RESUMEN LC3

Que es React y porque usarlo: React es una biblioteca Javascript para construir interfaces de usuario, su nombre viene de reactive, la idea es que podamos crear aplicaciones como en la de nuestros teléfonos, siendo los input que toca el usuario responsivos.
En este caso JavaScript corre del lado del cliente, manipulando el DOM del HTML, para lograr mayor fluidez y experiencia dentro del sitio web.
utilizamos REACT y no JS VANILLA debido a que como desarrolladores tenemos que buscar siempre la manera más eficiente de realizar los proyecto, buscando escribir mejor código en menos tiempo.

Single Page Application (SPA) con React: SPA son las aplicaciones web que traen todo el contenido de la misma en una sola pagina (en el index), la cual se encarga de contener la totalidad del front ent junto a su lógica. Se intenta que la navegación sea lo más fluida posible.

Diferencias React, Angular y Vue: Para comenzar React es una librería, mientras que Angular y Vue son Frameworks, por ende son mas completos y en teoría permiten mas funcionalidades. Mientras que react solo cubre las funcionalidades básicas del desarrollo frontend.
Angular actualmente no es considerado para proyectos pequeños y medianos, ya que se considera demasiado para los requerimientos del cliente.
Vue sin embargo junta lo mejor de los dos mundos, la estructura de angular y el manejo de componentes de react, pero todavía no cuenta con la comunidad gigante que tienen los otros.

JavaScript Moderno: Refiere a funcionalidades ES6+ (ecmascript 6 y poste.). Esto es debido a que dichas funcionalidades nos aportan mejoras (tal como la conocida arrow function que además de ser más sencilla de escribir nos libera del concepto de this en JS) como atajos de manera que escribamos menos código. ES LO QUE SE UTILIZA EN REACT.

El reemplazo a var: Para crear variables utilizamos let o const, let para variables que puede ser que en un futuro cambien de valor después de inicializarse, permite ahorrar un poco de memoria ya que estan acotadas al bloque en el que se definen. y const para valores que no se planea que cambien en el futuro

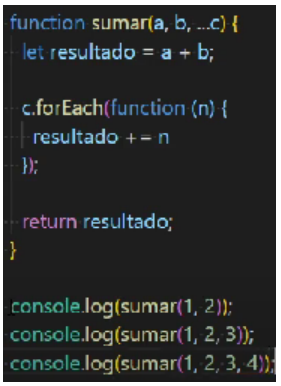
5 cambios de JavaScript ES6: **1** - Creación de clases como si fuera un lenguaje de POO. **2** - Funciones de flecha, con distinta escritura que las comunes que permiten ahorrar codigo. **3** - Variables con let y const. **4** - Módulos imports y exports: Permite tener codigo en distintos archivos logrando orden y importarlos solamente cuando los necesite **5** - Templete string, es una forma de definir un string de manera multilínea, no necesito concatenar.

Arrow Functions: Es una de las características más importantes agregadas en ES6, se caracteriza por ser mas corta que una función tradicional. Las funciones de flecha SIEMPRE son funciones anónimas. El return viene de manera concisa en la primera linea de la función.

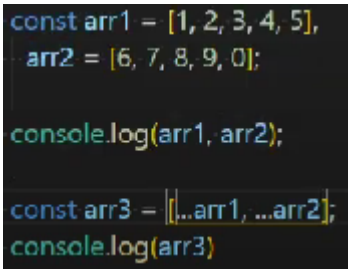
```
const square = (x) => x * x;
```

Parámetros Rest: Son una forma de ir agregando parámetros infinitos virtualmente ya sea a una función o una variable. Para definir los parámetros rest se ponen 3 puntos suspensivos antes de estos supuestos valores infinitos. PUEDE O NO ESTAR, y nunca va a dar error

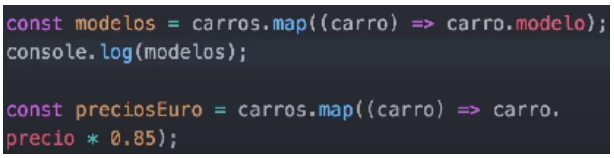
.



Spread operador: Nos va a permitir que cuando tengamos que expandir una expresión para guardar múltiples elementos de mas en un arreglo. Se utilizan tres puntos suspensivos antes del arreglo a concatenar.



