Life cycle of a component in Vue:

A Vue component goes through 4 phases.

1. Creation
2. Mounting
3. Updating
4. Unmounting

Each phase provides different hooks/methods to perform certain actions.

Creation:

When a component is about to be created. It provides two hooks

1. **beforeCreate():** Called when the instance is initialized, before any other options API and after props resolution.
2. **created():** Called after the instance has finished processing all state-related options. It is the best place to write data fetching APIs.

Mounting:

In this phase the component's template/HTML is mounted onto the DOM. It provides two hooks.

1. **beforeMount():** It is called when the component finishes setting up reactive state and no DOM node is created yet.
2. **mounted():** Called after the component has been mounted. This hook is typically used for performing side effects that need access to the component's rendered DOM.

Updating:

This phase is triggered when component’s data or computed property changes or when component re-renders. It provides two hooks.

1. **beforeUpdate():** Called right before the component is about to update its DOM tree due to a reactive state change.

2. **updated():** Called after the component has updated its DOM tree due to a reactive state change.

Unmounting:

This phase is triggered when a component is about to be removed from the DOM.

1. **beforeUnmount():** Called right before a component instance is to be unmounted.
2. **unmounted():** Called after the component has been unmounted. This hook can be used to remove or clear pending Ajax calls, added event listeners and timers.

There are some other hooks/methods which are not specific to any above phases.

1. **activated():** Called after the component instance is inserted into the DOM as part of a tree cached by <KeepAlive/>
2. **deactivated():** Called after the component instance is removed from the DOM as part of a tree cached by <KeepAlive/>
3. **errorCapture():** Called when an error propagating from a descendent component has been captured.
4. **renderTracked():** Called when a reactive dependency has been tracked by the component's render effect. It is used for debugging purposes.
5. **renderTriggered()**: Called when a reactive dependency triggers the component's render effect to be rerun. It is used for debugging purposes.