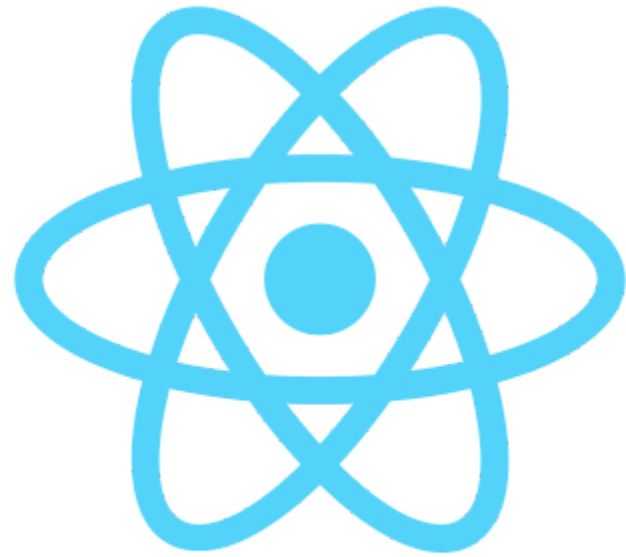


HENRY

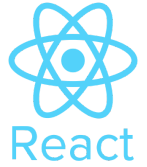
A bright yellow beam of light originates from the left edge of the frame and points towards the letter 'R' in the word 'HENRY'. The beam is wider on the left and tapers as it moves to the right. The word 'HENRY' is written in a bold, black, sans-serif font.



React

Estados y LifeCycles

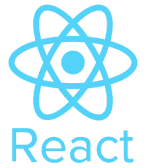
Estados



Las props son la **configuración inicial del componente**, son los datos con los que van a ser renderizados.

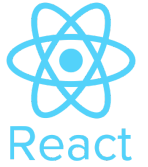
Pero la vida de un componente no termina ahí. De hecho, cada componente puede tener un **estado**. Podremos acceder al estado de cada componente a través del objeto en `this.state`.

Estados



```
1
2 class Contador extends React.Component {
3   constructor(props) {
4     super(props);
5     this.state = {contador: this.props.contador}
6     this.onButtonClick = this.onButtonClick.bind(this);
7   }
8   onButtonClick(e){
9     e.preventDefault();
10    this.setState({
11      contador: this.state.contador + 1,
12    });
13  }
14  render(){
15    return (
16      <div>
17        <button onClick={this.onButtonClick}>Suma uno!</button>
18        <p>Hola, van {this.state.contador}!!</p>
19      </div>
20    )
21  }
22  };
```

Estados

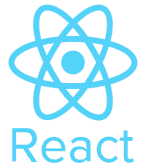


setState no siempre actualiza inmediatamente, por lo que habría que evitar leer this.state luego de haber utilizado this.setState

- Utilizar el lifecycle componentDidMount
- Agregar una función de callback al setState

```
1  ...
2
3  handleChange(event) {
4    this.setState({ username: event.target.username }, function() {
5      this.validateUsername();
6    });
7  }
```

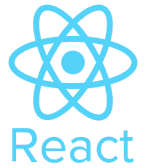
Estados



Si usamos `setState` dos veces de forma consecutiva, el valor del estado siempre va a apuntar al que tenia al momento del inicio del contexto de ejecucion de la funcion en el cual se ejecuta:

```
1 ...
2 this.state = {
3   contador: this.props.numero
4 } //valor inicial 1
5 ...
6 function increment() {
7   this.setState({
8     contador: this.state.contador + 1
9   })
10  this.setState({
11    contador: this.state.contador + 1
12  })
13 }
14 increment() //2
15 increment() //3
```

Estados

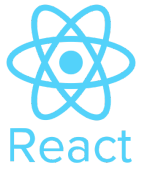


Podemos utilizar una función. Su primer parámetro va a ser el estado y el segundo las props.

Como los contextos de ejecución se ejecutan uno después del otro, el valor va a ser distinto en cada ejecución.

```
1 ...
2 this.state = {
3   contador: this.props.numero
4 } //valor inicial 1
5 ...
6 function increment() {
7   this.setState((state, props) => contador: state.contador + 1)
8   this.setState((state, props) => contador: state.contador + 1)
9 }
10 increment() //3
11 increment() //5
```

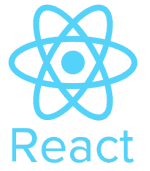
Estados



```
1 class Children extends React.Component {
2   constructor(props) {
3     super(props);
4     this.state = {
5       title: this.props.title
6     };
7   }
8
9   render(){
10    return (
11      <div>
12        <h2>{this.state.title}</h2>
13      </div>
14    )
15  }
16 };
```

