




REACT.js

CLASE 1



The image features abstract geometric shapes in the corners. On the left, there are overlapping shapes in green, blue, orange, and purple. On the right, there are overlapping shapes in green, blue, purple, and orange. The central text is in a dark blue, sans-serif font.

¿Qué es
React ?

Decorative geometric shapes in the top-left corner, including a green triangle, a purple parallelogram, and a blue parallelogram.

React.js es una **librería** Javascript *open source* desarrollada por Facebook, y focalizada en el desarrollo de **interfaces de usuario**.

Es la V del MVC.

Decorative geometric shapes in the top-right corner, including a purple parallelogram, a green triangle, a red parallelogram, and a blue parallelogram.

Server side rendering

vs

Client side rendering

Ecosistema React

Al ser solo una librería deja de lado muchas otras soluciones que nos aportan los frameworks.

Sin embargo existe todo un ecosistema de herramientas, aplicaciones y librerías que al final equiparan React a un framework.

The logo for ES6 (ECMAScript 2015) features the text "ES6" in a bold, black, sans-serif font, centered within a solid yellow square.

ES6

The Babel logo consists of the word "BABEL" written in a stylized, yellow, hand-drawn script font with black outlines and shadows, giving it a three-dimensional appearance.

BABEL

The React Router logo features a stylized icon above the text "REACT ROUTER". The icon is composed of three black dots and a blue shape resembling a stylized 'R' or a path. The text "REACT" is in black and "ROUTER" is in blue, both in a bold, sans-serif font.

REACT ROUTER

The Redux logo features a purple icon above the word "Redux". The icon is a stylized, continuous purple line forming a shape that resembles a combination of a 'C' and a '3'. The word "Redux" is in a bold, black, sans-serif font.

Redux

NPM

<https://nodejs.org>

NPM (Node Package Manager) es un **gestor de paquetes** de **Javascript** de Node.js.

Por medio de esta herramienta podemos crear, compartir y reutilizar módulos en nuestras aplicaciones de forma sencilla.

NPM

<https://nodejs.org>

1. Instalar NPM (nodejs.org)
2. Instalar paquete create-react-app con NPM
3. Crear proyecto React con create-react-app

facebook ofrece un paquete para crear una aplicación rápida en React sin tener que preocuparnos de las configuraciones Webpack y todas las que involucran para hacerlo funcionar.

```
npm install -g create-react-app
```


facebook ofrece un paquete para crear una aplicación rápida en React sin tener que preocuparnos de las configuraciones Webpack y todas las que involucran para hacerlo funcionar.

```
npm install -g create-react-app  
create-react-app app-react
```

facebook ofrece un paquete para crear una aplicación rápida en React sin tener que preocuparnos de las configuraciones Webpack y todas las que involucran para hacerlo funcionar.

```
npm install -g create-react-app  
create-react-app app-react  
cd app-react
```

facebook ofrece un paquete para crear una aplicación rápida en React sin tener que preocuparnos de las configuraciones Webpack y todas las que involucran para hacerlo funcionar.

```
npm install -g create-react-app  
create-react-app app-react  
cd app-react  
npm start
```

npm start

Importante: debemos estar parados sobre nuestro proyecto ya que la ejecución de **npm start** busca en el archivo package.json un script con la clave start y ejecuta el comando especificado como su valor que nos permite correr nuestra aplicación.

The slide features decorative geometric shapes on both the left and right sides. These shapes are composed of overlapping triangles and polygons in a variety of colors including green, blue, purple, red, and orange, creating a modern, abstract border.

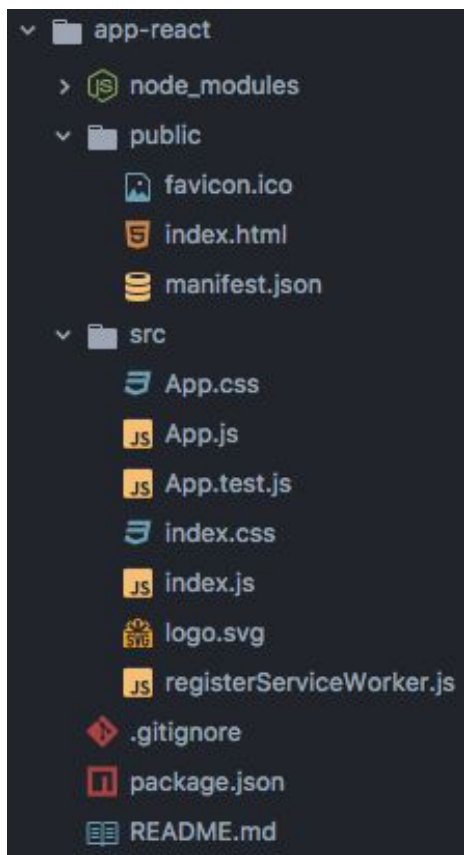
Ejercicio 1

Creamos nuestro proyecto en React.

