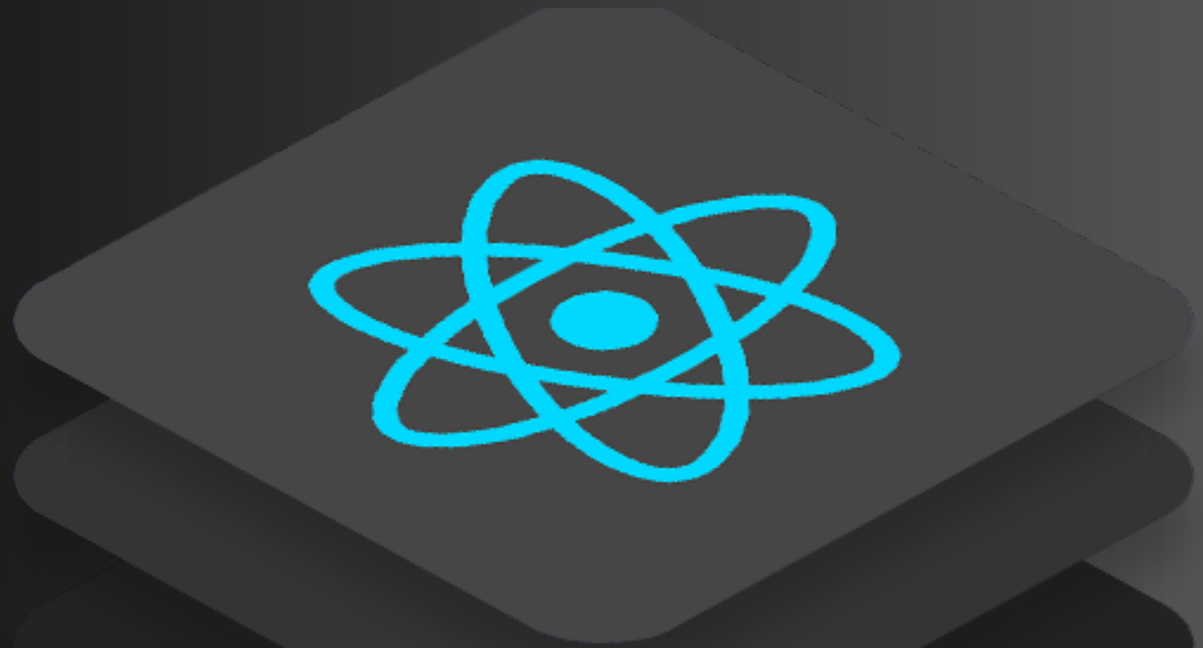


# React

## Frontend

# Hooks

O que são hooks?
Funções introduzidas a partir da versão 16.8 que permitem utilizar os recursos de estado e o ciclo de vida do componente a partir de componentes funcionais e não mais de componentes de classe.

## Antes dos Hooks:

- Desafios no gerenciamento de estado e efeitos colaterais em componentes funcionais.
- Limitações de componentes de função sem estado interno ou métodos de ciclo de vida.

## Problemas Anteriores:

- Lógica de estado e efeitos frequentemente tratados em componentes de classe.
- Uso extensivo de abstrações complicadas, como HOCs e Render Props.

## Vantagens dos Hooks:

- Abordagem mais declarativa e modular.
- Simplifica o código, tornando-o mais legível e fácil de manter.
- Promove a composição, permitindo a criação de funcionalidades complexas a partir de Hooks especializados.

### Hooks em Ação:

- useState: Introduz estados locais em componentes funcionais.
- useEffect: Facilita a execução de código relacionado a efeitos colaterais.

# Ciclo de Vida do Componente

**Montagem (Mounting):**

- Conceito Geral: Esta é a fase inicial quando o componente está sendo criado e inserido no DOM.

- Contexto do useState: O useState é frequentemente utilizado na fase de montagem para inicializar o estado do componente. O estado pode ser definido com um valor inicial durante a criação do componente.

