El concepto de React State es uno de los conceptos más importantes de React. Todo componente de React tiene asignadas unas propiedades y un estado . Cada vez que el estado de un componente cambia el componente se vuelve a renderizar. Vamos a construir un ejemplo de manejo de estado.

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8" />
    <title>Hello World</title>
    <script
src="https://unpkg.com/react@latest/dist/react.js"></script>
    <script
src="https://unpkg.com/react-dom@latest/dist/react-dom.js"></script>
    <script
src="https://unpkg.com/babel-standalone@6.15.0/babel.min.js"></script>
  </head>
  <body>
<div id="zona">
</div>
    <script type="text/babel" src="001.js">
    </script>

  </body>
</html>
```

El primer paso es crear un componente en React:

```
     var Numero = React.createClass({
```

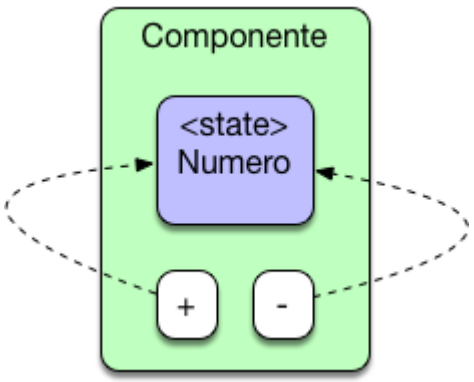
```
        getInitialState:function() {

           return {numero:0};
        },
        mas: function() {

          this.setState({numero:this.state.numero+1});

        },menos: function() {

          this.setState({numero:this.state.numero-1});

        },
        render: function() {
            return (
<div>
                {this.state.numero}
                <input type="button" value="+" onClick={this.mas}/>
                <input type="button" value="-" onClick={this.menos}/>
              </div>


            );
        }
    });

    ReactDOM.render(
<div>
           <Numero/>
           </div>
```

```
,document.getElementById('zona'));
```

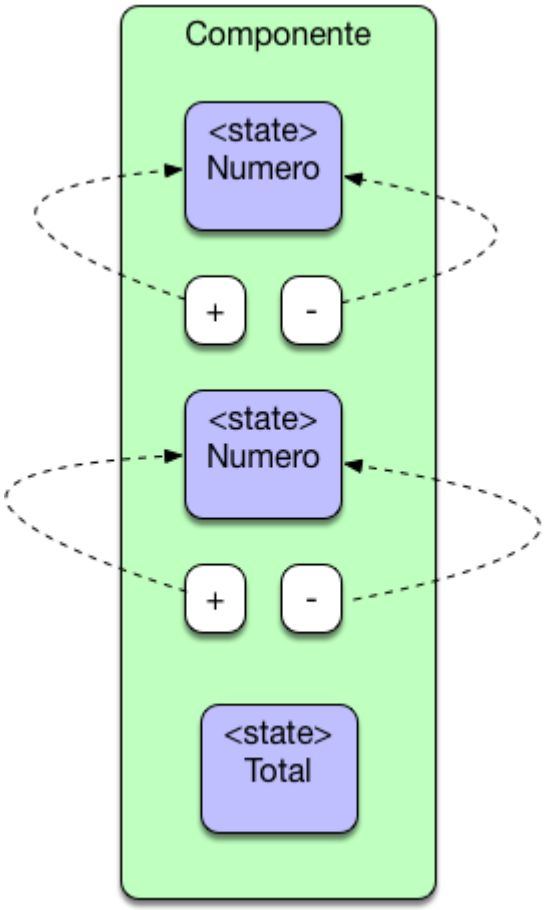
React State

En este ejemplo construimos un componente que almacena un estado (React State) . Este estado nos permite incrementan o decrementar el número a través de dos botones.

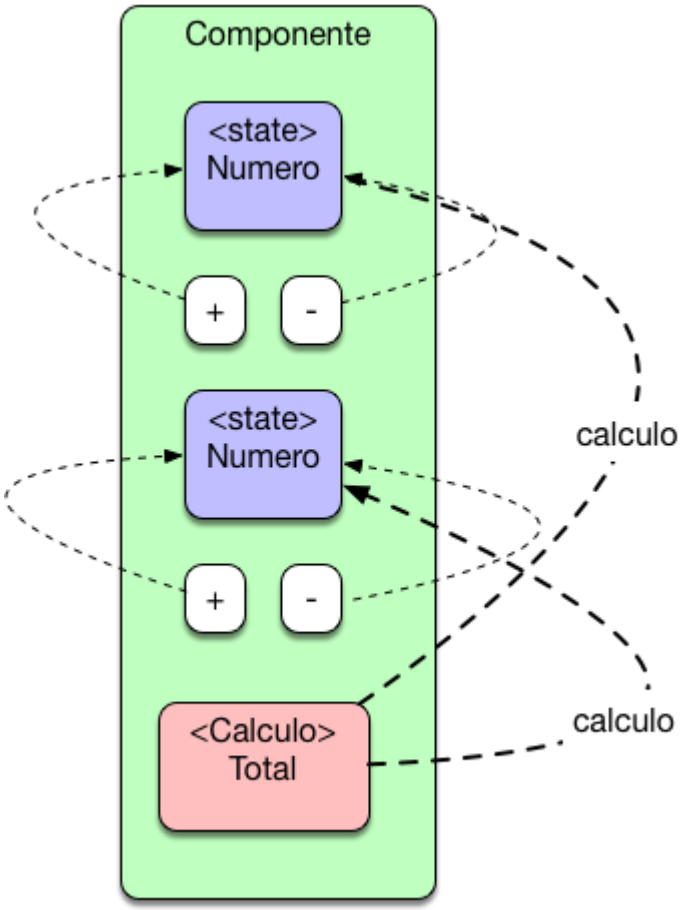
Para ello hemos definido dos botones que incrementan o decrementan el estado que el componente almacena.



Vamos a construir otro componente que disponga de dos números independientes a los cuales les podemos modificar su valor y contenga también un total que sume los valores de ambos botones. Muchas personas piensan en un primer momento que el nuevo estado del componente contendrá tres variables , numero1, numero2 y total.



Sin embargo esto no es así ya que el total se calcula como la suma de numero1 y numero2 por lo tanto no es parte del estado sino un valor calculado.



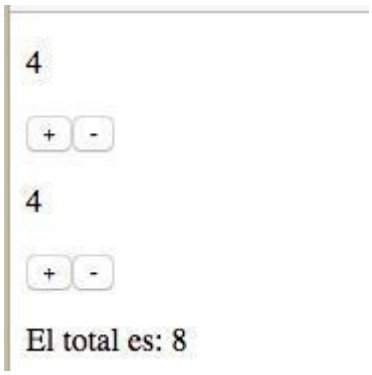
Vamos a ver una primera aproximación:

