## Interviews

* Javascript is event driven (dirigido por eventos)
* Función IIFE

(function explainVar(){})()

IIFE (Inmediately Invoked Function Expression)

(function () {

statements

})();

* ¿Cuál es la diferencia entre usar == y ===?

The difference is that while with the == before making the comparison, both data are converted to a common type. With === none of these values ​​are implicitly converted before being compared.

* ¿Cuál es la diferencia entre var, let y const?

Let and const do not support hoisting

Mutable

ES5 var ES6 var, let y const.

Var: Las variables declaradas con var son procesadas antes de la ejecución del código. El scope de una variable declarada con var, es su contexto de ejecución. El scope de una variable declarada fuera de la función es global.

Ej. var i = 60;

(function explainVar(){ for( var i = 0; i < 5; i++){

console.log(i) //Output 0, 1, 2, 3, 4 }

})();

console.log("Despues del loop", i) // Output 60

En resumen, la declaración con var define una variable en el ámbito local actual (lease función) y se hereda a scopes descendientes por referencia. Si la variable es declarada fuera de una función, la variable será una variable global.

let y const son dos formas de declarar variables en JavaScript introducidas en ES6 que reducen el ámbito de la variable a bloques (con var el ámbito era la función actual) y no admiten hoisting. Además, las variables declaradas con const no pueden ser reasignadas (aunque no significa que su valor sea inmutable).

CONST: Es una constante la cual NO cambiara su valor en ningún momento en el futuro.

VAR: Es una variable que SI puede cambiar su valor y su scope es local.

LET: Es una variable que también podra cambiar su valor, pero solo vivirá(Funcionara) en el bloque donde fue declarada.

* ¿Qué es implicit coercion?

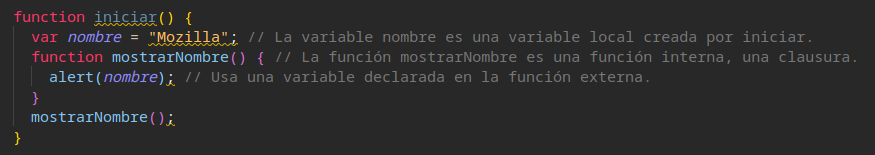
Is the process of conversion of data from one type to another

* ¿En CSS qué es especificidad y cuáles son las reglas para determinarla?

Specificity is how browsers decide which CSS property values ​​are most relevant to an element: Type selectors (e.g., h1) and pseudo-elements (e.g., ::before). Class selectors (e.g., .example), attribute selectors (e.g., [type="radio"]), and pseudo-classes (e.g., :hover). ID selectors (e.g., #example).

* Qué es un closure?

A closure is a function that stores references to the adjacent state (lexical scope). In other words, a closure allows access to the scope of an outer function from an inner function. In JavaScript, closures are created every time a function is created.



* ¿Qué es una pseudo-clase? A CSS pseudo-class is a keyword added to selectors that specifies a special state of the selected element. For example, :hover will apply a style when pass the pointer over the element specified by the selector.
* ¿En JS, qué es un efecto secundario o un side effect?

A "side effect" is any effect other than that return value.

Hacer llamadas a la API

Math.random()

* ¿Qué es el HTML semántico y cuáles son algunos de sus beneficios?

HTML nos brinda una serie de etiquetas con mayor significado, para cada parte, sección, o elemento de nuestra página, y que, aunque en la práctica no generen un resultado distinto al de usar una etiqueta como contenedor para todo, pueden darle mayor significado a nuestro código Código más claro y fácil de mantener Ayuda a tu sitio a ser accesible Mejora tu posicionamiento SEO

HTML gives us a series of tags with greater meaning

<article>

<aside>

<details>

<figcaption>

<figure>

<footer>

<header>

<main>

<mark>

<nav>

<section>

<summary>

<time>

* ¿En HTML, cuál es la diferencia entre un elemento en linea y un elemento en bloque?

