DigitalHouse>

Certified Tech Developer
The Ultimate Degree

Front End III

# Class Component Vs Function Component

Vamos a repasar el paso a paso de cómo se estructura un pedido a una API y las actualizaciones que puede tener el componente, tanto con una class como con una function component.

```
import React, { Component } from 'react'
import API from "./API"
import "./Global.css"

export default class ClassComponent extends Component {
   componentDidMount() {
       // Se monta por primera vez
     }

   render() {
       return (
           <div className="box">
               <h1>CLASS COMPONENT</h1>
           </div>
       )
   }
}
```

```
import React, {useEffect} from 'react'
import API from "./API"
import "./Global.css"

const HookComponent = () => {
    useEffect(() => {
        // Se monta por primera vez
    }, [])

    return (
        <div className="box">
              <h1>FUNCTION COMPONENT</h1>
        </div>
    )
}

export default HookComponent
```

1. El primer caso que vemos es una visualización inicial de cómo se verá cada componente. En ambos casos, se ejecuta una única vez.

```
import React, { Component } from 'react'
import API from "./API"
import "./Global.css"
import axios from 'axios'

export default class ClassComponent extends Component {
    constructor(){
        super();
        this.state = {name:"",id:1}
    }
    componentDidMount() {
        const fetchData = async () => {
            const result = await axios(`${API}/${this.state.id}`);
            this.setState({name: result.data.name});
        };
        fetchData();
```

```
    }

    render() {
        return (
            <div className="box">
                <h1>CLASS COMPONENT</h1>
                <h2>{this.state.name}</h2>
            </div>
        )
    }
}
```

```
import React, { useEffect, useState } from 'react'
import API from "./API"
import "./Global.css"
import axios from 'axios'

const HookComponent = () => {
    const [name, setName] = useState("");
    const [id, setId] = useState(1);

    useEffect(() => {
        const fetchData = async () => {
            const result = await axios(`${API}/${id}`);
            setName(result.data.name);
        };
        fetchData();
    }, [])

    return (
        <div className="box">
            <h1>FUNCTION COMPONENT</h1>
            <h2>{name}</h2>
        </div>
    )
}
export default HookComponent
```

En ambos casos, nos trae la información de la API por única vez. Podemos notar que en el hook ya no hacemos uso del this y utilizamos unos hooks denominados useEffect y useState.

**CLASS COMPONENT**

**Luke Skywalker**

**FUNCTION COMPONENT**

**Luke Skywalker**

**2.** Ahora veamos cómo sería para actualizar la data de la API.

```
import React, { Component } from 'react'
import API from "./API"
import "./Global.css"
import axios from 'axios'

export default class ClassComponent extends Component {
   constructor(){
       super();
       this.state = {name:"",id:1}
   }

   fetchData = async () => {
       const result = await axios(`${API}/${this.state.id}`);
       this.setState({name: result.data.name});
   };

   updateData = () => {this.setState({id: this.state.id + 1})};

   componentDidMount() {this.fetchData()};

   componentDidUpdate(prevProps, prevState){
       if (this.state.id !== prevState.id) this.fetchData();
   }

   render() {
       return (
           <div className="box">
```

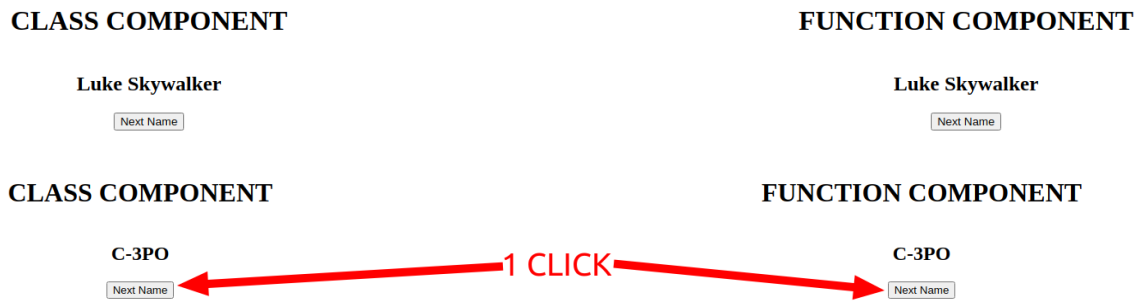
```
                <h1>CLASS COMPONENT</h1>
                <h2>{this.state.name}</h2>
                <button onClick={this.updateData}>Next Name</button>
            </div>
        )
    }
}
```

