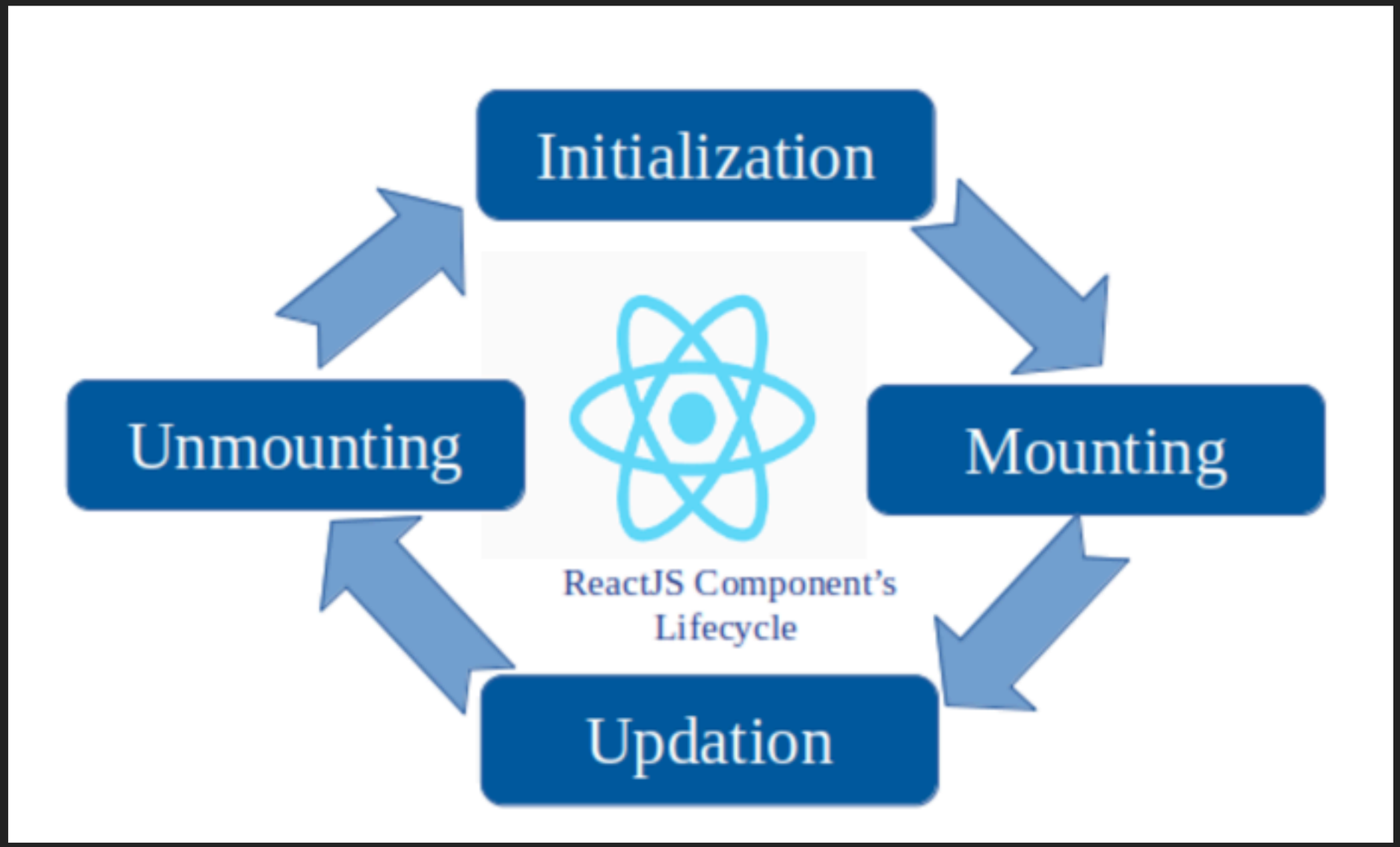
REACT COMPONENT LIFECYCLE

BY KIRK SULLIVAN

REACT COMPONENT LIFECYCLE

- A Components lifecycle is classified into four parts:
  - Initialization
  - Mounting
  - Updating
  - Unmounting

REACT COMPONENT LIFECYCLE



- Just like in life - Components in React are created, grow by updating, and then die (unmount from the DOM).