# Ciclo de Vida do Componente

### Atualização (Updating):

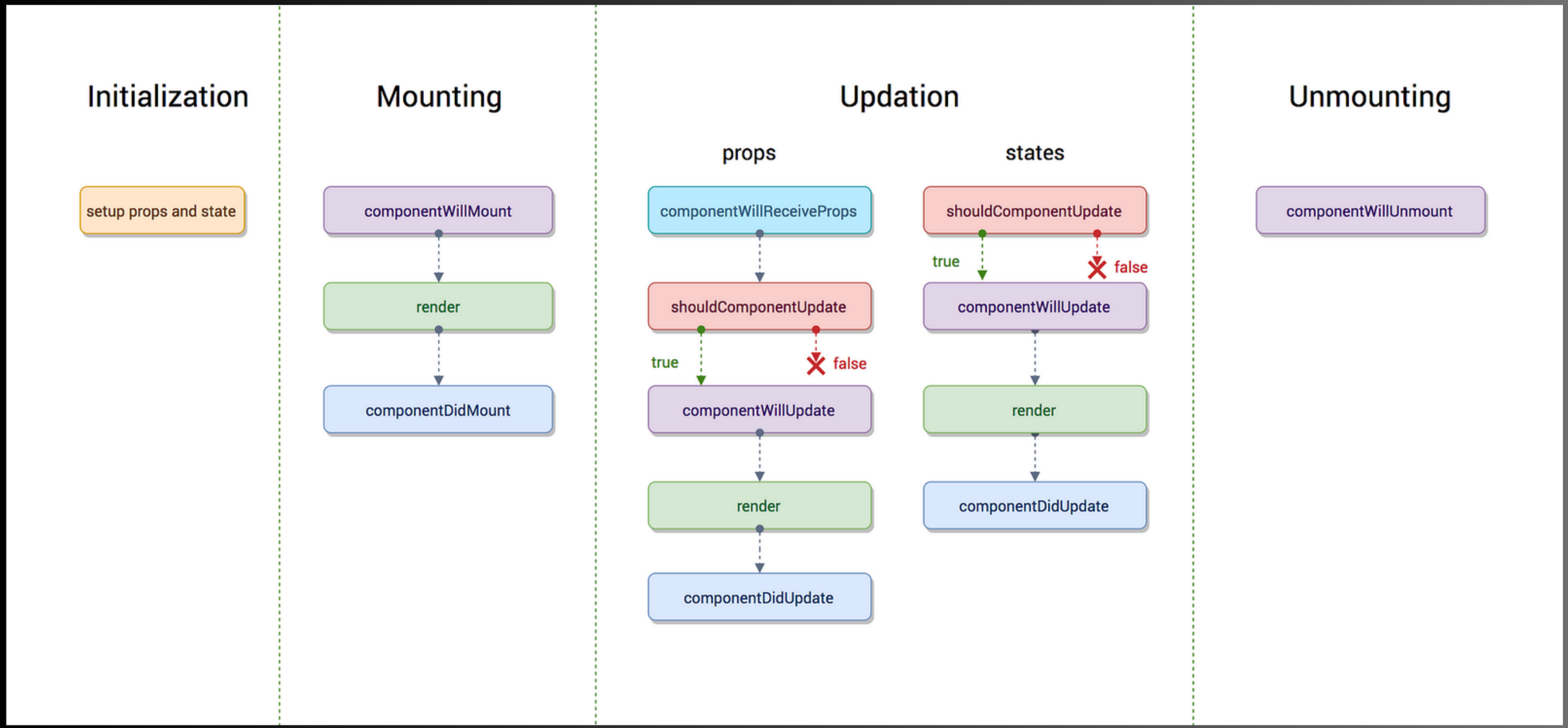
- Conceito Geral: Ocorre quando o componente é re-renderizado devido a mudanças no estado ou nas propriedades.

- Contexto do useState: Durante a fase de atualização, o useState é usado para modificar o estado do componente, desencadeando uma nova renderização. Ao chamar a função de atualização retornada pelo useState, o React entende que o estado está mudando e re-renderiza o componente.

### Desmontagem (Unmounting):

- Conceito Geral: A fase final, quando o componente está sendo removido do DOM.

# Ciclo de Vida do Componente



# Manipulação de Estado

## useState

Permite adicionar uma variável de estado no componente

```
import { useState } from "react";

function MeuComponente(){
    const [ state, setState]= useState("Valor inicial")


    return(
        <div>

        </div>
    )
}
```

# useState

```jsx
import { useState } from "react";

function MeuComponente(){
    const [ state, setState]= useState("Valor inicial")


    return(
        <div>

        </div>
    )
}
```

- Por convenção, costuma-se nomear o primeiro item com uma palavra para o estado e o segundo item com set o nome do estado em camelCase. Como pode ser visto na figura.

O useState retorna um array com o primeiro item sendo o estado e o segundo item uma função que modifica esse estado.

# useEffect

- Hook que permite sincronizar um componente com um sistema externa

```
import { useEffect } from "react";

function MeuComponente(){
  const [isAuth, setIsAuth] = useState(false)
  useEffect(()=>{
    //Aqui você coloca as ações que deseja realizar
    return()=>{
      //Aqui você pode limpar alguma ação ao final do useEffect
    }
  },[isAuth])

  return
  <h1> Meu componente</h1>
}
```

- Parâmetros

- Função de setup: Função que emprega a lógica aplicada no useEffect. Essa função pode opcionalmente retorna uma função de “limpeza” das ações.

- Lista de dependências(Opcional): O useEffect pode ter uma lista de itens reativos como lista de dependências que servirão de gatilho para ativação do useEffect

## Eventos

Como visto anteriormente no módulo de Javascript, os eventos são ações ou ocorrências do sistema que podem ser provocados pelos usuários .

Os desenvolvedores do React listaram os principais eventos encontrados no DOM e inseriram-nos no DOM virtual do React. Todos os eventos do React utilizam o camelCase. Exemplos: “onChange, onClick, onSubmit, onDrag etc”.

## onClick

Utilizado principalmente na tag button, esse evento é utilizado para disparar as ações.

## onChange

Encontrado principalmente nas tags de input, esse evento auxilia na mudança dos valores encontrados no valor do input.

## onSubmit

Utilizado principalmente nas ações de envio do formulário

# Vamos à prática