A line tag is one that occupies the minimum necessary space horizontally, and allows another element to be placed next to it. A block tag, on the other hand, occupies the entire available width and does not allow another element to be placed next to it

Etiquetas de línea (las más usadas): <a>, <span>, <strong>, <img>, <input>, <code>

Etiquetas de bloque (las más usadas):<h1>, <h2>, <h3>, <p>, <ul>, <li>, <div>, <header>, <nav>, <section>, <article>, <footer>, <form>, <table>

* ¿Cómo funciona la herencia de Prototipos en JS?

https://www.tutorialsteacher.com/javascript/prototype-in-javascript

* ¿Qué es un pseudo-elemento?

pseudo-elements are added to selectors, they do not describe a special state, but instead allow styling to be added to a specific part of the document.

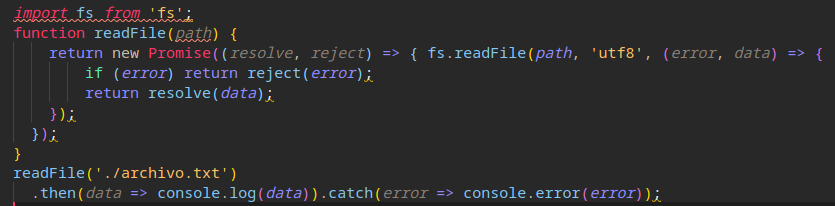
Por ejemplo, el pseudoelemento ::first-line selecciona solo la primera línea del elemento especificado por el selector.

* ¿Qué es hoisting?

a strict definition of hoisting suggests that variable and function declarations are physically moved to the beginning of the code

* ¿Cuál es la diferencia entre un callback y una promesa?

https://platzi.com/blog/que-es-y-como-funcionan-las-promesas-en-javascript/



Una Promise (promesa en castellano) es un objeto que representa la terminación o el fracaso de una operación asíncrona. Esencialmente, una promesa es un objeto devuelto al cuál se adjuntan funciones callback, en lugar de pasar callbacks a una función.

Un callback (llamada de vuelta) es una función que recibe como argumento otra función y la ejecuta.

* ¿Qué es un middleware?

A middleware is a block of code that is executed between the request made by the user (request) until the request reaches the server.

* ¿Qué es el session storage?

LocalStorage and sessionStorage, are properties that access the browser's Storage object and have the function of storing data locally

* ¿Qué es el package.json?

De cierta forma, podemos considerar este package.json como un manifiesto de nuestro proyecto.

* ¿Qué es una función pura?

In programming, Pure Functions are those that meet two basic requirements:  
Given input parameters of identical value, the function will always return the same result.  
The computation of the function, its logic, does not imply any collateral observable effect outside of it.

* ¿Qué es el virtual DOM?

The DOM (whose acronym is "Document Object Model") is the structure of objects generated by the browser when a document is loaded  
The Virtual DOM is an in-memory representation of the real DOM that acts as an intermediary between the state of the application and the DOM of the graphical interface that the user is seeing.

* Primary Components

There are three primary categories of components in React Router:

routers, like <BrowserRouter> and <HashRouter>

route matchers, like <Route> and <Switch>

and navigation, like <Link>, <NavLink>, and <Redirect>

We also like to think of the navigation components as “route changers”.All of the components that you use in a web application should be imported from react-router-dom.

* ¿Qué es un API?

Una API (siglas de ‘Application Programming Interface’) es un conjunto de reglas (código) y especificaciones que las aplicaciones pueden seguir para comunicarse entre ellas: sirviendo de interfaz entre programas diferentes de la misma manera en que la interfaz de usuario facilita la interacción humano-software.

* ¿Qué es prop drilling y cómo se puede evitar?
* ¿Qué es inmutabilidad y por qué es importante en React?

