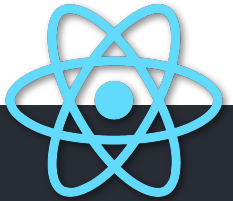

React

Contenido

- Conceptos Generales
- JSX
- Hello World
- Componentes y propiedades
- Estado y ciclo de vida
- Manejando eventos
- Formularios
- Hooks
- Build

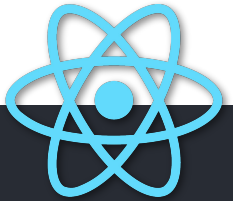


Conceptos Generales

ReactJS es una biblioteca Javascript de código abierto diseñada para crear interfaces de usuario con el objetivo de facilitar el desarrollo de aplicaciones en una sola página.

> React es una librería de JavaScript declarativa, eficiente y flexible para construir interfaces de usuario. Permite componer IUs complejas de pequeñas y aisladas piezas de código llamadas “componentes”.

URL: https://es.reactjs.org/



JSX

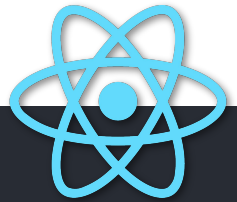
JSX es una extensión de la sintaxis de JavaScript. Más allá que parezca un híbrido entre HTML y Javascript, viene con todo el poder de JavaScript. No es necesaria para utilizar React, sin embargo, hace el código más legible, y escribirlo es una experiencia similar a HTML.

```
function formatName(user) {
  return user.firstName + ' ' + user.lastName;
}

const user = {
  firstName: 'Harper',
  lastName: 'Perez'
};

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

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

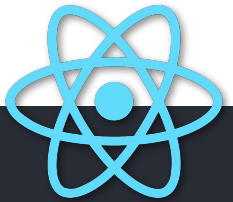


Hello World

Vamos a crear nuestra primer aplicación

```
npx create-react-app my-app
cd my-app
npm start
```

Repo Template: https://github.com/ElDwarf/React-project-vs-template



Componentes y propiedades

Los componentes permiten separar la interfaz de usuario en piezas independientes, reutilizables y pensar en cada pieza de forma aislada. Esta página proporciona una introducción a la idea de los componentes. (Doc).

Funcionales

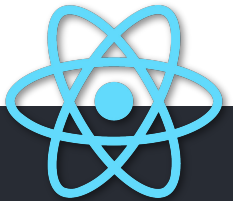
```
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}
```

De clase

```
class Welcome extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}</h1>;
  }
}
```

Uso o llamada al componente

```
const element = <Welcome name="Sara" />;
```



Estado y ciclo de vida

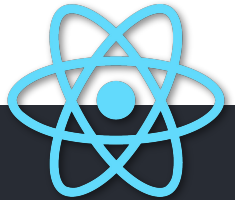
Como toda biblioteca de UI sus componentes tiene estados que permiten gestionar en el tiempo el estado del mismo y un ciclo vida que nos va a permitir agregar funcionen en ciertos eventos (Doc Componentes, Doc Ciclo vida)

```
class Clock extends React.Component {
  constructor(props) {
    super(props);
    this.state = {date: new Date()};
  }

  componentDidMount() {}

  componentWillUnmount() {}

  render() {
    return (
      <div>
        <h1>Hello, world!</h1>
        <h2>It is {this.state.date.toLocaleTimeString()}.</h2>
      </div>
    );
  }
}
```



Manejando eventos

Manejar eventos en elementos de React es muy similar a manejar eventos con elementos del DOM. Hay algunas diferencias de sintaxis:

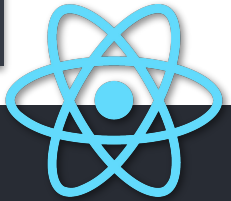
- Los eventos de React se nombran usando camelCase, en vez de minúsculas.
- Con JSX pasas una función como el manejador del evento, en vez de un string.

HTML

```
<button onclick="activateLasers()">
  Activate Lasers
</button>
```

React

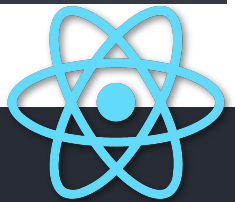
```
<button onClick={activateLasers}>
 Activate Lasers
</button>
```



Formularios

Los elementos de formularios en HTML funcionan un poco diferente a otros elementos del DOM en React, debido a que los elementos de formularios conservan naturalmente algún estado interno.

```
class NameForm extends React.Component {
  constructor(props) {
    super(props);
    this.state = {value: ''};
    this.handleChange = this.handleChange.bind(this);
    this.handleSubmit = this.handleSubmit.bind(this);
  }
  handleChange(event) {
   this.setState({value: event.target.value});
  }
  handleSubmit(event) {
    alert('A name was submitted: ' + this.state.value);
    event.preventDefault();
  }
  render() {
    return (
     <form onSubmit={this.handleSubmit}>
       <label>Name:
       <input type="text" value={this.state.value} onChange={this.handleChange} />
       </label>
       <input type="submit" value="Submit" />
      </form>
    );
  }
}
```

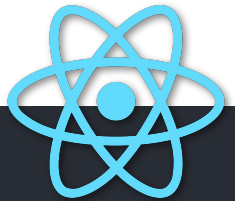


Hooks

Hooks son una nueva característica en React 16.8. Estos te permiten usar el estado y otras características de React sin escribir una clase.

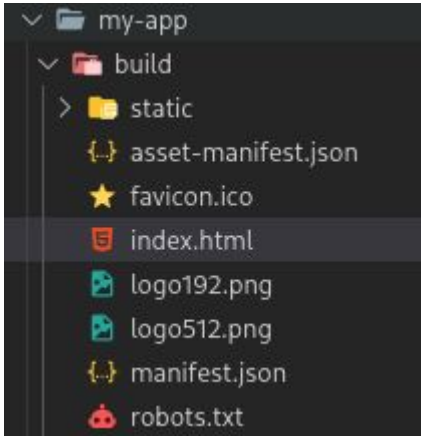
```
import React, { useState } from 'react';

function Example() {
 // Declara una nueva variable de estado, la cual llamaremos “count”
 const [count, setCount] = useState(0);
 return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>
        Click me
      </button>
    </div>
  );
}
```



Build

React al igual que Angular y deben generar un paquete de archivos estáticos basados en el código que generamos para poder publicar nuestro proyecto, por lo que necesitamos builder la solución para poder hacerlo.




```
npm run build
```