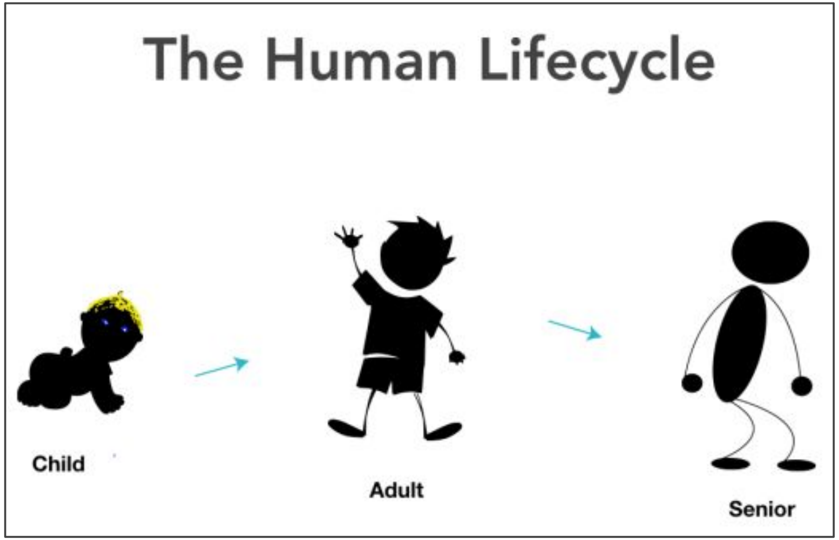
ReactJS Lifecycle

Isumi
Batam, Jan 2020

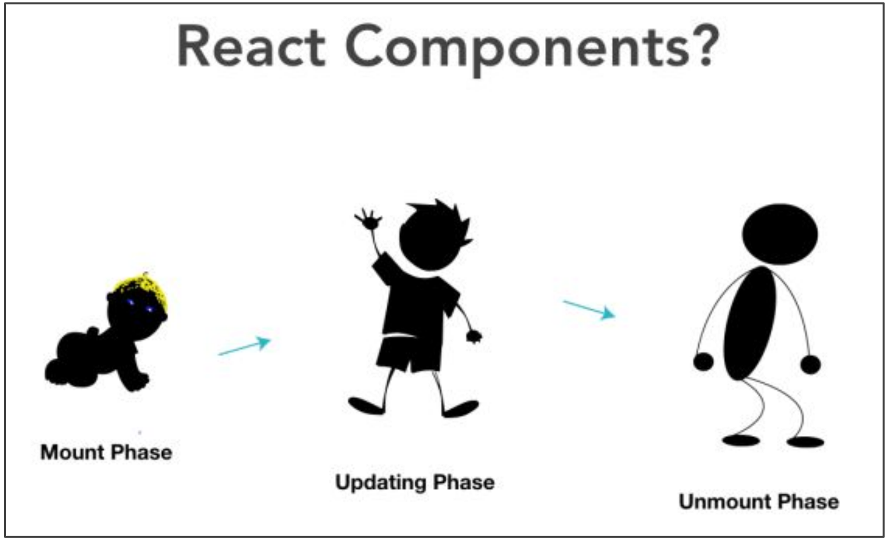
GOALS

Lifecycle Methods:

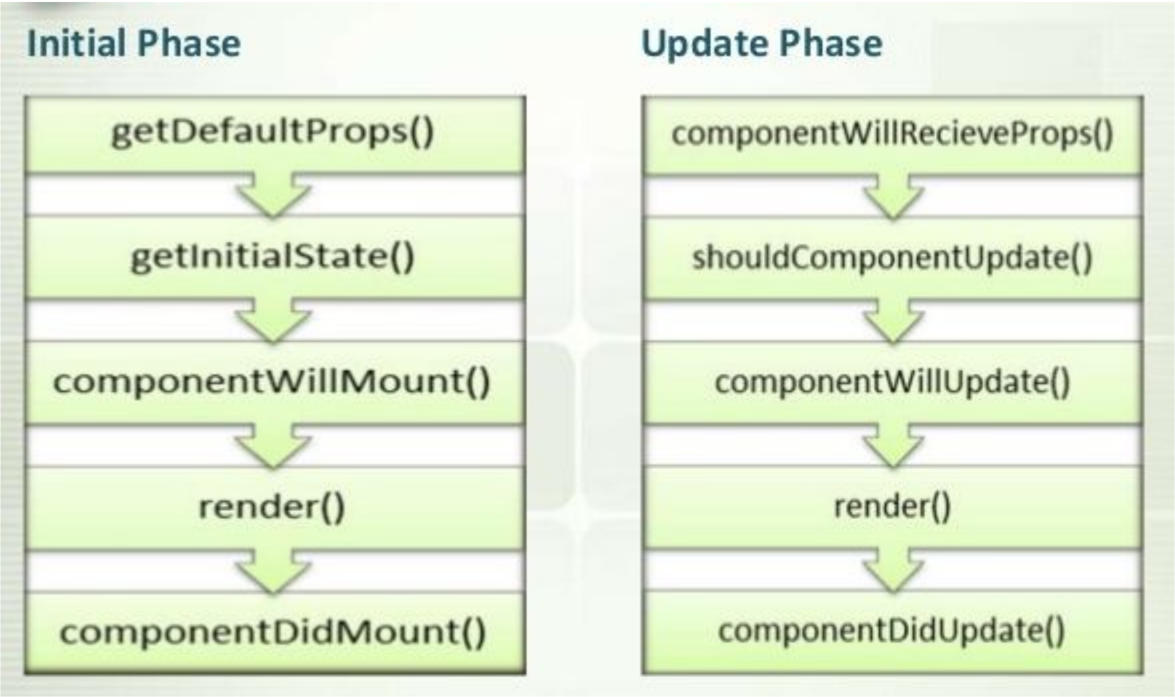
- Mounting
- Updating
- Unmounting
- Error Handling



