Para instalar angular

npm install -g @angular/cli

ng new my-first-project

assets para hojas de estilo

componente app en localhost 4200

en declaration se cargan los componentes

en import otros módulos

bootstrao indica cual es el componente principal por el cual se arrancara la aplicación

REACT

NPX CREATE-REACT-APP . PARA CREAR UNA APP EN UN FOLDER

NPM START PARA INICIALIZAR APP

EN app.js se borra todo menos el ultimo div

HTML

Para que sirve la etiqueta head?

-es la etiqueta utilizada para ingresar información para el navegador como metadatos que no estará visiblemente para el usuario

En que se diferencia el head del body?

-en el body se pone el contenido que se mostrara en la web y en el head se contiene la información para la creación de la web

Como poner imagen en html?

* Img src=”” y alt””

Como poner video en html?

- video src o etiqueta source

Para que sirve un iframe?

para cargar una pagina web externa dentro de nuestra pagina

Como crear un formulario?

Form y los campos como imput, text área, el botón y listo

Que hace el botón submit?

Te da posibilidad de enviar el formulario

Diferencias entre Vr y hr?

Vr hace salto de línea y hr dibuja una línea divisoria entre dos lineas

Etiqueta genérica para crear una caja?

div

cuales son las etiquetas semánticas para html?

Main, header, nav, aside, section, article, footer.

Diferencia entre ol y ul?

Ol para crear lista ordenada

Ul para lista no ordenada

Etiqueta para poner texto en cursiva?

<i>

Como hacer un enlace en html?

<a href=”#”/>

Diferencia entre tr y td?

Tr crea fila dentro de una tabla

Td para columna

Abrir enlace en una pestaña nueva?

Atributo target= “-(bajo)blank”}

CSS

Diferencia entre id y class?

Id para identificar un elemento en especifico y no se puede repetir y class para seleccionar muchos elementos dentro de la misma clase o que sean de la misma clase.

Como seleccionar un id o una clase?  
# id

.class

Con que propiedan puedes cambiar fuente de un texto?

Font-family

Como cambiar fondo de web?

Background-color

Propiedad para fijar un menú dentro de una web?  
 position: fixed;

De que distintas formas puedo posicionar cajas dentro de css?

Float, flexbox y grid

Propiedad para poner sombras dentro de css?

Para texto en text-shadow y si es para una caja box-shadow

Propiedad para redondear caja?

Border-radious;

Con que hacer web responsive?

Media query

Si una imagen sobre sale de una caja como hacer que entre bien dentro de la caja?

con un overflow hidden para eliminar lo que sobresalga de la caja

Diferencia entre flexbox vs grid?

Grid tiene cuadricula con areas y flexbox es solo niveles horizontales y verticales

Para que sirve preprocesador css?

Sirven para ahorrarnos trabajo con css, nos dootan a css de cualidades y características de lenguaje de programación por ejemplo las variables, funciones, anidaciones, etc.

Diferencia entre preprocesador css y framework css?

preprocesador nos da funcionalidades extra para programar mas rápido y los frameworks nos da ya los elementos hechos.

Que propiedades puedo usar para hacer animaciones?

Animation y @keyframes{} para deifnir la animación

Transition

Si quiero hacer una caja o algo en 3d dentro de css como se hace?

Jugando con las sombras

Como se manejan las capas en css?  
 z-index

JS

1 - ¿Cómo puedo mostrar un alert?

// Con la función alert

alert("hola youtube);

--------------------------------------------------------------------------------------------------------------------------------------------------------------

2- ¿Con qué función de JavaScript puedo pasar datos por consola?

// Con la función console.log

console.log("Hello people of YT");

--------------------------------------------------------------------------------------------------------------------------------------------------------------

3 - ¿Cuál es la diferencia entre Var, Let y Const? //

La diferencia es le alcance de la variable

Var puedo acceder a ella desde cualquier parte del código y cambia su valor en cualquier momento

Let se puede inicializar UNA vez y cambiar su valor las veces que se quiera - Definida dentro de un bloque, Let tiene un alcance de bloque

Const SOLO se puede inicializar UNA vez y solo se puede asignar su valor UNA vez -es una constante

var edad = 20;

var edad = 10;

console.log(edad); // edad = 10

let edad = 20;

let edad = 10; // aquí ya da error

console.log(edad); // error, no puedo inicializar dos veces la misma variable con let

let edad = 20;

edad = 10;

console.log(edad); // edad = 10

const edad = 20;

const edad = 10; // aquí ya da error, puesto que no puedo inicializar el valor de una const dos veces

edad = 10; // aquí daría otro error, puesto que no puedo reasignar el valor de una const

--------------------------------------------------------------------------------------------------------------------------------------------------------------

4 - ¿Cómo detecto el tipo de dato de una variable en JavaScript?

