (

<div style={{ margin: '5px' }}>

<h2 style={{ color: 'green' }}>

GeeksForGeeks | Reusable Components Example

</h2>

<ProductList products={products} />

</div>

);

};

**export** **default** App;

*//File path: src/ProductList.js*

**import** React **from** 'react';

**import** ProductItem **from** './ProductItem.js';

**const** ProductList = ({ products }) => {

**return** (

<div>

<h2>Products List</h2>

<table border={1}>

<thead>

<tr>

<th>No</th>

<th>Name</th>

<th>Price</th>

</tr>

</thead>

<tbody>

{products.map((product) => (

<ProductItem key={product.id}

product={product} />

))}

</tbody>

</table>

</div>

);

};

**export** **default** ProductList;

*//File path: src/ProductItem.js*

**import** React **from** 'react';

**const** ProductItem = ({ product }) => {

**return** (

<tr>

<td>{product.id}</td>

<td>{product.name}</td>

<td>{product.price}</td>

</tr>

);

};

**export** **default** ProductItem;

**To run the application use the following command:**

npm run start

**Output:** Now go to **http://localhost:3000** in your browser

