DigitalHouse>

Certified Tech Developer

The Ultimate Degree

Front End III

Otros eventos: onSubmit y onChange

Juguemos un poco con un form simple con validación básica y la ejecución de una acción luego de enviar los datos.

```
import React, { Component } from "react";
import Swal from 'sweetalert2'

import './styles.module.css';

const App = () =>  <MyLittleForm />;

class MyLittleForm extends Component {
   state = { name: "", error: false };

    handleChange = (event) => {
      this.setState({
           name: event.target.value,
           error: false
       });
       console.log('Has cambiado el nombre');
    }

    handleSubmit = (event) => {
       event.preventDefault();

       if(this.state.name === "")
           this.setState({ error: true});
       else {
           Swal.fire('Hello ' + this.state.name);
           this.setState({ name: ''});
       }
    }
```

```
    render() {
      return (
          <form onSubmit={this.handleSubmit}>
              <img src="/img/pony.jpg" alt="Pony" />
              <label>Name</label>
              <input
                  type="text"
                  value={this.state.name}
                  onChange={this.handleChange}
              />
              {this.state.error && <span>This field is required</span>}
              <button type="submit">Submit!</button>
          </form>
      );
   }
}

export default App;
```

Aquí tenemos, en el componente MyLittleForm, la renderización de un formulario.

```
      return (
          <form onSubmit={this.handleSubmit}>
              <img src="/img/pony.jpg" alt="Pony" />
              <label>Name</label>
              <input
                  type="text"
                  value={this.state.name}
                  onChange={this.handleChange}
              />
              {this.state.error && <span>This field is required</span>}
              <button type="submit">Submit!</button>
          </form>
      );
```

Respecto al estado local tenemos:

```
state = { name: "", error: false };
```

El componente, además, contiene dos eventos con sus correspondientes manejadores:

handleChange

Se encarga de actualizar name. Esto permite que podamos ver lo que vamos escribiendo en el <input>. Además, en este caso el manejador escribirá un mensaje en consola cada vez que el valor en el input cambie.

```
    handleChange = (event) => {
      this.setState({
           name: event.target.value,
           error: false
       });
       console.log('Has cambiado el nombre');
    }
```

handleSubmit

Es el manejador que controla lo que ocurre cuando el evento onSubmit es disparado. Si nosotros damos submit a una etiqueta form, por más que el mismo esté controlado por React, se impondrá el comportamiento del evento del DOM. Es por esto que hacemos un event.preventDefault(). Esto evitará que se refresque la pantalla del navegador.

```
handleSubmit = (event) => {
    event.preventDefault();
}
```

Validación

En este caso realizaremos una pequeña validación para que, al dar submit, el input tenga que tener algún caracter, de lo contrario se mostrará un mensaje de error.

```
handleSubmit = (event) => {
       event.preventDefault();

       if(this.state.name === "")
           this.setState({ error: true});
}
```

```
return ({this.state.error && <span>This field is required</span>})
```

Finalmente, de pasar la validación, el animalito nos dará una tierna bienvenida.

```
   handleSubmit = (event) => {
       event.preventDefault();

       if(this.state.name === "")
           this.setState({ error: true});
       else {
           Swal.fire('Hello ' + this.state.name);
           this.setState({ name: ''});
       }
    }
```

¡Hasta la próxima!