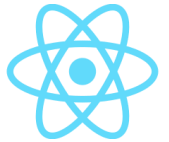
Function Components vs Class Components

NGUYEN TRONG TIEN



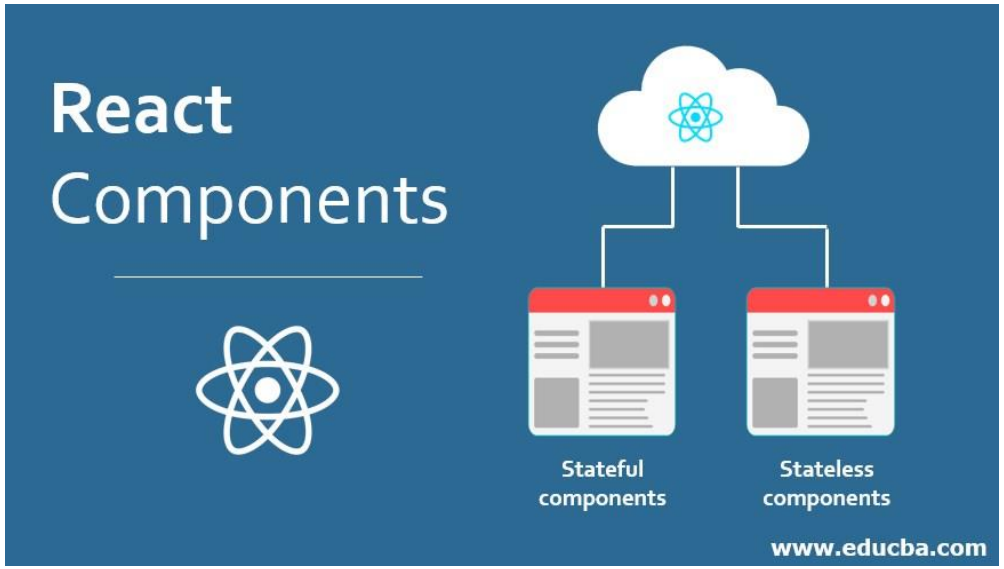
What are React Component?

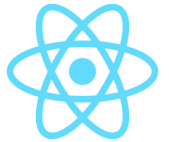
In React, components are the building blocks of a user interface. They are reusable, self-contained pieces of code that represent a part of the UI. React allows you to break down your UI into smaller components, which makes it easier to manage and maintain your codebase.

Component

There are two primary ways to create components: **function** and **class** components.

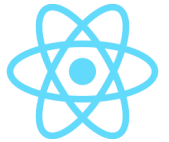
Each has its own syntax and use cases, although with the introduction of **React Hooks**, the **gap** between them has **narrowed** significantly. But the selection of appropriate component types is still very crucial for building efficient and maintainable React applications.





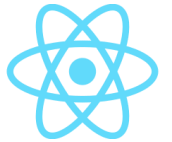
Function Component

Function Component: These are simple JavaScript functions that take props as input and return JSX elements. They are often used for presentational or stateless components.



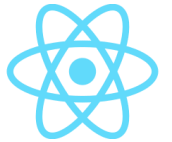
Class Component

These are ES6 classes that extend from **React.Component** or **React.PureComponent**. They have a **render()** method where you define the structure of your component's UI using JSX. Class components are used for components that need to manage state or have lifecycle methods.



State Management

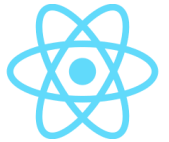
Traditionally, function components were stateless and couldn't hold their own state. However, with the introduction of React Hooks (like useState), function components can now manage state using Hooks.



Lifecycle Methods

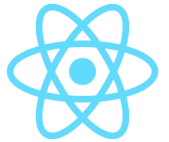
Function components don't have lifecycle methods.

However, with **React Hooks**, you can use the **useEffect** Hook to replicate lifecycle behavior (like **componentDidMount**, **componentDidUpdate**, **componentWillUnmount**, and so on).



componentDidMount

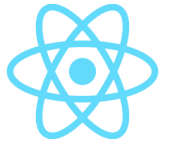
This method is invoked immediately after a **component is mounted** (that is, inserted into the DOM tree). It is commonly used to perform initial setup, **such as fetching data from an API** or **setting up event listeners**.



componentDidUpdate

This method is invoked immediately after updating occurs.

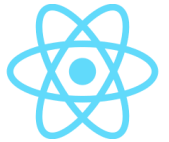
It is triggered whenever the component's props or state changes. **It is commonly used to perform actions based on the updated state or props, such as making additional API calls.**



componentWillUnmount

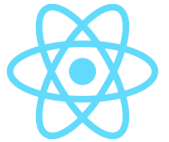
This method is invoked immediately before a **component is unmounted** and **destroyed**.

It is commonly used to perform cleanup, such as removing event listeners or cancelling any ongoing tasks.



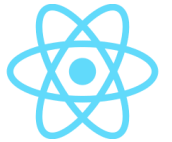
Class Component

Syntax: Class components are ES6 classes that extend from **React.Component** or **React.PureComponent**. They have a **render()** method where you define the structure of your component's UI using JSX.



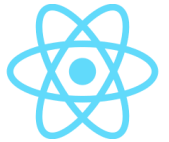
State Management

Class components can hold and manage local state using the **this.state** property. They can also update state using **this.setState().**



Lifecycle Methods

Class components have access to various lifecycle methods like **componentDidMount**, **componentDidUpdate**, and **componentWillUnmount**, which allow you to hook into different stages of a component's lifecycle.

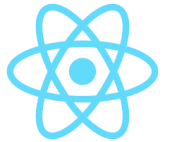


Explain

componentDidMount is used to fetch initial data when the component mounts.

componentDidUpdate is used to log a message whenever the data state changes.

componentWillUnmount is used to log a message before the component is unmounted.



Instance Methods

You can define custom methods directly on the class, which can be helpful for organizing your component's logic.


THANK YOU!