//utilizando el metodo typeOf()

let edad = 20;

const tipoDeDato = typeOf(edad);

console.log(tipoDeDato); // muestra en consola el mensaje "number"

--------------------------------------------------------------------------------------------------------------------------------------------------------------

5 - ¿Cuáles son las estructuras de control más usadas en JavaScript?

// Se trata de los condicionales y bucles.

Es decir: if, else if, else, switch , case , for , forEach , do , do While, for in, for of (faltan más)

if (true) {

console.log("hola gente"

} else if {

console.log("este mensaje nunca se va a mostrar porque el if es true siempre")

} else {

console.log("pasemos a como escribir otra estructura")

switch() {

case 1:

alert ("Hola España")

break;

case 2:

alert ("Hola Argentina")

break;

case 3:

alert ("Hola hermanos latinoamericanos")

break;

default:

alert ("Basta de guerra")

break;

}

6 - ¿Qué es un template string en JavaScript? //

Es una cadena de texto en la cual se pueden agregar variables o funcionalidades de javascript dentro del mismo.

Es un string que se abre con las comillas invertidas (alt +96).

Es una forma de escritura más sencilla en lugar de estar aplicando concatenaciones entre strings y el simbolo "+".

function holaMundo(pais) {

return `Hola gente de ${pais}`

}

holaMundo(argentina); // se retorna un string que dice "Hola gente de argentina"

--------------------------------------------------------------------------------------------------------------------------------------------------------------

7 - ¿Qué es una función de Call Back?

Es una función que se pasa como argumento/parametro a otra.

Es una función que se ejecuta dentro de otra.

function showAlert() {

alert("Si estas viendo esto, la alerta se activo");

}

showAlert(); //muestra una alerta con el mensaje "Si estas viendo esto, la alerta se activo"

--------------------------------------------------------------------------------------------------------------------------------------------------------------

8 - ¿Qué es un array multidimensional?

// Se trata de un array que contiene otros arrays, los cuales, a su vez, podrían contener otros arrays. Ósea tiene varias dimensiones

let array = [

[ España, Italia, Alemania, Inglaterra ] ,

[ Argentina, Chile, Brasil, Paraguay ] ,

[ China, Japon, Corea del Norte, India ] ,

];

--------------------------------------------------------------------------------------------------------------------------------------------------------------

9 - ¿ Qué diferencia hay entre for in y for of?

// Ambos ciclos recorren un array y extraen información del mismo.

For...in recorre el array y obtiene los indices y los métodos.

For...of recorre el array y obtiene los valores de esos indices.

let array = [ 33, 57, 74 ] ;

array.test = "beta" ;

for (let i in array) {

console.log (i) ;

};

//se mostraran en consola los índices del arreglo, que son el 0, el 1 y el 2. Tambien se mostrara la palabra "test"

for (let i of array) {

console.log (i) ;

};

// se mostraran en consola los valores de los índices del arreglo, que son el 33, el 57 y el 74. No se muestra la palabra "beta"

--------------------------------------------------------------------------------------------------------------------------------------------------------------

10 - ¿Cómo contar los elementos que hay dentro de un array?

//Utilizando el método .lenght

let array = [ php, javascript, html, css]

const numberElements = array.lenght;

console.log(numberElements); // Se mostrara el numero 4 en consola

11 - ¿Qué diferencia hay entre DOM y BOM?

Document Object Model !== Browser Object Model

El BOM depende del navegador que se esta ejecutando, mientras que el DOM es independiente de este.

