Lifecycle Methods

This lesson briefly introduces the lifecycle methods that can be overridden in a React ES6 class component

Now it's time to get real with APIs and move past sample data. If you are not familiar, I encourage you to read my article on how I got to know APIs.

For our first foray into the concept, we will be using Hacker News, a solid news aggregator about tech topics. In this exercise, we will use the Hacker News API to fetch trending stories. There are basic and search APIs to get data from the platform. Search makes sense in the application that we are building in this course because we want to be able to search Hacker News stories. Visit the API specification to get an understanding of the data structure.

Lifecycle Methods

You may remember lifecycle methods were mentioned briefly in the last chapter, as a hook into the lifecycle of a React component. They can be used in ES6 class components, but not in functional stateless components. Besides the `render()` method, there are several methods that can be overridden in a React ES6 class component. All of these are the lifecycle methods.

Mounting

We have already covered two lifecycle methods that can be used in an ES6 class component:

- The constructor is only called when an instance of the component is created and inserted in the DOM. The component gets instantiated in a process called mounting.
- The `render()` method is called during the mount process too, but also when the component updates. Each time the state or the props of a

component changes, the `render()` method is called.

There are two more lifecycle methods when mounting a component: `getDerivedStateFromProps()` and `componentDidMount()`. The constructor is called first, `getDerivedStateFromProps()` is called before the `render()` method, and `componentDidMount()` is called after the `render()` method.

Overall, the mounting process has 4 lifecycle methods, invoked in the following order:

- constructor()
- getDerivedStateFromProps()
- render()
- componentDidMount()

Updating

For the update lifecycle of a component when the state or the props change, there are 5 lifecycle methods, in the following order:

- getDerivedStateFromProps()
- shouldComponentUpdate()
- render()
- getSnapshotBeforeUpdate()
- componentDidUpdate()

Unmounting

Lastly, there is the unmounting lifecycle. It has only one lifecycle method: `componentWillUnmount()`.

Description of Lifecycle Methods

You don't need to know all the lifecycle methods from the beginning, and even in a large React application you'll only use a few of them besides the

`constructor()` and the `render()` methods. Still, it is good to know each lifecycle method can be used for specific purposes:

- **constructor(props)** is called when the component gets initialized. You can set an initial component state and bind class methods during that lifecycle method.

- **static getDerivedStateFromProps(props, state)** is called before the `render()` lifecycle method, both on the initial mount and on the subsequent updates. It should return an object to update the state, or null to update nothing. It exists for **rare** use cases where the state depends on changes in props over time. It is important to know that this is a static method and it doesn't have access to the component instance.

- **render()** is a mandatory lifecycle method that returns elements as an output of the component. The method should be pure, so it shouldn't modify the component state. It gets an input as props and state and returns an element.

- **componentDidMount()** is called once, when the component mounted. That's the perfect time to do an asynchronous request to fetch data from an API. The fetched data is stored in the local component state to display it in the `render()` lifecycle method.

- **shouldComponentUpdate(nextProps, nextState)** is always called when the component updates due to state or props changes. You will use it in mature React applications for performance optimization. Depending on a boolean that you return from this lifecycle method, the component and all its children will render or will not render on an update lifecycle. You can prevent the render lifecycle method of a component.

- **getSnapshotBeforeUpdate(prevProps, prevState)** is a lifecycle method, invoked before the most recently rendered output is committed to the DOM. In rare cases, the component needs to capture information from the DOM before it is potentially changed. This lifecycle method enables the component to do it. Another method (`componentDidUpdate()`) will receive any value returned by `getSnapshotBeforeUpdate()` as a parameter.