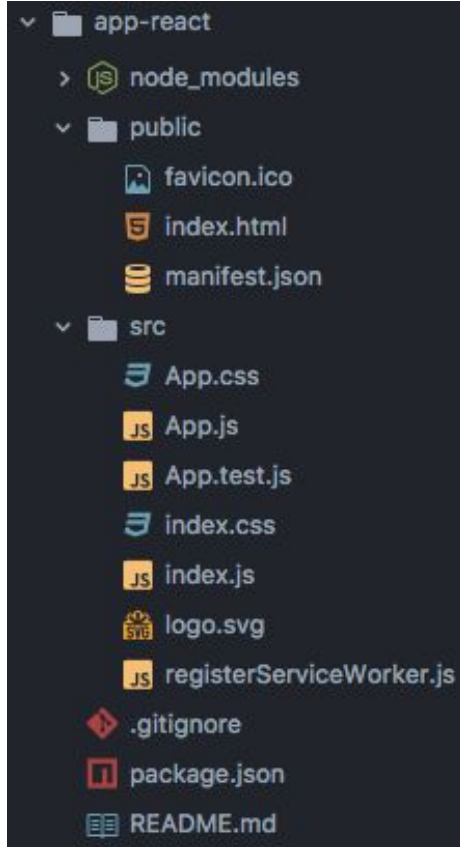
¿Y que nos acabamos de descargar?

create-react-app nos descarga un conjunto de paquetes para comenzar rápidamente con una aplicación basada en React. Incluye:

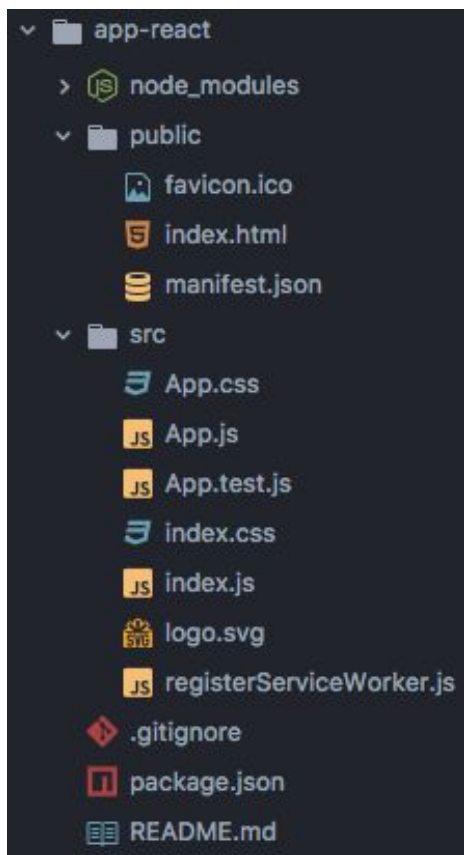
- Librerías de React
- Ecosistema Babel+Webpack configurado
- HMR
- ¡Y varios módulos más!



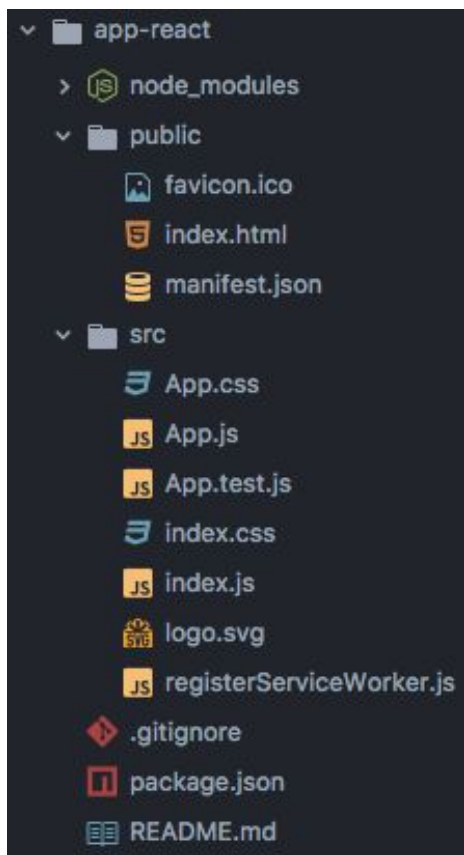
En **node_modules**
están todos los
paquetes de node.js
instalados para el
proyecto en React.



En la carpeta **public** se encuentra el archivo **index.html** que es el archivo html principal que se va cargar cuando el usuario ingresa a la url de nuestra aplicación.



En **src** están todos los archivos donde vamos a trabajar en nuestro proyecto en React.



En **package.json** se especifican las dependencias y las versiones de los paquetes que depende el proyecto.

my-app/src/App.js

```
import React, { Component } from 'react';
import logo from './logo.svg';
import './App.css';

class App extends Component {
  render() {
    return (
      <div className="App">
        <header className="App-header">
          <img src={logo} className="App-logo" alt="logo" />
          <h1 className="App-title">Welcome to React</h1>
        </header>
        <p className="App-intro">
          To get started, edit <code>src/App.js</code> and save to reload.
        </p>
      </div>
    );
  }
}

export default App;
```



my-app/src/App.js

```
import React, { Component } from 'react';
import logo from './logo.svg';
import './App.css';

class App extends Component {
  render() {
    return (
      <div className="App">
        <header className="App-header">
          <img src={logo} className="App-logo" alt="logo" />
          <h1 className="App-title">Welcome to React</h1>
        </header>
        <p className="App-intro">
          To get started, edit <code>src/App.js</code> and save to reload.
        </p>
      </div>
    );
  }
}

export default App;
```

ES6

A cluster of overlapping, semi-transparent geometric shapes in shades of green, blue, orange, and purple, located in the top-left corner of the slide.

ES6

ECMAScript 6 es la nueva versión de Javascript lanzada en el 2015.

A cluster of overlapping, semi-transparent geometric shapes in shades of green, blue, purple, and orange, located in the top-right corner of the slide.

¿Qué tiene de nuevo ES6?

<http://es6-features.org>

- ◆ Arrow functions
- ◆ Clases
- ◆ let & const
- ◆ import & export
- ◆ Template String
- ◆ Entre otras...

Classes

ES6

ES6

```
class Notebook {  
  constructor(brand){  
    this.brand = brand;  
  }  
  getBrand(){  
    return this.brand;  
  }  
  setBrand(brand){  
    this.brand = brand;  
  }  
}
```

```
var notebook = new Notebook("Dell", "Alienware");  
notebook.getBrand();  
notebook.setBrand("Apple");
```

ES6

```
class Notebook {  
  constructor(brand){  
    this.__brand = brand;  
  }  
  get brand(){  
    return this.__brand;  
  }  
  set brand(brand){  
    this.__brand = brand;  
  }  
}
```

```
var notebook = new Notebook("Dell");  
notebook.brand;  
Notebook.brand = "Apple";
```


Import & export

A yellow square containing the text 'ES6' in bold black font.

ES6

```
import Componentes from './componente'
```

Se utiliza para importar funciones u objetos que han sido exportadas desde un módulo externo.

```
export componente;
```

Es usada para exportar funciones, objetos o tipos de dato primitivos a partir de un archivo (o módulo).

Import & Export

simpsons.js

```
var simpsons = {  
  name: "Bart",  
  age: 10  
}  
  
export default simpsons;
```

app.js

```
import simpsons from './simpsons.js';  
  
var personaje = simpsons.name;
```

A yellow square with the text "ES6" in bold black font.

ES6

Import & Export

simpsons.js

```
var simpsons = {  
  name: "Bart",  
  age: 10  
}  
  
export default simpsons;
```

app.js

```
import simpsons from './simpsons.js';  
  
var personaje = simpsons.name;
```



ES6

¡También
podemos
exportar e
importar
clases!

Arrow functions

ES6

Como su nombre lo indica, los arrow functions son funciones definidas usando una **flecha** =>

ES5

```
var duplica = function(num) {  
    return num * 2;  
}
```

ES6

```
var duplica = (num) => num * 2;
```

my-app/src/App.js

```
import React, { Component } from 'react';
import logo from './logo.svg';
import './App.css';

class App extends Component {
  render() {
    return (
      <div className="App">
        <header className="App-header">
          <img src={logo} className="App-logo" alt="logo" />
          <h1 className="App-title">Welcome to React</h1>
        </header>
        <p className="App-intro">
          To get started, edit <code>src/App.js</code> and save to reload.
        </p>
      </div>
    );
  }
}

export default App;
```





Ejercicio 2

Practiquemos ES6



The slide features abstract geometric shapes in the corners. On the left, there are overlapping triangles in shades of green, blue, orange, and purple. On the right, there are larger, more complex shapes in shades of green, blue, purple, and orange, some with internal gradients.

Componente

Nos permiten desglosar el desarrollo de aplicaciones web en pequeños contenedores reusables

¿Podemos crear elementos html de la siguiente forma?

```
var html = '<div class="alert alert-warning">  
    Esto es un HTML </div>';
```

```
document.body.append(html);
```


¿Podemos crear elementos html de la siguiente forma?

```
var html = '<div class="alert alert-warning">  
    Esto es un HTML </div>';
```

```
document.body.append(html);
```



¿Cómo creamos elementos de html desde la programación?

```
var saludo = document.createElement('h1');  
saludo.innerText = 'Hello world!';  
saludo.className = 'text-primary';  
document.body.append(saludo);
```

En React:

```
React.createElement(type, [props], [...children]);
```

Ejemplo:

```
React.createElement(  
  "h1",  
  {className: "text-primary"},  
  "Hello World!"  
);
```

Componentes

```
<div class="text-primary">  
  Hello World!  
</div>;
```



```
React.createElement(  
  "div",  
  {className: "text-primary"},  
  "Hello World!"  
);
```

Componentes

```
<ul id="nav">
  <li><a href="#">Home</a></li>
  <li><a href="#">About</a></li>
  <li><a href="#">Contact Us</a></li>
</ul>
```

```
var nav = React.createElement(
  "ul",
  { id: "nav" },
  React.createElement(
    "li",
    null,
    React.createElement(
      "a",
      { href: "#" },
      "Home"
    )
  ),
  React.createElement(
    "li",
    null,
    React.createElement(
      "a",
      { href: "#" },
      "About"
    )
  ),
  React.createElement(
    "li",
    null,
    React.createElement(
      "a",
      { href: "#" },
      "Contact Us"
    )
  )
);
```

Componentes

```
<ul id="nav">  
  <li><a href="#">  
  <li><a href="#">  
  <li><a href="#">  
</ul>
```

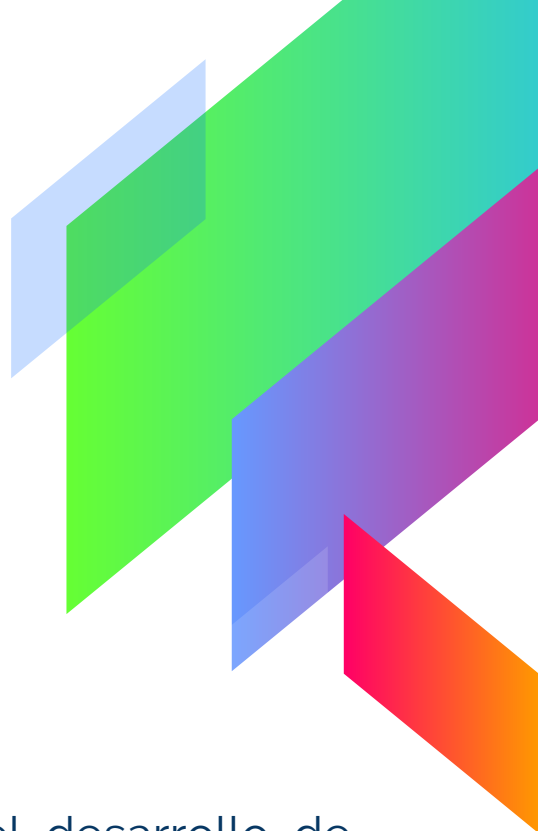


```
var nav = React.createElement(  
  "ul",  
  { id: "nav" },  
  React.createElement(  
    "li",  
    null,  
    React.createElement(  
      "a",  
      { href: "#" },  
      "Contact Us"  
    )  
  )  
);
```

A cluster of overlapping, semi-transparent geometric shapes in shades of green, blue, orange, and purple, located in the top-left corner of the slide.

JSX

JSX es una extensión de ECMAScript.
Es un pseudolenguaje que facilita el desarrollo de aplicaciones para crear elementos en el DOM gracias a su sintaxis muy parecida al XML.

A cluster of overlapping, semi-transparent geometric shapes in shades of green, blue, purple, and orange, located in the top-right corner of the slide.

JSX

```
var nav = <ul id="nav">  
  <li><a href="#">Home</a></li>  
  <li><a href="#">About</a></li>  
  <li><a href="#">Contact Us</a></li>  
</ul>
```


JSX

```
var nav = <ul id="nav">  
  <li><a href="#">Home</a></li>  
  <li><a href="#">About</a></li>  
  <li><a href="#">Contact Us</a></li>  
</ul>
```



Creamos componentes en React con las clases de ES6 que extienden de la clase **Component**.

```
import React, { Component } from 'react';
import logo from './logo.svg';
import './App.css';

class App extends Component {
  render() {
    return (
      <div className="App">
        <header className="App-header">
          <img src={logo} className="App-logo" alt="logo" />
          <h1 className="App-title">Welcome to React</h1>
        </header>
        <p className="App-intro">
          To get started, edit <code>src/App.js</code> and save to reload.
        </p>
      </div>
    );
  }
}

export default App;
```

El componente tiene un método **render** que es el que se encarga de renderizar en el navegador el HTML correspondiente al componente.

En nuestro método render usamos **JSX** para facilitar el desarrollo y creación de elementos HTML.

```
import React, { Component } from 'react';
import logo from './logo.svg';
import './App.css';

class App extends Component {
  render() {
    return (
      <div className="App">
        <header className="App-header">
          <img src={logo} className="App-logo" alt="logo" />
          <h1 className="App-title">Welcome to React</h1>
        </header>
        <p className="App-intro">
          To get started, edit <code>src/App.js</code> and save to reload.
        </p>
      </div>
    );
  }
}

export default App;
```

¡¡IMPORTANTE!

El método render
retorna un elemento.
Por lo tanto, si
tenemos más de un
elemento debemos
meterlos en un
contenedor padre.

class -> className

```
<div className="username"></div>
```

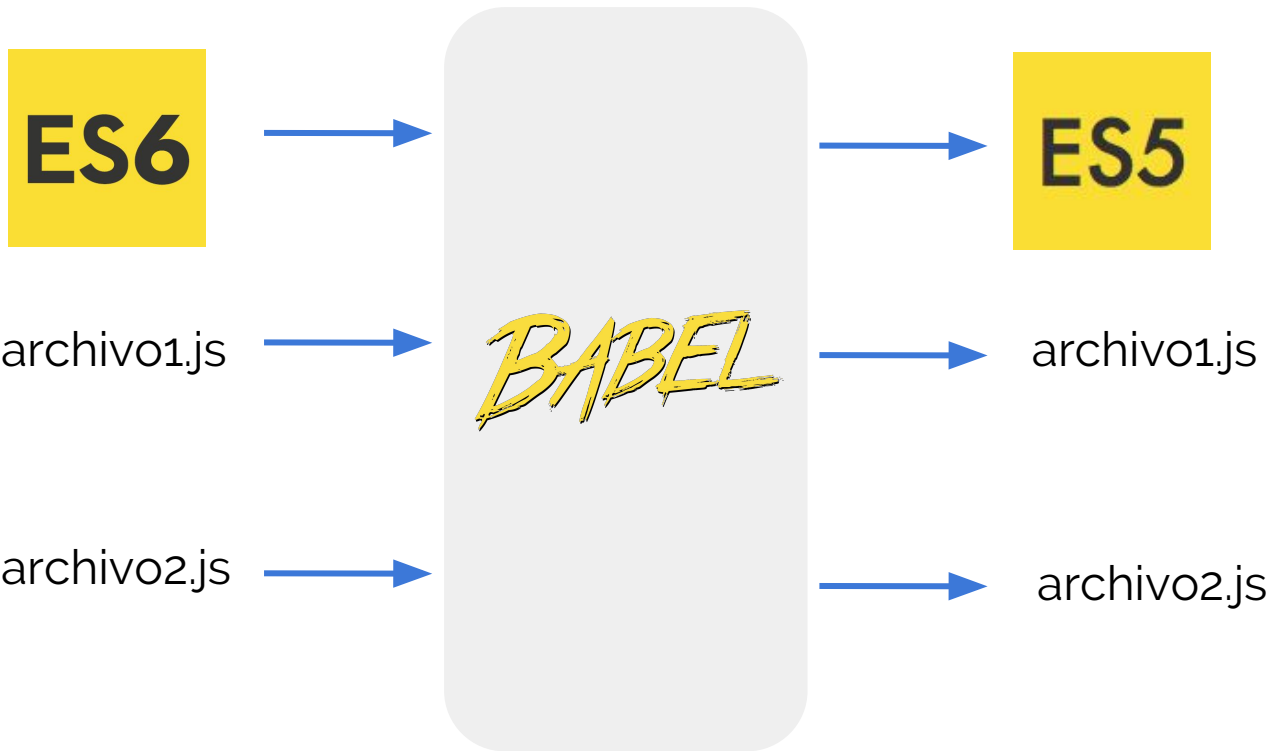
for -> htmlFor

```
<label htmlFor="username">Usuario</div>
```

Un momento...

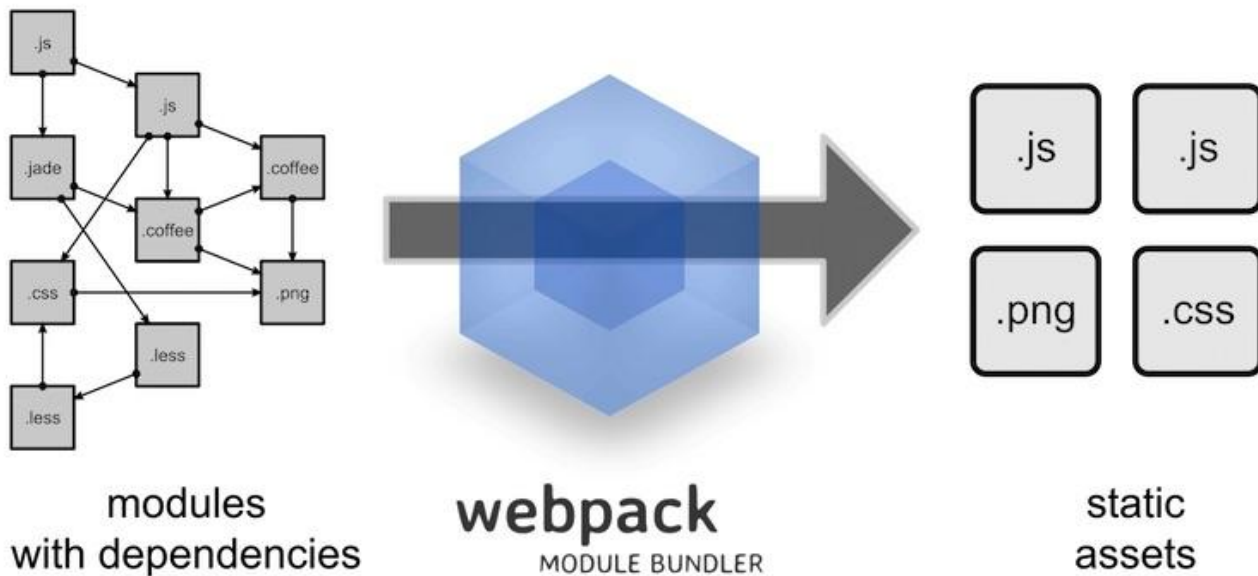


¿Eso es Javascript válido?



Webpack

Es un empaquetador de módulos, te permite generar **un** archivo único con todos aquellos módulos que se necesitan.



BABEL



WEBPACK



JS



my-app/src/App.js

```
import React, { Component } from 'react';
import logo from './logo.svg';
import './App.css';

class App extends Component {
  render() {
    return (
      <div className="App">
        <header className="App-header">
          <img src={logo} className="App-logo" alt="logo" />
          <h1 className="App-title">Welcome to React</h1>
        </header>
        <p className="App-intro">
          To get started, edit <code>src/App.js</code> and save to reload.
        </p>
      </div>
    );
  }
}

export default App;
```



Entonces...supongamos que hago un segundo componente:

//Segundo.js

```
class Segundo extends Component {  
  render() {  
    return (  
      <marquee>Yeah!</marquee>  
    );  
  }  
}  
export default Segundo;
```

Y lo queremos incorporar en **App.js**

App.js se debería ver así:

```
import Segundo from './Segundo.js';

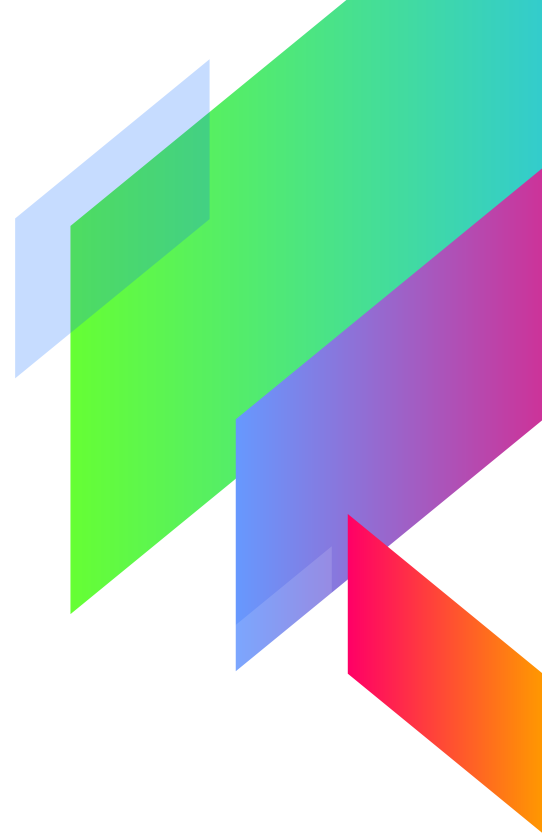
class App extends Component {
  render() {
    return (
      <div className="App">
        <div className="App-header">
          <img src={logo} className="App-logo" alt="logo" />
          <h2>Welcome to React</h2>
          <Segundo/>
        </div>
      </div>
    );
  }
}
```



Ejercicio 3 y 4

Creemos e integremos componentes

Handling Events



```
<button onclick="saludar()">  
  Click me  
</button>
```



En React:

```
<button onClick={saludar()}>  
  Click me  
</button>
```

```
class BotonSaludar extends Component {  
  saludar = () => {  
    alert('Hello World!');  
  }  
  render() {  
    return (  
      <button onClick={this.saludar}>  
        Click me  
      </button>  
    );  
  }  
}
```

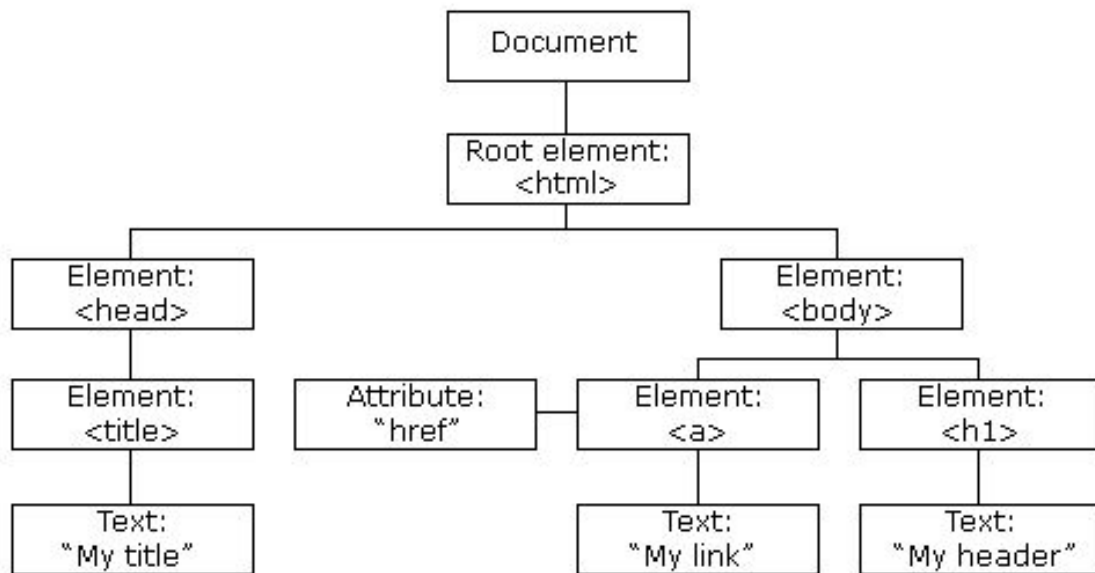
The image features abstract geometric shapes in the corners. On the left, there are overlapping triangles in shades of green, blue, orange, and purple. On the right, there are overlapping triangles in shades of green, blue, purple, and red. The central text is positioned between these decorative elements.