DOM hace referencia a todos los elementos que hay en nuestra página, como las etiquetas HTML (aside, footer, title, nav, html, body, head)

El BOM hace referencia a los elementos del navegador en sí: las pestañas. la barra de URL, las opciones, el menú, la ventana donde se muestra el contenido.

--------------------------------------------------------------------------------------------------------------------------------------------------------------

12 - Mencione métodos para acceder a elementos del DOM dentro de la página. // Varios métodos (no todos)

document.GetElementbyId()

document.getElementByTagName()

document.getElementByClassName()

dicument.querySelector()

document.querySelectorAll()

--------------------------------------------------------------------------------------------------------------------------------------------------------------

13 - ¿Cuál es la diferencia entre Focus y Blur? // Se trata de eventos opuestos

Focus: Es un evento que se produce cuando uno se enfoca sobre un elemento.(dentro del elemento)

Blur: Es un evento se que se produce cuando uno sale de un elemento.(fuera del elemento)

Al posar el puntero sobre una etiqueta <a> se produce un focus a la etiqueta <a>

Al quitar el puntero de encima de una etiqueta <a> se produce un blur a la etiqueta <a>

--------------------------------------------------------------------------------------------------------------------------------------------------------------

14 - ¿Cómo manejar eventos en JavaScript? // Solo algunas formas

Manejar un evento en JavaScript implica ejecutar código JavaScript ante las acciones de un usuario.

Algunos manejadores de eventos son:

Con el atributo y llamada a una funcion // Su uso es menos conveniente

onchange()

onclick()

como propiedad o método de JavaScript: // Su uso es más recomendado

onclick()

con el addEventListener()

--------------------------------------------------------------------------------------------------------------------------------------------------------------

15 - ¿Cómo guardar un objeto json en el localstorage?

Primeramente es necesario convertir el objeto a un string json, ya que localstorage no permite guardar objetos javascript como tales.

localStorage.setItem("user",JSON.stringify(my\_object));

16 - ¿Cómo hacer una petición a una API REST?

Se puede consumir una API con AJAX, la cual es la forma más tradicional y antigua, pero este método esta en desuso.

Se puede consumir una API con fetch, el cual es un método mas efectivo y amigable.

Se puede llamar a una API utilizando AXIOS.

Usando AJAX:

const http = new XMLHttpRequest();

const url = "https://jsonplaceholder.typicode.com/posts"

http.open("GET", url);

http.send();

http.onreadysetchange = (e) => {

console.log(http.responseText);

}

Usando Fetch:

const url = "https://jsonplaceholder.typicode.com/posts" ;

fetch(url)

.then (data => {return data.json()})

.then (res => {console.log(res)})

Usando Axios: Se trata de una biblioteca de código abierto que se utiliza para hacer peticiones HTTP. Se debe instalar e "importar" para su uso. Chequear en google como se utiliza en forma de código.

--------------------------------------------------------------------------------------------------------------------------------------------------------------

17 ¿Con cuál objeto de JavaScript puedes manejar fechas?

// Con el objeto Date

const now = new Date();

console.log(now); // Se muestra en consola la fecha y hora actual relativo a tu pais.

Chequear más formas de tratar con date aqui:

https://es.javascript.info/date

--------------------------------------------------------------------------------------------------------------------------------------------------------------

18 - ¿Qué es una promesa en JS?

Se trata de un objeto asincrónico al cual se le adjuntan funciones de tipo call back, dando como resultado una función asíncrona.

Básicamente nos promete que obtendrá los datos aunque estén retrasasdos.

const promise = new Promise((resolve, reject) => {

const number = Math.floor(Math.random()\*10);

setTimeout(

() => number > 5

? resolve(number)

: reject(new Error("menoro igual a 5")),

1000

);

});

promise

.then(number => console.log(number));

// Después de 1 segundo, se obtiene un número al azar, el cual es mostrado en consola siempre y cuando este sea mayor a 5.

//De lo contrario, aparecerá un error diciendo que el numero es menor a 5.

--------------------------------------------------------------------------------------------------------------------------------------------------------------

19 - Menciona dos Frameworks SPA para JavaScript.

(Angular, React, Vue, Svelte).js

--------------------------------------------------------------------------------------------------------------------------------------------------------------

20 - ¿Es posible realizar backend con JavaScript? En caso de si, ¿con cuál tecnología lo harías?

