

Component Lifecycles and ReactDOM

Developing Applications using ReactJS



Objectives



- To understand the component lifecycle methods and when they are called
- To know how to use lifecycle methods
- To understand the ReactDOM package and its methods

The Component Lifecycle

- Methods that can be overridden to run code at particular times in the process
- Along with render() there are three different types:
 - Mounting
 - Called when an instance of component is being created an inserted into the DOM
 - Updating
 - Called when a component is being re-rendered, usually because of a change to props or state
 - Unmounting
 - Called when a component is being removed from the DOM
- Some of these methods are prefixed with will indicating that they are called right before something happens
- Others are prefixed with did and are called right after something has happened



The render() method

- Method is required in all components and application will fail if not included
- Should examine this.props and this.state and return single React element
 - Can be native DOM element or custom composite element
 - Can return null or false to indicate nothing to be rendered
- Should not change component's state
 - Returns same result each time it is called
 - Does not directly interact with browser
 - Browser interaction should be done within lifecycle methods



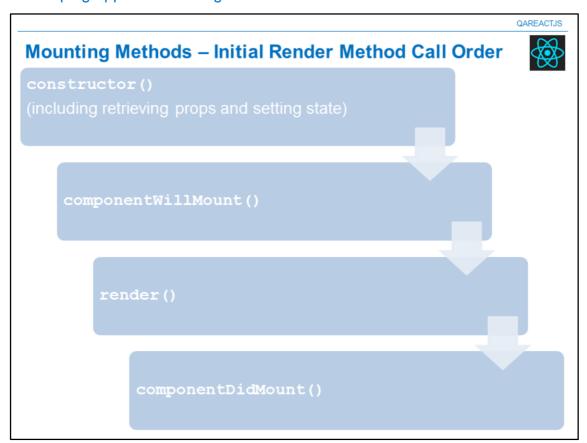
Mounting Methods

- Called when an instance of a component is being created and inserted into the DOM
 - constructor()
 - Constructor for a React component
 - Should have a call to super (props) before any other statement
 - Defines this.props in the constructor
 - Correct place to initialise state
 - If state is not initialised and methods are not bound, there is no need for a constructor



Mounting Methods

- Called when an instance of a component is being created and inserted into the DOM
 - componentWillMount()
 - Invoked immediately before mounting occurs
 - Called before render() so changes in state do not trigger rerendering
 - Generally recommended to use a constructor instead
 - componentDidMount()
 - Invoked immediately after a component is mounted
 - Initialisation that requires DOM nodes should go here
 - Good place to instantiate request for data loads from remote endpoint
 - Setting state will trigger component re-rendering





Updating Methods

- Called when a component is being re-rendered after changes to props or state
 - componentWillReceiveProps(nextProps)
 - Invoked before a mounted component receives new props
 - Can compare this.props and nextProps and perform state transitions, using this.setState() in this method
 - May be called even if props haven't changed
 - Not invoked if setState () is called

SF



Updating Methods

- Called when a component is being re-rendered after changes to props or state
 - shouldComponentUpdate(nextProps, nextState)
 - Lets React know if a component's output is not affected by current change in state or props
 - Default behaviour is to re-render on every state change
 - Invoked before rendering when new props or state are being received, defaulting to true
 - Not called for initial render or when forceUpdate() is used
 - Returning false doesn't prevent child components from rerendering when their state changes
 - Does stop componentWillUpdate(), render() and componentDidUpdate() from being called