The Human Lifecycle



React Components?



Initial Phase

getDefaultProps()

getInitialState()

componentWillMount()

render()

componentDidMount()

Update Phase

componentWillRecieveProps()

shouldComponentUpdate()

componentWillUpdate()

render()

componentDidUpdate()

Mounting

At this phase the component is created (your code, and react’s internals) then inserted into the DOM.

When a component is rendered for the *first* time.

Initialization/mounting (step by step):

- constructor -> set initial state using this.state = {}
- componentWillMount(){} -> **before** the render method is executed the first time
- render() -> returned markup is added to the DOM
- componentDidMount(){} -> **after** the component renders (for AJAX requests)

Updating

In this phase a React component undergoes growth by being updated via changes in props or state.

Whenever we call setState, we can tap into quite a few life cycle hooks:

- componentWillReceiveProps(nextProps) {} -> only called when the props have changed and when this is not an initial rendering.
- static getDerivedStateFromProps -> to change state depending on changes to the component and should be used instead of componentWillReceiveProps
- shouldComponentUpdate(nextProps, nextState) {} -> allows us to pass on a re-rendering if we know one is not needed.
- componentWillUpdate(nextProps, nextState) {} -> gets called as soon as the the shouldComponentUpdate returns **true**
- render -> changing state triggers a re-render unless you modify shouldComponentUpdate
- getSnapshotBeforeUpdate(prevProps, prevState) {} -> before the most recently rendered output is committed to e.g. the DOM.
- componentDidUpdate(prevProps, prevState) {} -> after the render method

Unmounting

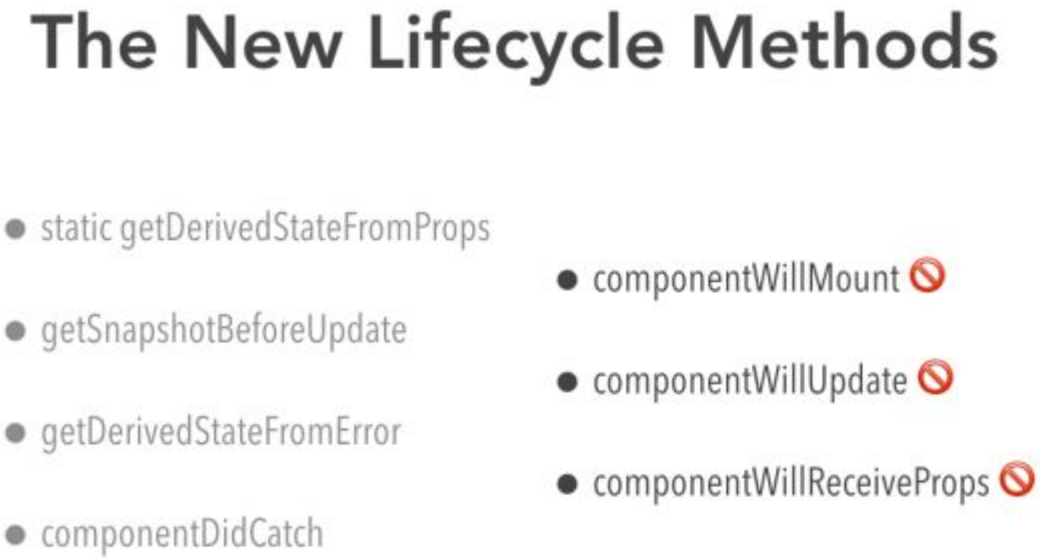
This is the phase the component is removed from the DOM.

- componentWillUnmount() {} -> before the component is removed from the DOM.

Error Handling

Sometimes your code doesn’t run or there’s a bug somewhere.
When this happens, the component is in the error handling phase.

- static getDerivedStateFromError(error) {} -> updating state to display an error screen.
- componentDidCatch(error, info) {} -> catch errors during mounting, rendering and in other lifecycle methods.



The New Lifecycle Methods

- static getDerivedStateFromProps
- getSnapshotBeforeUpdate
- getDerivedStateFromError
- componentDidCatch

- componentWillMount 🚫
- componentWillUpdate 🚫
- componentWillReceiveProps 🚫

Exercise

Just try it! ;)

Tutorial in here

Complete tutorial (new lifecycle method implementation and using Hooks) in here