Se puede emplear Node.js para utilizar JavaScript en un entorno backend.

Combinando Node, Express y Mongo, es posible incluso realizar un proyecto web completo.

¿Qué es la consola de desarrollo?

La ​consola de desarrollo​ es una herramienta con la que cuentan los browsers modernos. Esta consola nos permite, entre otras cosas, ver el código que muestra nuestro browser: ​html​, ​css ​y ​javascript​, aunque también podría mostrar otros lenguajes como ​typescript​, ​less ​y ​sass, ​entre otros.

SASS

**¿Qué es SASS y cuáles son sus usos?**

Yo veo a SASS como un sistema que se utiliza para programar css, lo que ayuda a reducir el tiempo de programación y a su vez aumenta la reutilización del código y reduce la redundancia del código.

#### ¿En qué se diferencia SASS de CSS?

CSS es hojas de estilo en cascada, SASS es un preprocesador CSS que permitirá usar las diferentes operaciones y usar variables, mixins, funciones, importaciones y diferentes tipos de bucles.

#### ¿Qué es una variable y cómo se define en SASS?

Una variable en SASS se utiliza para almacenar la información que se puede reutilizar en toda la hoja de estilo. Una variable se definirá comenzándola con un signo de dólar ($).

#### ¿Cuáles son los diferentes tipos de operaciones en SASS?

Los diferentes tipos de operaciones que se usan en SASS son Operaciones con números Operaciones con listas Operaciones con cadenas Operaciones con colores y Operaciones booleanas. Las operaciones numéricas implican operaciones como sumar, restar, multiplicar y dividir. Las operaciones de lista implican operaciones tales como un conjunto de valores que están separados por comas en forma de matriz. Las operaciones de cadena implican operaciones como la concatenación o la división de las cadenas. Las operaciones de color implican operaciones como el uso de los componentes de color con operaciones aritméticas y, por último, las operaciones booleanas implican operaciones como el uso de operadores AND, OR, NOT para realizar operaciones booleanas en SassScript.

#### ¿Qué es Mixin?

Un Mixin en SASS es un SASS Mixin que se llama como funciones CSS.

#### ¿Qué es SassScript y en qué se diferencia de SASS?

SassScript proporciona diferentes tipos de mecanismos en CSS, como Nesting, Variables, Mixins, Selector Inheritance, etc., SassScript es un conjunto de pequeñas extensiones que SASS utilizará. Esto también se puede usar con nombres de propiedades y selectores mientras se usa con mixins.

#### ¿Cuáles son los diferentes tipos de argumentos Mixin?

Los diferentes tipos de argumentos Mixin son Argumentos de valor y Argumentos de palabra clave. Los argumentos pueden ser valores SassScript en el caso de Mixins. Los argumentos variables se utilizan para pasar cualquier número de argumentos a Mixin.

#### ¿Cuáles son las diferentes funciones disponibles en SASS?

Las diferentes características en SASS son símbolo de referencia, parciales y directivas, interpolación, marcadores de posición, funciones, listas, etc. También superestimarán el CSS al proporcionar diferentes funcionalidades para facilitar el desarrollo y proporcionar la reutilización del código. Las variables, la anidación y las matemáticas también están disponibles en el SASS. Los diferentes tipos de estilos se pueden reutilizar con Mixins.

#### ¿Qué es la función de interpolación en SASS?

La interpolación también se llama sustitución de variable o interpolación de variable, que se define como la interpolación de una expresión o una variable. Este es un proceso de una cadena o una expresión que contiene más de una variable donde estas variables serán reemplazadas en la memoria con sus valores correspondientes. La interpolación tendrá sintaxis para ser utilizada en el SASS.

#### ¿Cuáles son las ventajas y desventajas de SASS?

SASS permitirá escribir código CSS limpio y evita la redundancia de código. SASS es un superconjunto de CSS que ayudará a los desarrolladores a escribir el código de manera más eficiente y rápida. Las desventajas de CSS son que tendrá más curva de aprendizaje para que los desarrolladores aprendan todas las características involucradas en SASS.

**REACT**

¿Que es React?

React es una biblioteca o librearía escrita en JavaScript, desarrollada en Facebook para facilitar la creación de componentes interactivos, reutilizables, para interfaces de usuario.