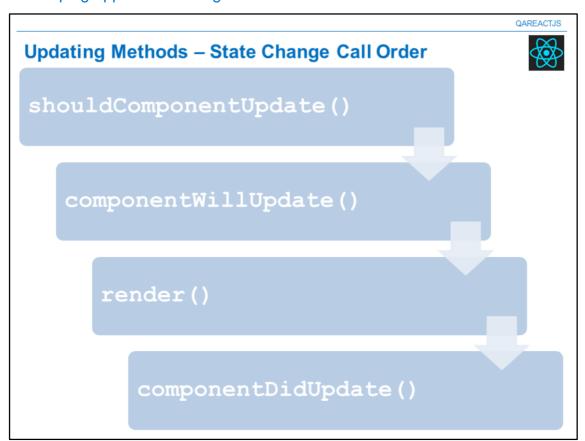
Updating Methods

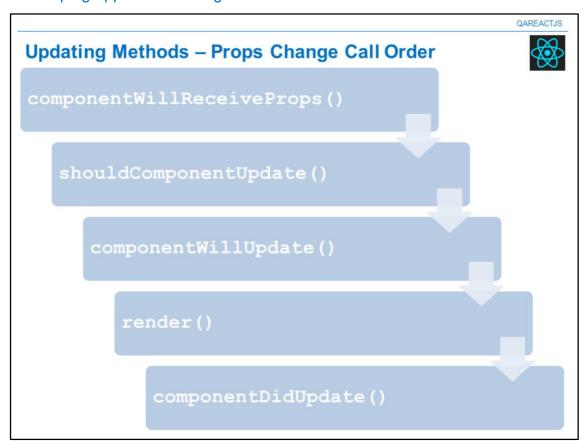
- Called when a component is being re-rendered after changes to props or state
 - componentWillUpdate(nextProps, nextState)
 - Invoked immediately before rendering when new props or state are being received
 - Opportunity to perform preparation before an update occurs
 - Not called during the component's initial render
 - Cannot use this.setState() in this method
 - Updating state in response to a prop change should be done in the componentWillReceiveProps() method instead
 - Not invoked if shouldComponentUpdate() returns false



Updating Methods

- Called when a component is being re-rendered after changes to props or state
 - componentDidUpdate(prevProps, prevState)
 - Invoked immediately after an update occurs
 - Opportunity to operate on the DOM after a component update
 - Good place to do network requests
 - Compare current props to previous props as network request may not be necessary if props have not changed
 - Not called during the component's initial render
 - Not invoked if shouldComponentUpdate() returns false







Unmounting Methods

- Called when a component is being removed from the DOM
 - componentWillUnmount()
 - Invoked immediately before a component is unmounted and destroyed
 - Opportunity to perform necessary cleanup, e.g. invalidating timers, cleaning up DOM elements created in componentDidMount()



React and the DOM

- Strength of React is its relationship to the DOM
 - JavaScript is very fast
 - DOM is very slow
- React creates a copy of the DOM known as the 'Virtual DOM'
 - Actual DOM only affected if changes in the Virtual DOM
 - Only changes elements in actual DOM changed in Virtual DOM
 - Means whole DOM is not re-rendered and therefore is much quicker
- ReactDOM was split from the core library in React 0.14
 - Provides render(), findDOMNode() and umountComponentAtNode() methods
 - Already seen and used the render () method in main.js several times

ReactDOM.render()



- DIFFERENT TO THE LIFECYCLE render() METHOD!
- Called with an element, container and optional callback

```
ReactDOM.render(
        element,
        container,
        [callback]
```

- Renders a React element into the DOM to the container and returns a reference to the component (or null)
- If element already exists it will only be updated if necessary
 - DOM only mutated to reflect last React element if it needs to
- Optional callback executed after the component is rendered or updated



findDOMNode()

Called with a component argument to find

ReactDOM.findDOMNode(component)

- If component is mounted into DOM, returns corresponding native DOM element
 - Useful for reading values out of DOM and performing DOM measurements
 - However, in most cases a 'ref' can be attached to DOM node avoiding use of findDOMNode
- NOTE: Only works on mounted components
 - Calling on component that is not yet mounted would cause exception to be thrown
 - Cannot be used on Functional Components
- Use is generally discouraged due to problems with component abstraction



unmountComponentAtNode()

Called with an argument of a component container

ReactDOM.unmountComponentAtNode(container)

- Removes a mounted React component from the DOM
 - Cleans up its event handlers and state
 - If no component was mounted in the container, function call does nothing
 - Returns true if a component was unmounted and false if no component to unmount was found



Objectives

- To understand the component lifecycle methods and when they are called
- To know how to use lifecycle methods
- To understand the ReactDOM package and its methods

Exercise Time! • EG05 – Using Component Lifecycle Methods