- **componentDidUpdate(prevProps, prevState, snapshot)** is a lifecycle method that is invoked immediately after updating, but not for the initial render. You can use it as to perform DOM operations or to perform more

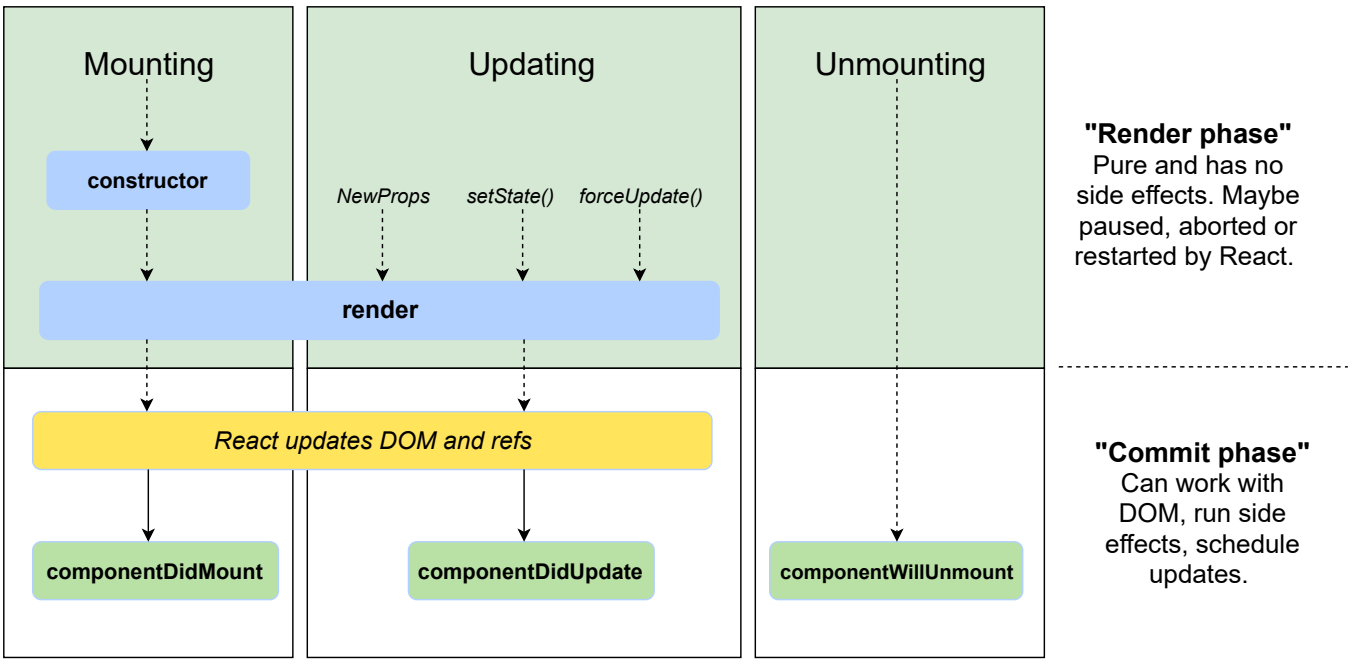
asynchronous requests. If your component implements the

`getSnapshotBeforeUpdate()` method, the value it returns will be received as the `snapshot` parameter.

- **componentWillUnmount()** is called before you destroy your component. You can use this lifecycle method to perform any cleanup tasks.

As you may have gathered, the `constructor()` and `render()` lifecycle methods are the most commonly used lifecycle methods for ES6 class components. The `render()` method is always required to return a component instance.

Lastly, `componentDidCatch(error, info)` was introduced in React 16 as a way to catch errors in components. For instance, displaying the sample list in your application works fine, but there could be a time when a list in the local state is set to `null` by accident (e.g. when fetching the list from an external API, but the request failed and you set the local state of the list to null). It becomes impossible to filter and map the list because it is `null` and not an empty list. The component would be broken, and the whole application would fail. Using `componentDidCatch()`, you can catch the error, store it in your local state, and show a message to the user.



The React LifeCycle Diagram (Taken from http://projects.wojtekmaj.pl/react-lifecycle-methods-diagram/)

Q

There is a lifecyle method in React component which is overidden to prevent updating the components. Which of the following is that method?

COMPLETED 0%

1 of 1

Further Reading:

- Read about lifecycle methods in React
- Read about the state related to lifecycle methods in React
- Read about error handling in components