En primer lugar, lo inmutable es lo opuesto a mutable, y mutable significa cambiante, modificable … capaz de ser molestado.

Entonces, algo que es **in**mutable, entonces, es algo que no se puede cambiar.

Una princesa besa a una rana con la esperanza de que se convierta en un apuesto príncipe. El concepto de inmutabilidad establece que una rana siempre será una rana.

Array.sort() muta

# Array.prototype.filter() no muta, devuelve un nuevo array

Immutability gives you instant tighter control over your data and makes your code more secure and predictable.

<https://cacrmsoftware.com/por-que-es-tan-importante-la-inmutabilidad-en-javascript/>

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

A series of functions that are executed at different moments of the component's life and allow us to execute different actions at these moments.

El ciclo de vida se puede dividir en 3 fases, el montado, actualización y desmontado del componente. Estas fases a su vez se dividen en varios métodos que puede tener el componente.

componentWillMount()

render()

componentDidMount()

componentWillReceiveProps(nextProps)

shouldComponentUpdate(nextProps, nextState)

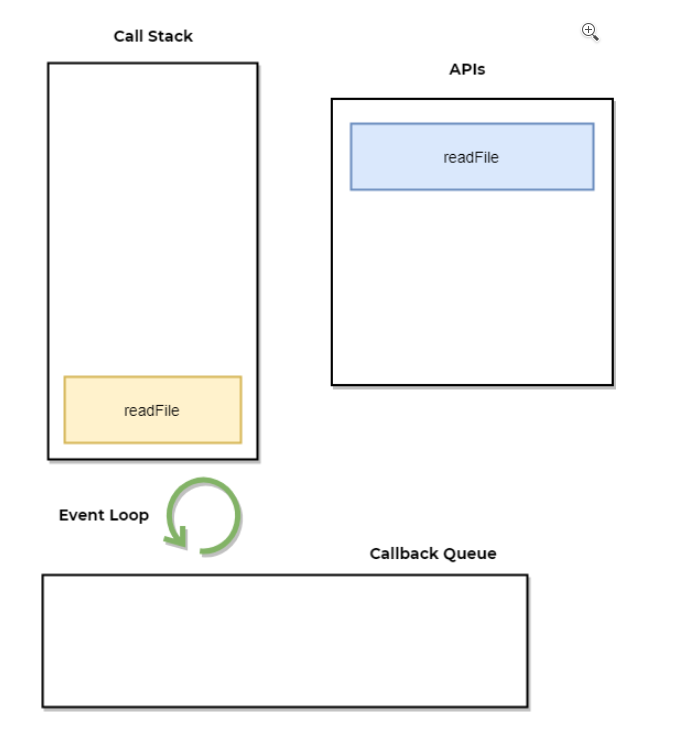
componentDidUpdate(prevProps, prevState)

componentWillUnmount()

* Que es el Event Loop?

http://latentflip.com/loupe

https://www.youtube.com/watch?v=EI7sN1dDwcY



Cuando se ejecuta el código y en que orden

<https://medium.com/@_ferh97/nodejs-y-el-event-loop-21b33fea6b03>

* ApiRest JSON

https://www.redhat.com/es/topics/api/what-is-a-rest-api

### REST

REST no es un protocolo ni un estándar, sino que se trata de un conjunto de principios de arquitectura.

* Diferencia npm npx

NPM es un gestor de paquetes de Node.js cuyo objetivo es gestionar los paquetes y las dependencias. Significa que podemos determinar en un fichero los paquetes que necesitamos (package.json) y luego instalar las dependencias que hay en él (npm install).

NPX es una herramienta para ejecutar paquetes de Node y viene incluido en la instalación de NPM desde la versión 5.2. NPX funciona de la siguiente manera: Comprueba si el paquete a ejecutar está instalado Si no está instalado lo instala Si está instalado lo ejecuta Con el parámetro --no-install podemos indicarle a NPX que si el paquete no existe no lo instale.

