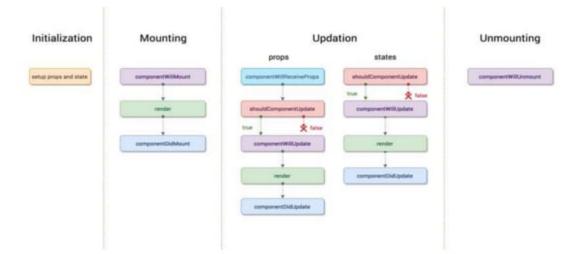
## What is lifecycle methods and why it is important?

- Around us everything goes through a cycle of taking birth, growing and at some point of time it will die.
- Consider trees, any software application, yourself, a div container or UI component in a web browser, each of these takes birth, grows by getting updates and dies.
- The lifecycle methods are various methods which are invoked at different phases of the lifecycle of a component.

## Four phases of a React component

- The React component goes through the following phases
  - Initialization
  - Mounting
  - Update
  - Unmounting

# Visual representation of the phases and the methods of ReactJs lifecycle.



#### Initialization

- In this phase the React component prepares for the upcoming tough journey, by setting up the initial states and default props, if any.
- The component is setting up the initial state in the constructor, which can be changed later by using the setState method.

## Mounting

- After preparing with basic needs, state and props, our React Component is ready to mount in the browser DOM.
- This phase gives hook methods for before and after mounting of components.
- The methods which gets called in this phase are
  - componentWillMount()
  - render()
  - componentDidMount()

## Mounting conti.....

#### componentWillMount()

- This method is executed just before the React Component is about to mount on the DOM.
- This method is executed once in a lifecycle of a component and before first render.
- Usage: This method is used for initializing the states or props, there is a huge debate going on to merge it with the constructor.

## Mounting conti.....

#### render()

- This method is mounts the component onto the browser.
- This is a pure method, which means it gives the same output every time the same input is provided.

## Mounting conti.....

#### componentDidMount()

- This this is the hook method which is executed after the component did mount on the dom.
- This method is executed once in a lifecycle of a component and after the first render.
- As, in this method, we can access the DOM
- Usage: this is the right method to integrate API

#### Update

- This phase starts when the react component has taken birth on the browser and grows by receiving new updates.
- The component can be updated by two ways:
  - sending new props
  - updating the state.

- sending new props
  - The methods which gets called in this phase are:
    - componentWillReceiveProps()
    - shouldComponentUpdate()
    - componentWillUpdate()
    - render()
    - componentDidUpdate()

#### componentWillReceiveProps()

- This method gets executed when the props have changed and is not first render.
- Sometimes state depends on the props, hence whenever props changes the state should also be synced, This is the method where it should be done.
- Usage: This is how the state can be kept synced with the new props.

#### shouldComponentUpdate()

This method tells the React that when the component receives new props or state is being updated, should React re-render or it can skip rendering?

#### componentWillUpdate()

- This method is executed only after the shouldComponentUpdate() returns true
- This method is only used to do the preparation for the upcoming render, similar to componentWillMount() or constructor.

#### render()

This method is mounts the component onto the browser

#### componentDidUpdate()

- This method is executed when the new updated component has been updated in the DOM.
- This method is used to re trigger the third party libraries used to make sure these libraries also update and reload themselves.

- updating the state
  - The methods which gets called in this phase are:
    - shouldComponentUpdate()
    - componentWillUpdate()
    - render()
    - componentDidUpdate()

## Unmounting

- In this phase, the component is not needed and the component will get unmounted from the DOM.
- The method which is called in this phase:
  - componentWillUnmount()

## Unmounting Conti....

#### componentWillUnmount

- This method is the last method in the lifecycle.
- This is executed just before the component gets removed from the DOM.
- Usage: In this method, we do all the cleanups related to the component.
  - For example, on logout, the user details and all the auth tokens can be cleared before unmounting the main component.