¿Que es un componente?

Una sección de código re utilizable en cualquier otro componente o página del sitio.

¿Que son las props?

Son las propiedades o información que viaja de un componente padre a un hijo para ser utilizada.

¿Cuál es el ciclo de vida de un componente de clase?

componentWillMount

componentDidMount

componentWillReceiveProps

componentWillUpdate

( Más importante que se sepan los nombres es que sean conscientes de este ciclo )

¿Qué es un estado?

Lugar donde el componente almacena datos de manera local y en tiempo real.

¿Qué es JSX?

JSX es una extensión de JavaScript creada por Facebook para el uso con su librería React. Sirve de preprocesador (como Sass o Stylus a CSS) y transforma el código a JavaScript. Nos permite usar HTML dentro de archivos de Javascript.

¿Qué librería se usa para implementar las rutas?

React Router

¿Qué son los 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.

¿Qué es redux o el context API?

Redux es un patrón de arquitectura de datos que permite manejar el estado de la aplicación de una manera predecible. Está pensado para reducir el número de relaciones entre componentes de la aplicación y mantener un flujo de datos sencillo. Y en muchas ocasiones funciona para comunicar componentes entre sí cuando no tienen una relación directa (Padre – Hijo)

¿Qué es AXIOS o FETCH?

Son métodos para solicitar peticiones a una API y nos ayudan a hacer más sencilla la comunicación con servicios de backend que alimenten con información nuestra aplicación.

## ¿El método setState() es asíncrono?

Sí. Si bien esta naturaleza puede no manifestarse claramente en muchos escenarios, **el método**

**setState()**

**es asíncrono** ya que las acciones realizadas a través del mismo pueden ser agrupadas y atendidas en lotes.

React puede retrasar la actualización tras un

setState()

 o agrupar los cambios necesarios en lotes, por ejemplo, para renderizar varios componentes dependientes de forma simultánea. Este proceso transitorio busca un mejor rendimiento y **no garantiza un comportamiento síncrono ni un renderizado inmediato del componente.**

Esto puede derivar en que el acceso al estado inmediatamente después de invocar

setState()

 no muestre aún los cambios transferidos, pero varias de las configuraciones para

setState()

 evitarían esta situación.

## ¿Qué configuraciones admite

## setState()

## ?

El método

setState()

 admite tres posibles combinaciones de argumentos:

* Un único argumento en forma de objeto que pasaría a ser el nuevo estado del componente.
* Un único argumento en forma de función: un updater con dos parámetros por defecto (el estado actual y las props actuales) cuyo objeto retornado sería el nuevo estado del componente:

**this**.setState((state, props) => {

**return** {counter: state.counter + props.increment}

})

El objeto disponible como primer parámetro no debe ser mutado aquí: es un estado que sirve de referencia frente a cambios que dependan del propio estado antes de que haya sido actualizado, como en el ejemplo superior.

* Dos argumentos: un objeto que que pasaría a ser el nuevo estado del componente, y un callback que se invocaría una vez React haya procesado los cambios solicitados sobre el estado:

**this**.setState({counter: 10}, () => **console**.log(`Contador actualizado a ${this.state.counter}`))

¿Qué es el DOM virtual?

El DOM virtual o VDOM es un patrón de React basado en abstraer el DOM del navegador (también conocido como “DOM real” o “HTML DOM”) en una copia liviana que, almacenada en memoria, se encarga de sincronizar de forma constante con el DOM real. Esta copia virtual representa la UI.

Cuando un componente sufre un cambio en su estado, React hace uso de la API de Reconciliación para diferenciar en memoria qué ha cambiado. Este algoritmo diferencial permite detectar las variaciones, actualizando entonces el DOM real con los resultados de la comparativa.

Esta réplica virtual del DOM es conveniente en términos de agilidad frente al DOM real: carecer de los pesados detalles de implementación del navegador la hace más liviana y manejable. Básicamente, es más eficiente manipular un objeto que representa al DOM que al DOM en sí.

## ¿Qué son las referencias en React?

Las referencias o

Refs

 permiten a React **manipular directamente el DOM sin necesidad de actualizar un componente**.