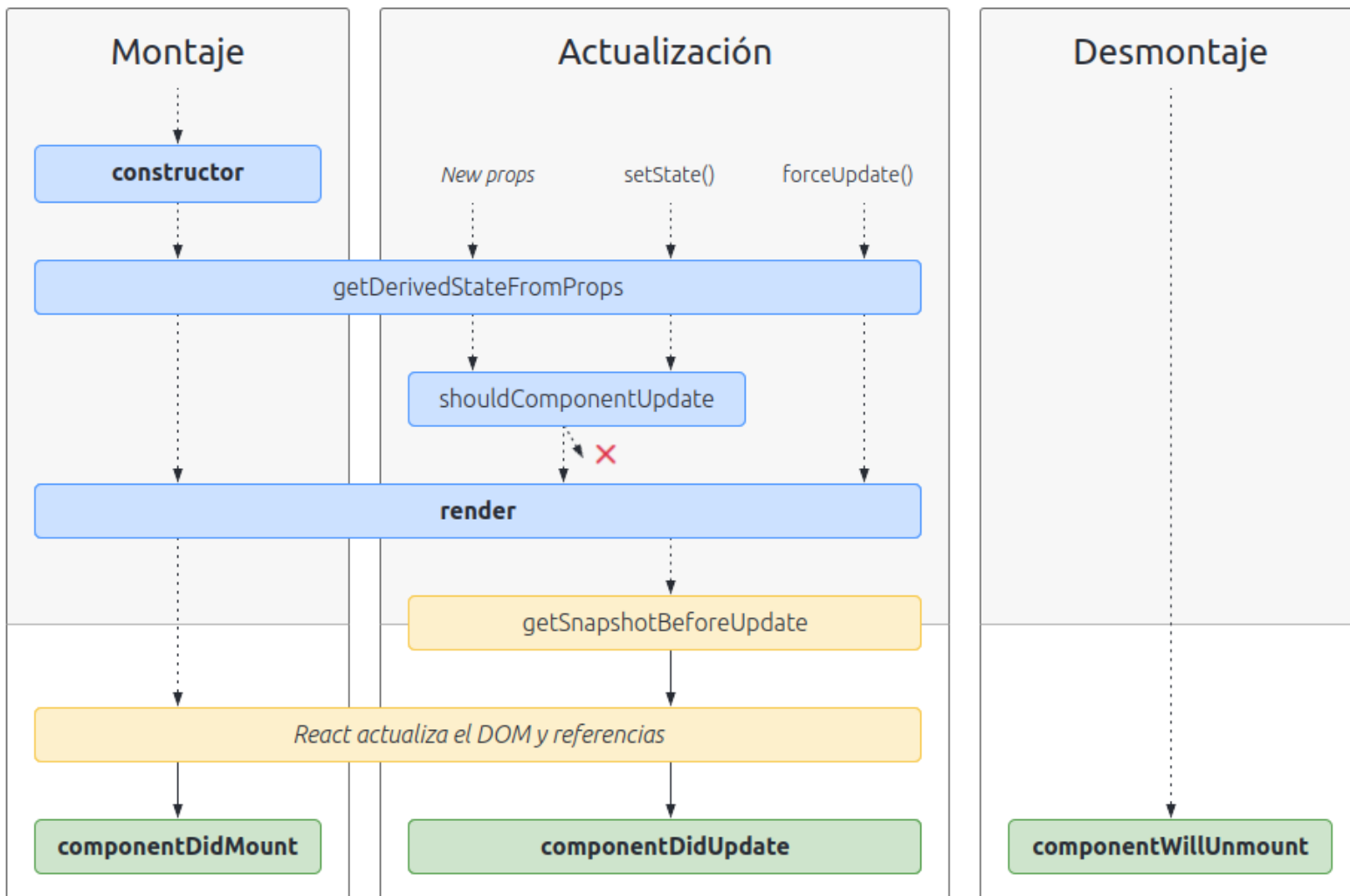


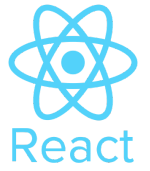
<Demo copyState/>



<Demo />

Ciclo de Vida

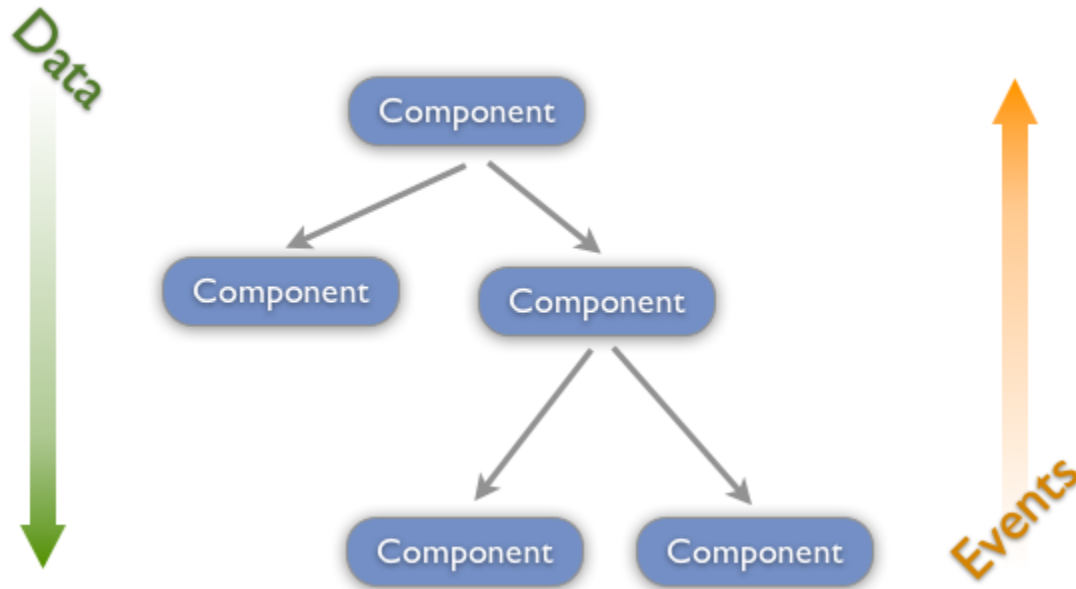


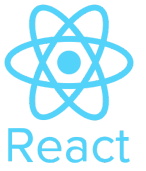


<Demo />

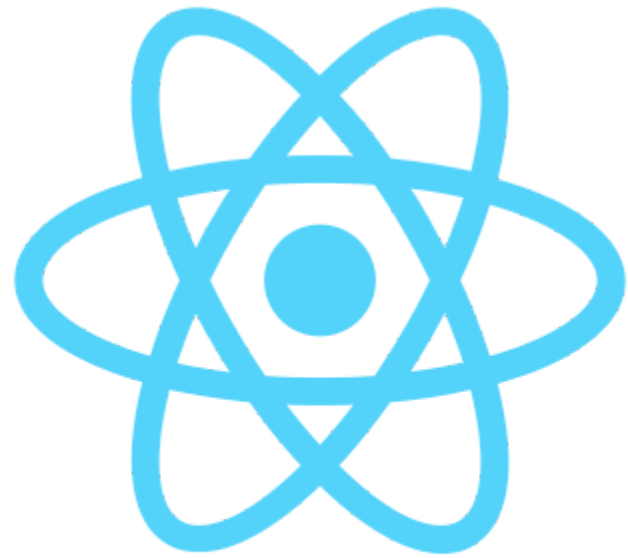
One Way Data Flow

Components y Containers





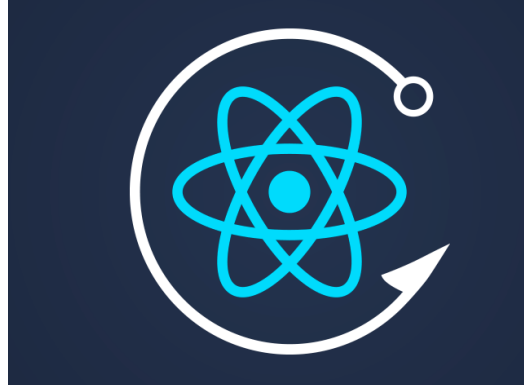
<Demo />



React

Hooks

Hooks



Los Hooks son una nueva API de la librería de React que nos permite tener estado, y otras características de React, en los componentes creados con una function. Esto, antes, no era posible y nos obligaba a crear un componente con class.



Hooks

useState

Devuelve un valor con estado y una función para actualizarlo.

```
1  
2  
3 const [state, setState] = useState(initialState);
```


Hooks

```
1  const { useState } = React;
2
3  function MuestraCuenta(props) {
4    return (
5      <p>Hola, van {props.contador}!!</p>
6    );
7  }
8
9  function Contador(props) {
10   const [contador, setContador] = useState(props.contador);
11
12   const onClick = () => {
13     setContador(contador + 1)
14   }
15
16   return (
17     <div>
18       <button onClick={onClick}>Suma uno!</button>
19       <MuestraCuenta contador={contador} />
20     </div>
21   );
22 };
```

<https://es.reactjs.org/docs/hooks-reference.html>