React-Functional Components

Here's a summary of functional components in React, along with an example:

What are Functional Components?

Functional components in React are a type of component that is defined as a function rather than a class. They were introduced in React 16.8 and provide a more concise way to write simple components.

Characteristics:

1. **Functions**: Functional components are defined using the `function` keyword.
2. **No `this` context**: Unlike class-based components, functional components do not have access to the `this` context.
3. **No lifecycle methods**: Functional components do not have lifecycle methods like `componentDidMount`, `componentWillUnmount`, etc.
4. **Stateless**: By default, functional components are stateless.

Example:

```
 import React from 'react';

function MyComponent() {
  return <h1>Hello, World!</h1>;
}
```

In this example, we define a simple component called `MyComponent` as a function. This function returns an `<h1>` element with the text "Hello, World!".

Using Hooks:

One of the key features of functional components is that they can use React Hooks to manage state and side effects. Here's an updated version of our example:

```
 import React, { useState } from 'react';

function MyComponent() {
  const [count, setCount] = useState(0);

  return (
    <div>
      <p>Count: {count}</p>
      <button onClick={() => setCount(count + 1)}>Increment</button>
    </div>
  );
}
```

In this example, we use the `useState` Hook to create a state variable called `count` and an `increment` button that updates the count when clicked.

I hope this summary helps! Let me know if you have any questions or need further clarification.

Curated by Brajesh Kumar