```
var Numero = React.createClass({

      getInitialState:function() {

         return {numero1:0,numero2:0};
      },
      mas1: function() {
```

```
        this.setState({numero1:this.state.numero1+1});

      },menos1: function() {

        this.setState({numero1:this.state.numero1-1});

      },
      mas2: function() {

        this.setState({numero2:this.state.numero2+1});

      },
      menos2: function() {

        this.setState({numero2:this.state.numero2-1});

      }
      ,
      render: function() {
          return (
            <div>
              {this.state.numero1}
              <input type="button" value="+" onClick={this.mas1}/>
              <input type="button" value="-" onClick={this.menos1}/>
                {this.state.numero2}
                <input type="button" value="+" onClick={this.mas2}/>
                <input type="button" value="-"
onClick={this.menos2}/>
                  El total es:
{this.state.numero1+this.state.numero2}
            </div>
```

```
            );
        }
    });
    ReactDOM.render(
<div>
<Numero/>
</div>
,document.getElementById('zona'));
```

Aquí podemos ver como el componente tiene dos variables a nivel de estado pero , el total es un campo calculado por lo tanto no hace falta que le definamos a nivel de state.
Simplemente hacemos el cálculo a la hora de renderizarse.



El código del componente se puede simplificar utilizando funciones que admitan parámetros apoyándonos en el método bind().

```
    var Numero = React.createClass({

        getInitialState:function() {
```

```
           return {numero1:0,numero2:0};
        },
        operacion: function(numero,i) {

        var objeto={};
        objeto[numero]=this.state[numero]+i;
        this.setState(objeto);

        },


        render: function() {
            return (
                <div>
                {this.state.numero1}
                <input type="button" value="+"
onClick={this.operacion.bind(this, "numero1",1)}/>
                <input type="button" value="-"
onClick={this.operacion.bind(this, "numero1",-1)}/>
                {this.state.numero2}
                <input type="button" value="+"
onClick={this.operacion.bind(this, "numero2",1)}/>
                <input type="button" value="-"
onClick={this.operacion.bind(this, "numero2",-1)}/>
                El total es: {this.state.numero1+this.state.numero2}
              </div>
            );
        }
    });
    ReactDOM.render(
            <div>
```

```
            <Numero/>
            </div>
,document.getElementById('zona'));
```

El resultado será el mismo. El concepto de React State siempre hace referencia al mínimo estado posible que el componente debe almacenar. En este caso el total no pertenece al estado.

Otros artículos relacionados:

1. Desarrollo Web con React.js , mi nuevo libro
2. React vs Angular 2 , frameworks vs librerias
3. Sublime React y el uso de JSX
4. Introducción a React.js
5. RxJS y la programación reactiva.
6. React FaceBook