Agregamos un botón que nos incrementa el id en 1, movemos las funciones fuera para reutilizarlas y vemos que el componentDidUpdate, comprueba si hubo cambios en el estado.

```
import React, { useEffect, useState } from 'react'
import API from "./API"
import "./Global.css"
import axios from 'axios'

const HookComponent = () => {
    const [name, setName] = useState("");
    const [id, setId] = useState(1);

    const fetchData = async () => {
        const result = await axios(`${API}/${id}`);
        setName(result.data.name);
    };

    const updateData = () => {setId(id + 1)};

    useEffect(() => {fetchData()}, []);

    useEffect(() => {fetchData();}, [id]); // Escucha cambios en el state ID

    return (
        <div className="box">
            <h1>FUNCTION COMPONENT</h1>
```

```
            <h2>{name}</h2>
            <button onClick={updateData}>Next Name</button>
        </div>
    )
}

export default HookComponent
```

Con el hook, hacemos exactamente lo mismo. Tenemos el botón y las funciones por fuera, pero podemos notar cómo la sintaxis es menor a lo que teníamos en classComponent.

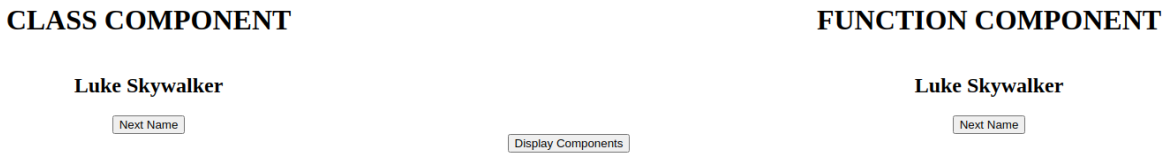



3. El último caso es para desmontar un componente. Para esto, primero vamos a modificar el App.js para que tenga un botón para desmontar el componente.

```
import React, {useState} from "react";
import ClassComponent from "./ClassComponent";
import HookComponent from "./HookComponent";
import "./Global.css"

function App() {
 const [display, setDisplay] = useState(true);
 return (
   <div className="columns">
   <div className="container">
     {display ? (
         <>
         <ClassComponent />
         <HookComponent />
```

```
          </>
        ) : <p>UNMOUNTED</p>
      }
    </div>
    <button onClick={() => setDisplay(!display)}>Display Components</button>
    </div>
  );
}

export default App;
```

Esto se vería así:




Y por último, modificamos los componentes:

```
import React, { Component } from 'react'
import API from "./API"
import "./Global.css"
import axios from 'axios'

export default class ClassComponent extends Component {
    constructor(){
        super();
        this.state = {name:"",id:1}
    }

    fetchData = async () => {
        const result = await axios(`${API}/${this.state.id}`);
        this.setState({name: result.data.name});
    };
```

```
    updateData = () => {this.setState({id: this.state.id + 1})};

    componentDidMount() {this.fetchData()};

    componentDidUpdate(prevProps, prevState){
        if (this.state.id !== prevState.id) this.fetchData();
    }

    componentWillUnmount() {
        alert("Component Class se va desmontar")
    }

    render() {
        return (
            <div className="box">
                <h1>CLASS COMPONENT</h1>
                <h2>{this.state.name}</h2>
                <button onClick={this.updateData}>
                    Next Name
                </button>
            </div>
        )
    }
}
```

```
import React, { useEffect, useState } from 'react'
import API from "./API"
import "./Global.css"
import axios from 'axios'

const HookComponent = () => {
    const [name, setName] = useState("");
    const [id, setId] = useState(1);

    const fetchData = async () => {
        const result = await axios(`${API}/${id}`);
        setName(result.data.name);
    };

    const updateData = () => {setId(id + 1)};

    // Obtengo la data por primera vez
    useEffect(() => {fetchData()}, []);

    // Se actualiza por cada cambio
    useEffect(() => {fetchData()}, [id]);

    // Uso la Clean function para desmontar
    useEffect(() => {return () => {
            alert("Component Hook se va desmontar")
        }
    }, []);

    return (
        <div className="box">
            <h1>FUNCTION COMPONENT</h1>
            <h2>{name}</h2>
            <button onClick={updateData}>Next Name</button>
        </div>
    )
}
export default HookComponent
```

Nuestro ejemplo finalizará de esta manera: