**REACT** – Facebook 2003

IDE ONLINE: https://www.programiz.com/javascript/online-compiler/

# DECLARAR UNA VARIABLE

var x = 0; // Scope Global - Se evita var para no sobreescribir variables

let x = 1; // Scope Local

const c = true; // Scope Local

c = false; // No se puede cambiar el valor asignado inicialmente

# ARRAYS

let arr = [1,'a',"auto",[1,2,3,4],true,{name: 'Luciano', age: 31}]

console.log(arr); // (6) [1, 'a', 'Luciano', Array(4), true, {name: 'Luciano', age: '31'}]

console.log(arr[1]); // a

console.log(arr[3][1]); // 2

console.log(arr[5].name); // Luciano

# MANIPULAR VALORES DE UN ARRAY

let num = [1,2,3] //--------------- (3) [1, 2, 3]

num.unshift('x') // (4) [x, 1, 2, 3] - Agrega un valor al principio

num.push('y') // (5) [x, 1, 2, 3, y] - Agrega un valor al final

num.shift() // (4) [1, 2, 3, y] ------ Elimina el primer valor

num.pop() // (3) [1, 2, 3] ----------- Elimina el ultimo valor

let numeros = [1,2,3] // (3) [1, 2, 3]

let letras = ['a','b','c'] // (3) ['a', 'b', 'c']

let caracteres1 = numeros + letras // 1,2,3a,b,c

let caracteres2 = numeros.concat(letras) // (6) [1, 2, 3, 'a', 'b', 'c']

let caracteres3 = [numeros, letras] // (2) [Array(3), Array(3)]

# SPREAD OPERATOR …

let caracteres4 = [...numeros, ...letras] // (6) [1, 2, 3, 'a', 'b', 'c']

# OBJETO

let estudiante = {

    name: 'Luciano', // Se compone de clave y valor

    age: 31

}

console.log(estudiante.name); // Luciano

let infoEstudiante = {

    //estudiante, // {estudiante: {…}, carrera: 'CTD', bimestre: 2, time: 'PT'}

    ...estudiante, // {name: 'Luciano', age: 31, carrera: 'CTD', bimestre: 2, time: 'PT'}

    carrera: 'CTD',

    bimestre: 2,

    time: 'PT'

}

# FUNCIONES

1. Funciones Expresadas

function myFunc1(){

}

1. Funciones Anonimas

const myFunc2 = function(nombre) {

}

1. Arrow Function: Se utiliza en callback

En ves de declarar funciones con nombres, se declara una funcion anonima que funcione a modo de callback y no va hacer falta declararla o asignarla ya que solo funciona en ese contexto

const myFunc3 = () => {

}

# LIBRERÍA REACT  
Metodo: React.CreateElemento(component, props, …children)

1. Tipo de elemento HTML
2. Propiedades del elemento
3. Contenido dentro del elemento

let titulo = React.createElement('h1', {class: 'titulo'}, 'Hola Lucho') // <h1 class='titulo'>Hola</h1>

HTML

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Documento</title>

    <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>

    <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>

</head>

<body>

    <div id="root"> // root Original

    </div>

    <script src="/LibReact.js"></script>

</body>

</html>

# EJEMPLO N°1

let titulo = React.createElement('h1', null, 'Hola Lucho') // <h1>Hola</h1>

let perro = 'https://linckdelperro.com.ar';

let img = React.createElement('img', {src: perro, alt:''})

let seccion = React.createElement('div', null, titulo, img)

ReactDOM.render( //---------------------- Renderizar

    seccion, //-------------------------- Componente que deseo agregar

    document.getElementById('root') //--- Donde lo deseo agregar

)

HTML - (Renderizado con JavaScript)

    <div id="root">

        <div>

            <h1>Hola Lucho</h1>

            <img src="https://linckdelperro.com.ar" alt="">

        </div>

    </div>



**# Ejemplo N°2**

Se define una constante llamada ‘lista’ que contiene una matriz de nombres de peliculas

const lista = [ // Lista de peliculas

    'Rapido y Furioso',

    'Titanic',

    'Mar de plastico',

    'La casa de papel',

    'El patron del mal'

]

document.getElementById('root').innerHTML = `<ul>${lista.map((pelis) => `<li>${pelis}</li>`)}</ul>`;

document.getElementById(‘root’): Obtener el elemento con el identificador "root"

root: Contenedo donde se mostrara la lista de peliculas

InnertHTML: Se asigna el contenido del elemento root

lista.map(pelis): Se utiliza el metodo map en la matriz lista para crear una serie de elementos de lista <li> que contienen los nombres de las peliculas

HTML - (Renderizado con JavaScript)

    <div id="root">

        <ul>

            <li>Rapido y Furioso</li>

            <li>Titanic</li>

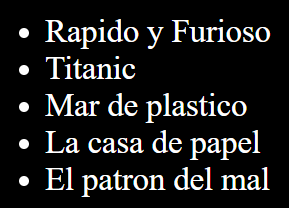
            <li>Mar de plastico</li>

            <li>La casa de papel</li>

            <li>El patron del mal</li>

        </ul>

    </div>



lista.map(): --------------------- Recorre la matriz “lista”

React.createElement(): ----- Crea cada elemento html

movies: -------------------------- Contiene los elementos <li></li>

const Peliculas = () => {

    let movies = lista.map( (peli, i) => {

        return React.createElement('li', null, (i+1) + '.' + peli) // <li>1.Rapido y Furioso</li>

    });

    return React.createElement('ul', null, movies) // <ul><li>1.Rapido y Furioso</li></ul>

}

ReactDOM.render(): Renderiza el contenido de la funcion peliculas en el elemento con id root

ReactDOM.render( // Agrega la lista de peliculas al <div id="root"> .... </div>

     Peliculas(), // Contenido

     document.getElementById('root') // Contenedor

)