* Diferencias entre ES5 y ES6.

Arrow function

Asignación “clave:valor” en un objeto En los objetos, no siempre es necesario poner la clave:valor.

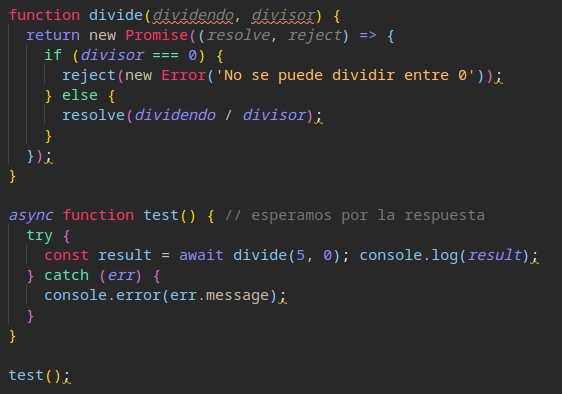
Valores por defecto en las funciones

Importación de JS require ES5 import ES6

## Concatenación de cadenas let concatName = `${name1} ${name2}`;

<https://medium.com/@jagogutierrez/ecmascript-es6-diferencias-notables-al-es5-83d3e33ae201>

* Promises



* Diferencias function y arrow function <https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Funciones/Arrow_functions> Una de las razones por las que se introdujeron las funciones flecha fue para eliminar complejidades del ámbito (this) y hacer que la ejecución de funciones sea mucho más intuitiva.
* El modo estricto de JavaScript: qué es y para qué sirve

https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Strict\_mode

Una forma de trabajar con JavaScript bastante común en código profesional es utilizar siempre por defecto el Modo Estricto de este lenguaje. Se trata de una variante del lenguaje que es menos permisiva con ciertos tipos de comportamientos en el código y que hace que éste se comporte de un modo más estricto, como su propio nombre indica.

# Qué es NodeJS y para qué sirve

Node.js es un entorno de tiempo de ejecución de JavaScript (de ahí su terminación en .js haciendo alusión al lenguaje JavaScript). Este entorno de tiempo de ejecución en tiempo real incluye todo lo que se necesita para ejecutar un programa escrito en JavaScript.

Lo transformaron de algo que solo podía ejecutarse en el navegador en algo que se podría ejecutar en los ordenadores como si de aplicaciones independientes se tratara.

* Que es react

React es una librería Javascript focalizada en el desarrollo de interfaces de usuario.

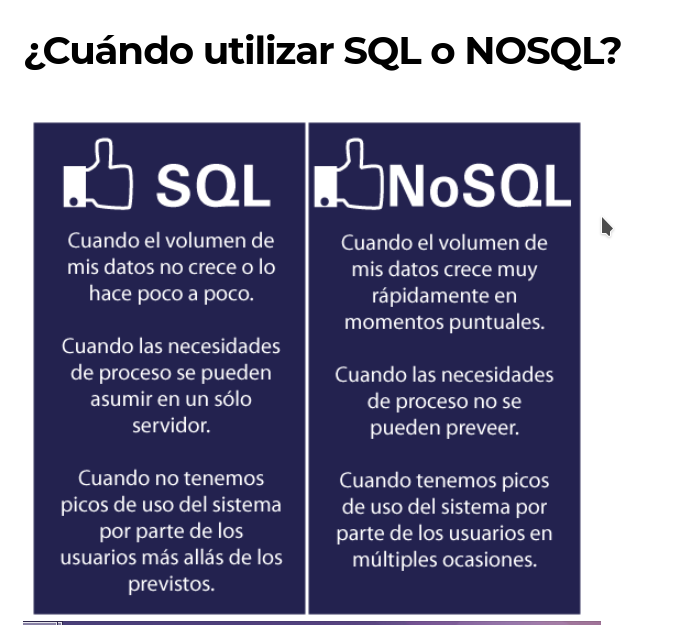
[React](https://es.reactjs.org/)está basado en un paradigma llamado programación orientada a componentes en el que cada componente es una pieza con la que el usuario puede interactuar. Estas piezas se crean usando una sintaxis llamada JSX permitiendo escribir HTML (y opcionalmente CSS) dentro de objetos JavaScript.  
Estos componentes son reutilizables y se combinan para crear componentes mayores hasta configurar una web completa.  
Esta es la forma de tener HTML con toda la funcionalidad de JavaScript y el estilo gráfico de CSS centralizado y listo para ser abstraído y usado en cualquier otro proyecto.