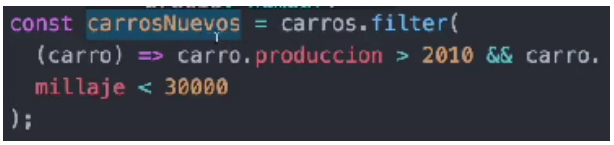
Método map(): Ejecuta una función para cada elemento del array y lo que hace es crear un array con los resultados de esta función



aca podemos agarrar un elemento especifico para modificarlo

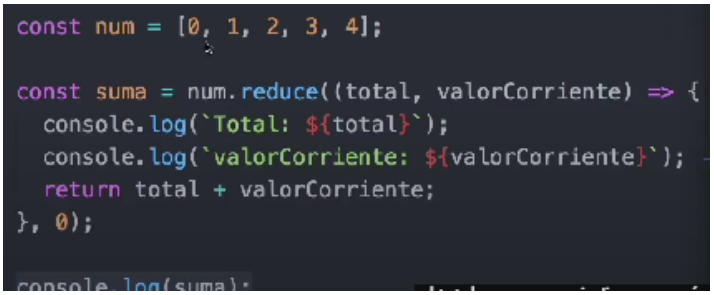
El metodo map no se debe usar cuando no queremos crear un nuevo array, en ese caso usamos el metodo forEach().

Metodo filter(): Tambien se crea un nuevo array, pero este coloca los mismos elementos del original menos que los “filtra” cumpliendo cierto requisito.



Toma cada elemento del array original y los somete a cierta condicion y en base eso los agrega o no.

Metodo reduce(): Este va a ejecutar una función sobre cada elemento del array, pero en vez de devolver un nuevo array solamente va a devolver un solo valor.



Extrae valor por valor para este caso sumar la totalidad del mismo.

Metodo concat(): concatena los arreglos uno atras del otro sin ordenar

```
let str1 = "Hello Worold!";
let str2 = "Weolcome to Web Design House";
let newStr = str1.concat(str2);

document.write(newStr)
```

Metodo find(): devuelve el primer elemento que encuentre que cumpla con cierta condición.

```
const userlist = users.find((user) => user.age < 30);
console.log(userlist);
// { id: 3, name: "Tracey", age: 28}
```

Componentes en React: Bloques de código que funcionan como unidad principal de trabajo en React, sus principales características son: *- Reusabilidad:* Después de crear un componente, este puede ser usado varias veces en distintas partes de nuestra página web, evitando repeticiones de código. *-Separación de funcionalidades:* cada componente podrá tener su lógica interna independiente o no al resto.

Los componentes están compuestos por HTML5, CSS Y JS, en el caso de react este usa un acercamiento declarativo en la construcción de estos. Nosotros definimos el estado deseado de estos comp y React los traduce en algo que el navegador puede entender nativamente. JSX: JavaScript XML.

HOOKS: Conjunto de funciones incluidas en react. Están las basic Books, que se usan continuamente y las Additional Hooks.

Hook de estado en React (useState): Nos permite manipular los estados de un componente para asi conseguir la renderización de los mismos, así dejaría de ser un componente estático. Basicamente le informamos a react que hubo una modificacion y que debemos actualizar LOS COMPONENTES AFECTADOS.

```
App.sx > Likes
import { useState } from 'react';

const Likes = () => {
    const [likes, setLikes] = useState(0);

    return <button onClick={() => setLikes(likes + 1)}>{likes} likes</button>;
};

const App = () => {
    return <Likes />;
};

export default App;
```

useEffect: Se utiliza como un apartado de codigo paralelo dentro del componente el cual se carga una vez se termino de renderizar la pagina, puede servir para llamar api, o realizar tareas las cuales se necesite que un apartado de codigo se ejecute varias veces pero sin intervenir en el resto del componente. Todo esto sin volver a renderizar necesariamente.

useRef(Additional Hook): Nos permite acceder al DOM de JS, que por defecto en React esta prohibido. Podemos hacer modificaciones en el componente y demas cosas como si fuera JavaScript Vanilla.

```
function App() {

  const inputRef = useRef(null);

  const makeThings = () => {
    alert(inputRef.current.value);

  }


  return (
    <div>
      <input type="text" ref={inputRef} />
      <button onClick={makeThings}>Hacer Cosas</button>
```

Props en React: Los props nos permiten pasar datos o atributos de un objeto entre componentes y funciones. Los pasa por medio de objetos.
Es muy importante para la reutilización de componentes ya que podemos hacer un comp. base el cual mediante props reciba distintos datos para ir renderizando varias veces.
TAMBIEN SIRVE PARA PROPIEDADES CSS.

```
function Person(props) {
  console.log(props);
  return (
    <div className="person">
      <h3>Nombre: {props.nombre}</h3>
      <p>Edad: {props.edad}</p>
    </div>
  );
}

function App() {
  return (
    <div className="App">
      <h1>Hola Mundo</h1>
      <Person nombre="Pedro" edad="25" />
    </div>
  );
}
```