Web Front-End Development
Week 13: React Lifecycle

React Lifecycle

React makes lifecycle hooks available through methods that are called throughout a component's lifecycle. Once you understand what hooks are available and when these methods get called you can use them to enhance your component's functionality.

Lifecycle methods are kind of like events – they get called automatically at various stages of a component's life.
(slides)

Initial Rendering Phase

When your component is about to start its life and make its way to the DOM, the following lifecycle methods get called:

getDefaultProps()
- gets called before your component is even created or any props from parents are passed in
- allows you to specify the default value of this.props
- returns an object that will represent the default state of this.props on the component.
- done within the constructor function of your component class

getInitialState()
- gets called before your component is created
- allows you to specify the default value of this.state before your component is created
- returns an object with the state of the component when it initially renders
- done within the constructor function of your component class

componentWillMount()
- Legacy, don't use

render()
- Responsible for returning a single root node (which may have many child nodes inside it)
- Every component must have this method defined
- If you don't wish to render anything return null or false

componentDidMount()
- This method gets called immediately after the component has been rendered to the DOM.
- You can now safely perform any DOM querying operations as the component is ready

With the exception of the render method, all of these lifecycle methods can fire only once. After your components get added to the DOM, they can potentially update and re-render when a prop or state change occurs.

State Change
When a state change occurs your component will call its render method again. Any components that rely on the output of this component will also get their render methods called as well. This is done to ensure that our component is always displaying the latest version of itself. The following lifecycle methods also get called:

shouldComponentUpdate(nextProps, nextState)

- If you don't want your component to update when a state change occurs this method allows you to control this updating behavior.
  - return a true value for the component to update
  - return a false value for this component to NOT update

componentWillUpdate(nextProps, nextState)

- Legacy, don't use

render()

componentDidUpdate(prevProps, prevState)

- gets called after your component updates and the render method has been called
- good place to execute any code after the update takes place

Props Change
The other time your component updates is when its prop value changes after it has been rendered into the DOM. There's only one method we haven't already seen, the others have identical behavior as when state changes.

componentWillReceiveProps(nextProps)

- Legacy, don't use

shouldComponentUpdate(nextProps, nextState)
render()
componentDidUpdate(prevProps, prevState)

Unmounting Phase
When a component is removed from the DOM it's called "unmounting". In applications with many components, it's very important to free up resources taken by the components when they are destroyed.
componentWillUnmount()

- gets called when the component is removed from the DOM
- perform any cleanup-related tasks here such as removing event listeners, stopping timers, etc. After this method gets called, your component is removed from the DOM