* Que es un framework

En el ámbito del desarrollo del software, el término “**JavaScript Framework”** significa una biblioteca que proporciona a los desarrolladores plantillas preconstruidas y código JavaScript preescrito para tareas de programación estándar. Millones de desarrolladores usan este tipo de frameworks para acelerar el flujo de trabajo de desarrollo y aplicar las mejores prácticas de una manera fluida y fácil.

Debido a su gran variedad su uso depende de los objetivos principales, la funcionalidad general de la plataforma, los requisitos del proyecto y cómo se puede implementar dentro de cada escenario específico.

Al usar este tipo de marcos desarrollo para JavaScript, se puede ahorrar una gran cantidad de tiempo y esfuerzo en el desarrollo de sitios web y aplicaciones basados ​​en este lenguaje. Simplifica todo el procedimiento y permite a los desarrolladores crear aplicaciones web a gran escala de manera eficiente.



Tanto SQL como NoSQL son tipos de bases de datos recomendadas para utilizar a la hora de comenzar con tus proyectos, cada una de ellas con ventajas y desventajas.  
  
Por ejemplo los sistemas contables, o de inventario, son sistemas que requieren transacciones de varias filas, para este tipo de trabajos la mejor opción son las bases de datos SQL (MySq). Si los sistemas son de gestión de contenido, aplicaciones móviles, sistemas de análisis en tiempo real, bases de datos con un crecimiento rápido, con un esquema descentralizado, la mejor opción son bases de datos NOSQL (MongoDB).

Algunas de las ventajas de SQL: mayor soporte y más variedad de herramientas debido a que lleva más tiempo en el mercado, es útil para manejar y obtener los datos, permite agregar otros servidores de SQL, por ejemplo una persona puede acceder a la base de datos de otra.  
  
Como desventajas de SQL están: No es flexible (antes de ingresar los objetos, deben estar correctamente validados), mientras más compleja sea la base de datos, requiere mayor procesamiento y eso se puede ver reflejado en el rendimiento y consumo de recursos.  
  
Ahora, conozcamos algunas de las ventajas de las bases de datos NOSQL: permite una alta escalabilidad (ayuda a reducir la carga de trabajo), flexible a diferentes tipos de datos, los datos deben cumplir con el tipo de dato definido, y algunas desventajas: la integridad de los datos se afecta por el poco soporte, menos seguridad al ejecutar consultas, no existe estandarización, en la mayoría de los casos son poco compatibles con las bases de datos SQL, casi todo su mantenimiento se debe realizar por consola debido a que existen pocas herramientas.

* Javascript 2021

https://www.arsys.es/blog/javascript-2021/

[**Spread (…)**](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Spread_syntax)

[**Nullish Coalescing (??)**](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Nullish_coalescing_operator)

[**Optional chaining (?.)**](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Optional_chaining)

HTTTP errors

1xx - Informational: The server has received the request and is continuing the process

2xx - Successful: The request was successful and the browser has received the expected information , 200Ok, 201Created

3xx (Redirection): You have been redirected and the completion of the request requires further action

4xx (Client Error): The website or the page could not be reached, either the page is unavailable or the request contains bad syntax , 400Bad request, 401Unauthorized

5xx (Server Error): While the request appears to be valid, the server could not complete the request