VIRTUAL DOM

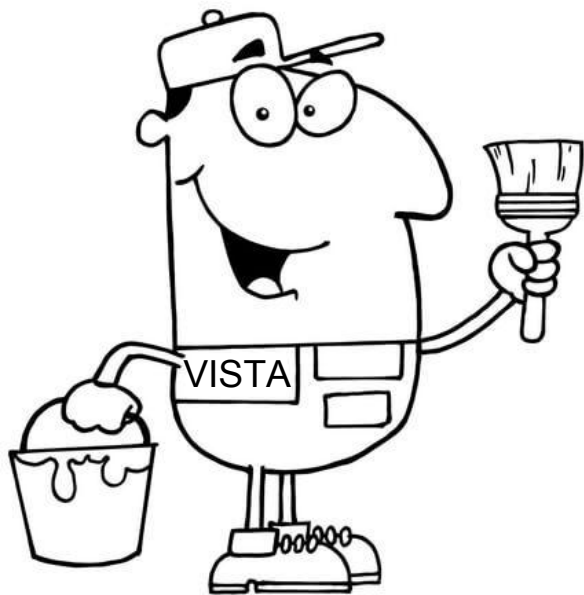
Es una abstracción del DOM.

DOM

Es una representación estructurada del documento HTML y define de qué manera los programas pueden acceder al fin de modificar, tanto su estructura, estilo y contenido.

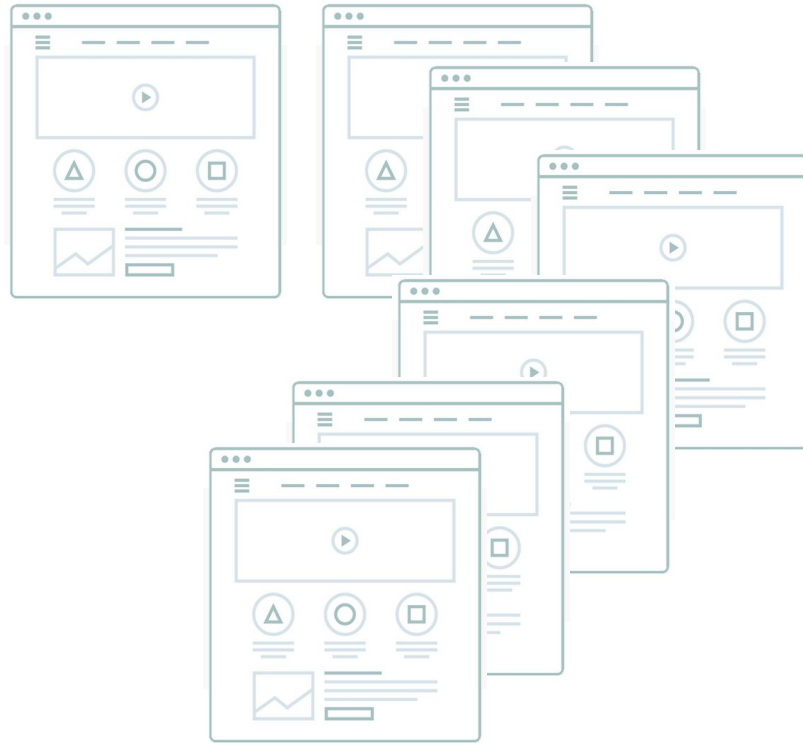


DOM



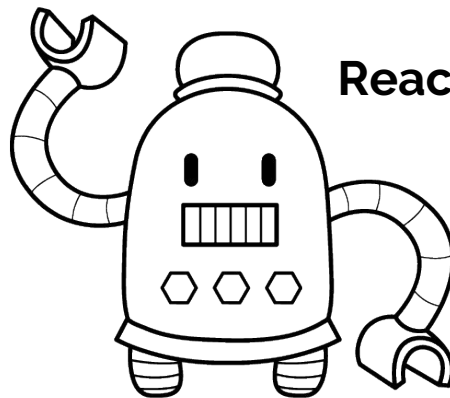
DOM

¿Qué sucede si se reconstruye el DOM cada vez que hay cambios?



React implementa **Virtual DOM**.

React crea una copia ligera del DOM y en cada cambio lo compara con el DOM Real. En lugar de renderizar el DOM completo en cada cambio, los aplica exclusivamente en las partes que varían.



ReactJS

my-app/src/index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import registerServiceWorker from './registerServiceWorker';

ReactDOM.render(<App />, document.getElementById('root'));
registerServiceWorker();
```

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



Ejercicio integrador



Gracias!

¿Preguntas?

