

React JSX

“

JSX es una extensión de la sintaxis de JavaScript.

Se usa para describir cómo debería ser la interfaz de usuario.

JSX puede recordarte a un lenguaje de plantillas pero viene con todo el poder de JavaScript.



REACT

Insertando expresiones

En el ejemplo a continuación, declaramos una variable llamada **name** y luego la usamos dentro del JSX envolviéndola **dentro de llaves**:

{ }

```
const name = 'John Doe';  
const element = <h1>Hello, {name}</h1>;  
  
ReactDOM.render(  
  element,  
  document.getElementById('root')  
)
```

REACT

Insertando expresiones

Se puede colocar cualquier expresión de JavaScript dentro de llaves en JSX. Por ejemplo, `2 + 2`, `user.firstName`, o `formatName(user)` son todas expresiones válidas de Javascript.

En el ejemplo a continuación, insertamos el resultado de llamar la función de JavaScript, `formatName(user)`, dentro de un elemento `<h1>`.

REACT

Insertando expresiones

{ }

```
const user = {  
  firstName: 'John',  
  lastName: 'Doe',  
  formatName: function () {  
    return this.firstName + ' ' + this.lastName;  
  }  
}
```

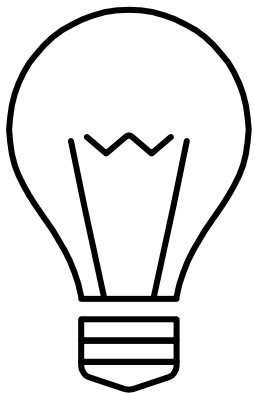
```
const element = (  
  <h1>  
    Hello, {user.formatName()}!  
  </h1>  
);
```

```
ReactDOM.render(  
  element,  
  document.getElementById('root')  
);
```

¡ ATENCIÓN !



Dado que JSX es más cercano a JavaScript que a HTML, React DOM usa la convención de nomenclatura camelCase en vez de nombres de atributos HTML. Por ejemplo, **class** se vuelve **className** en JSX, y **tabindex** se vuelve **tabIndex**.



```
{ }
```

```
const element = (  
  <h1 className="greeting">  
    Hola, Mundo!!!  
  </h1>  
);
```

¡ ATENCIÓN !



Si una etiqueta está vacía, puedes cerrarla inmediatamente con `</>`

`{ }`

```
const element = <img src={user.avatarUrl} />;
```



REACT JSX

Para saber más puedes acceder a la documentación oficial de Sequelize haciendo click en el siguiente [Link](#)

