## **¿Qué son los métodos del ciclo vida?**

Todos los componentes en React pasan por una serie de fases que generalmente se denominan “Ciclo de Vida del componente” es un proceso que React hace en cada componente, en algunos casos no podemos verlos como un bloque de código y en otros podemos llamarlos en nuestro componente para asignar una actividad según sea el caso necesario.

Los componentes en react pasan por un Montaje, Actualización, Desmontaje y Manejo de errores.

## **Montaje:**

En esta fase nuestro componente se crea junto a la lógica y los componentes internos y luego es insertado en el DOM.

## **Actualización:**

En esta fase nuestro componente está al pendiente de cambios que pueden venir a través de un cambio en “state” o “props” esto en consecuencia realizan una acción dentro de un componente.

## **Desmontaje:**

En esta etapa nuestro componente “Muere” cuando nosotros no necesitamos un elemento de nuestra aplicación, podemos pasar por este ciclo de vida y de esta forma eliminar el componente de la representación que tiene en el DOM.

## **Manejo de Errores:**

Cuando nuestro código se ejecuta y tiene un error, podemos entrar en una fase donde se puede entender mejor qué está sucediendo con la aplicación.

Algo que debemos tener en cuenta es que un componente NO debe pasar por toda las fases, un componente puede ser montado y desmontado sin pasar por la fase de actualización o manejo de errores.

Ahora que entendemos las fases que cumple el ciclo de vida en React vamos a entrar a detalle en cada uno de ellos para ver qué piezas de código se ejecutan y nos ayudarán a crear aplicaciones en React pasando por un ciclo de vida bien estructurado.

## **Montado:**

**Constructor()**

Este es el primer método al que se hace un llamado, aquí es donde se inicializan los métodos controladores, eventos del estado.

**getDerivedStateFromProps()**

Este método se llama antes de presentarse en el DOM y nos permite actualizar el estado interno en respuesta a un cambio en las propiedades, es considerado un método de cuidado, ya que su implementación puede causar errores sutiles.

**render()**

Si queremos representar elementos en el DOM en este método es donde se escribe esta lógica, usualmente utilizamos JSX para trabajar y presentar nuestra aplicación.

**ComponentDidMount()**

Este método se llama inmediatamente que ha sido montado en el DOM, aquí es donde trabajamos con eventos que permitan interactuar con nuestro componente.

## **Actualización:**

**getDerivedStateFromProps()**

Este método es el primero en ejecutarse en la fase de actualización y funciona de la misma forma que en el montaje.

**shouldComponentUpdate()**

Dentro de este método se puede controlar la fase de actualización, podemos devolver un valor entre verdadero o falso si queremos actualizar o no el componente y es utilizado principalmente para optimización.

**render()**

Se llama el método render que representa los cambios en el DOM.

**componentDidUpdate()**

Este método es invocado inmediatamente después de que el componente se actualiza y recibe como argumentos las propiedades y el estado y es donde podemos manejar nuestro componente.

## **Desmontado**

**componentWillUnmount()**

Este método se llama justo antes de que el componente sea destruido o eliminado del DOM.

## **Manejo de Errores:**

**getDerivedStateFromError()**

Una vez que se lanza un error este es el primer método que se llama, el cual recibe el error como argumento y cualquier valor devuelto en este método es utilizado para actualizar el estado del componente.

**componentDidCatch()**

Este método es llamado después de lanzarse un error y pasa como argumento el error y la información representada sobre el error.

Ahora que entendemos cada una de las fases que tiene el ciclo de vida de react, podemos utilizarlas según sea necesario en nuestra aplicación y de esta forma crear las interacciones que necesitemos.