- Initialization
  - This is the phase in which the component is going to start its journey by setting up the state (see below) and the props. This is usually done inside the constructor method.

```
3  class App extends Component {
4    constructor(props) {
5      super(props);
6      this.state = {
7        items: [],
8        isLoaded: false,
9      };
10   }
```

- Mounting
  - Mounting is the phase in which our React Component mounts on the DOM.
  - This phase is after the initialization is completed. In this phase our component renders for the first time. The methods used during this phase are:

1. componentWillMount()
   - This method is called just before a component mounts on the DOM or the render method is called.
   - Nothing can be done with the DOM (i.e. updating the data with API response) as it has not been mounted.

2. componentDidMount()

   - This method is called after the component gets mounted on the DOM. Like componentWillMount, it is called once in a lifecycle. Before the execution of this method, the render method is called (we can access the DOM). We can make API calls and update the state with the API response.

REACT COMPONENT LIFECYCLE

```
3  class LifeCycle extends React.Component {
4      componentWillMount() {
5          console.log('Component will mount!')
6      }
7      componentDidMount() {
8          console.log('Component did mount!')
9          this.getList();
10     }
11     getList=()=>{
12     /*** method to make api call***/
13     }
14     render() {
15         return (
16             <div>
17                 <h3>Hello mounting methods!</h3>
18             </div>
19         );
20     }
21 }
```

Updating

- After the mounting phase where the component has been created, the update phase comes into the scene. This is where component's state changes and hence, re-rendering takes place.
- In this phase, the data of the component (state & props) updates in response to user events like clicking, typing etc... This results in the re-rendering of the component. The methods that are available in this phase are:

1. shouldComponentUpdate()

   - This method determines whether the component should be updated or not. By default, it returns true. At some point, if you want to re-render the component on some condition, then shouldComponentUpdate method is the right place.

   - For example, you want to only re-render your component when there is a change is props - then use the power of this method. It receives arguments nextProps and nextState which will help us decide whether to re-render by doing a comparison with the current prop value.

2. componentWillUpdate()

- It’s called before the re-rendering of the component takes place. It’s called once after the ‘shouldComponentUpdate’ method. If you want to perform some calculation before re-rendering of the component and after updating the state & props, then this it the best place to do it. Similar to the ‘shouldComponentUpdate method it also receives arguments like nextProps and next State

3. componentDidUpdate()

- This method is called just after the re-rendering of the component. After new (updated) component gets updated on the DOM, the ‘componentDidUpdate’ method is executed. This method receives arguments like prevProps and prevState.