Disponibles sólo en componentes de clase, asumen cambios en los mismos que no podemos resolver mediante el traspaso de props, o cuyo impacto se limita al propio componente no siendo rentable renderizarlo de nuevo.

## ¿Por qué es inviable retornar elementos adyacentes en JSX?

JSX es simplemente edulcorante sintáctico para generar elementos que, transpilados, dan lugar a su equivalencia en Javascript puro. **El código que resulta de esta fase es una representación jerárquica del componente** cuya estructura parental supone una limitación: sólo puede haber una raíz.

## React es una biblioteca o un framework?

**React es una biblioteca** que resuelve interfaces de usuario. Simplemente eso.

Una definición básica y comprensible de un framework sería un sistema donde incluyes tu código dando lugar a una aplicación. **React, en cambio, es solo una parte mas de las que componen una aplicación**.

Asume la capa de las vistas como otra de las dependencias que dan lugar a un producto.

## ¿Qué son los fragmentos?

React no permite retornar elementos adyacentes en JSX. Esto se ha resuelto históricamente anidándolos en otros elementos que resuelven el problema pero generan código residual en el DOM.

Un [fragmento](https://es.reactjs.org/docs/fragments.html) es un componente presente en React que resuelve la problemática anterior **sin agregar nodos extra** y por tanto sin impacto sobre el DOM real.

## Qué función cumplen las keys en React?

[Las keys o claves](https://es.reactjs.org/docs/lists-and-keys.html) permiten **diferenciar entre elementos únicos en el DOM virtual**. Son favorables en términos de rendimiento al limitar la actualización del componente a las partes estrictamente necesarias.

Este recurso se utiliza fundamentalmente al generar elementos en una iteración (lista) donde la

key

 de cada elemento lo identifica de manera única. De esta forma, una actualización sobre el componente permitiría diferenciar los elementos cambiantes de los inmutados. **React recicla así estos últimos** que, si bien dependen del mismo componente, no se han visto afectados.

## ¿Qué objetivo cumple ReactDOM?

[ReactDOM](https://es.reactjs.org/docs/react-dom.html) es una dependencia que actúa en el nivel superior de una SPA, conectando el ecosistema de React con el DOM del navegador.

Su función se basa en **interactuar con el navegador** para transferir al DOM real los cambios que React estipule en el DOM virtual.

## ¿Cómo limitar el alcance de los estilos en React?

Si bien React permite importar hojas de CSS ordinarias en componentes, esta tecnología tiene un alcance global que puede dar lugar a contaminación de estilos – a la vez que desaprovecha todas las funcionalidades dinámicas de una solución programática.

Algunas técnicas más avanzadas de estilizado en React delimitan el alcance de los estilos localmente, generando un namespace que da lugar a **un tratamiento estético más predecible y modular**.

## ¿Qué función cumple

## shouldComponentUpdate()

## ?

El valor boleano retornado por el método

shouldComponentUpdate()

**controlaría la actualización de un componente**, evitando su renderizado incluso frente a cambios en su estado o props.

## Qué función cumple el componente

## StrictMode

## ?

El componente

StrictMode

 permite **identificar problemas potenciales en la aplicación**.

## Deben favorecerse los componentes funcionales frente a los de estado?

Sí, siempre que sea posible. Hasta la entrada de [React Hooks](https://es.reactjs.org/docs/hooks-intro.html" \t "_blank), la carencia de estado y de métodos para controlar el ciclo de vida de los componentes funcionales los ha relegado a un uso puramente representacional.

## Qué diferencias existen entre el montaje y el renderizado de un componente?

* **Montaje**: React monta cada componente cuando este se muestra en la interfaz por primera vez. Es una fase única para cada componente. Este montaje está asociado a su método

**constructor**()

 y culmina invocando su método

componentDidMount()

.

* **Renderizado**: React renderiza un componente mediante

render()

 la primera vez que se muestra así como en respuesta a los cambios en sus props o estado. Cada renderizado culmina invocando el método

componentDidUpdate()

 correspondiente.

Los **Hooks** son funciones que te permiten “enganchar” el estado de **React** y el ciclo de vida desde componentes de función. Los **hooks** no funcionan dentro de las clases — te permiten usar **React** sin clases.