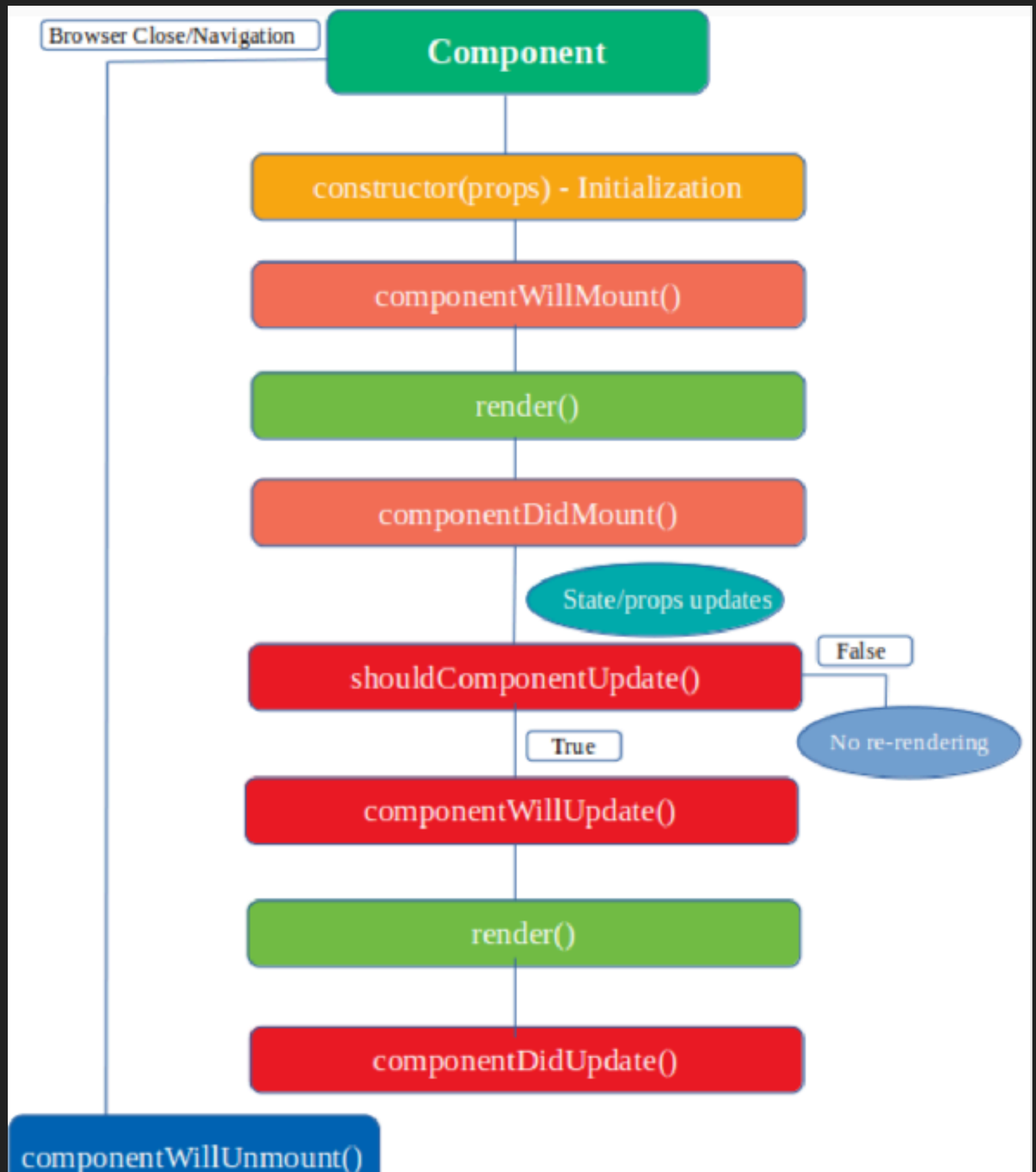
REACT COMPONENT LIFECYCLE

```
class LifeCycle extends React.Component {
  constructor(props)
  {
    super(props);
    this.state = {
      date : new Date(),
      clickedStatus: false,
      list:[]
    };
  }
  componentWillMount() {
      console.log('Component will mount!')
  }
  componentDidMount() {
      console.log('Component did mount!')
      this.getList();
  }
  getList=()=>{
  /*** method to make api call***/
  fetch('https://api.mydomain.com')
      .then(response => response.json())
      .then(data => this.setState({ list:data }));
  }
  shouldComponentUpdate(nextProps, nextState){
    return this.state.list!==nextState.list
  }
  componentWillUpdate(nextProps, nextState) {
      console.log('Component will update!');
  }
  componentDidUpdate(prevProps, prevState) {
      console.log('Component did update!')
  }
  render() {
      return (
          <div>
              <h3>Hello Mounting Lifecycle Methods!</h3>
          </div>
      );
  }
}
```

- Unmounting
  - This is the last phase in the component's lifecycle. The component gets unmounted from the DOM in this phase. The method that is available is:

1. componentWillUnmount()
   - This method is called before the unmounting of the component takes place. Before the removal of the component from the DOM, 'componentWillUnmount' executes. This method denotes the end of the component's lifecycle.

REACT COMPONENT LIFECYCLE



REACT COMPONENT LIFECYCLE

- Links:
  - React Documentation
  - MDN