# **Promesas**

Una [Promise](https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global_Objects/Promise) 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.

Diagrama

Descripción generada automáticamente

Las promesas en Javascript se representan a través de un objeto, y cada promesa estará en un estado concreto: pendiente, aceptada o rechazada. Además, cada promesa tiene los siguientes métodos, que podremos utilizar para utilizarla:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Obsérvese además que hemos añadido el método. finally() para añadir una función callback que se ejecutará **tanto si la promesa se cumple o se rechaza**, lo que nos ahorrará tener que repetir la función en el .then() como en el .catch().

**FLUJO DE UNA PROMESA**

Cuando una promesa es disparada esta ingresa a un estado la cual puede ser las siguientes:

**fulfilled**: se cumplió con éxito.

**rejected**: no se pudo cumplir a casa de un error. Depende de ustedes como hagan el reject.

**pending**: se está procesando.

**settled**: simplemente finalizó.

Ejercicio #1:

Texto

Descripción generada automáticamente

Ejercicio #2

Texto

Descripción generada automáticamente

Imagen que contiene Escala de tiempo

Descripción generada automáticamente

**Ejercicios:**

1. Crear un algoritmo que lea el nombre de un usuario y lo imprima.
2. Crear un algoritmo que le pida al usuario 3 números, calcular la suma de los 3 números e imprimir el resultado si es mayor que 25
3. Crear un algoritmo que permita leer la edad y peso de una persona y posteriormente imprimirla, si el peso es superior a 100 imprimir que el usuario tiene sobre peso

# **Movies App**

Pantalla de un video juego

Descripción generada automáticamente

Crear la estructura básica de html

Vincular los archivos externos (script, css)

Texto

Descripción generada automáticamente

Agregar la cabecera de la página con un buscador

Texto

Descripción generada automáticamente con confianza bajaTexto

Descripción generada automáticamente con confianza baja

Para el contenido principal de la aplicación vamos a crear un selector de tipo clase con el nombre movie, agregar una imagen

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Agregar información referente a la película

Texto

Descripción generada automáticamente

Agregar estilos básicos:

https://image.winudf.com/v2/image1/Y29tÑ,1vdllc2

Fuente: Roboto

Texto

Descripción generada automáticamente

Resultado:

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente

Crear variables en el archivo css

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Usar variable

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Resultado:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Agregar estilos a la etiqueta search



Texto

Descripción generada automáticamente

Agregar más estilos:Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamenteTexto

Descripción generada automáticamente

Agregar pseudoelementos y pseudoclases

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente

Darle estilos a la estructura de la película

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Agregar estilos de la etiqueta span

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Icono

Descripción generada automáticamente

Duplicar la estructura del html para las diferentes calificaciones de categorías de películas

cada película tiene una calificación la cual la vamos catalogar en green, orange, red



Agregar estilos según el selector tipo clase del span

Texto

Descripción generada automáticamente

Resultado:

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

Darle estilos a la información de la película Pantalla de un video juego

Descripción generada automáticamente con confianza mediaTexto

Descripción generada automáticamente

Agregar una transformación para observar el contenido de la película

Texto

Descripción generada automáticamente

Con la propiedad hover se desplaza el contenido de la película

Interfaz de usuario gráfica

Descripción generada automáticamente

Agregamos una transition para desplazar el contenido en el eje y

**Ejercicios:**

1. Leer y profundizar las temáticas que no tenga claro o le de dificultad implementarlo
2. Agregar el título de **Movie** en el header
3. Agregar el mismo espacio entre todos los componentes (usar propiedades css de flex)
4. Revisar el contenido del Themoviesdb

Themoviesdb:

<https://www.themoviedb.org/?language=es>

<https://developers.themoviedb.org/3/getting-started/introduction>

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

# **En el archivo script.js**

const API\_URL = 'https://api.themoviedb.org/3/discover/movie?sort\_by=popularity.desc&api\_key=3fd2be6f0c70a2a598f084ddfb75487c&page=1'

const IMG\_PATH = `https://image.tmdb.org/t/p/w1280`

const SEARCH\_URL = 'http://api.themoviedb.org/3/search/movie?api\_key=3fd2be6f0c70a2a598f084ddfb75487c&query="'

Crear variables con los elementos de HTML necesarios

Texto

Descripción generada automáticamente

# **Consumir Promesas**:

Crear una función para consumir el api, todo este proceso es de forma sincrónico 

En la función para obtener las películas invocamos una función de mostrar películas, el cual necesita parámetro para poder ser ejecutada. En la función showMovies mandamos la información que está llegando desde la url del API.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

El resultado de la data que llega por la solicitud al servidor de themoviedb, nos permite observar que los datos vienen internamente él un objeto llamado results.

Para consumir los datos del objeto results debemos navegar por niveles por medio del signo punto(.) como lo vemos en la imagen.

Otra forma de obtener el mismo resultado es usando. **then** y obtenemos el mismo resultado

Texto

Descripción generada automáticamente

Como ven en la imagen se logra la data es perada, pero este código no está limpio, no es una forma estándar de trabajar.

Texto

Descripción generada automáticamente

La solución a la problemática anterior, la resolvemos con peticiones anidadas como se ve en la imagen.

Si la información sigue estando en objetos más internos, seguimos utilizando el punto hasta el nivel que necesitemos.



Eliminar la estructura de la película en el index.html, para agregar todo este contenido desde el archivo script.js

Texto

Descripción generada automáticamente

Borrar el contenido del interior de la etiqueta main

Agregar la estructura de la película desde el archivo de JavaScript

1. Limpiar el contenido de la etiqueta main

Texto

Descripción generada automáticamente

1. Recorrer el array de películas

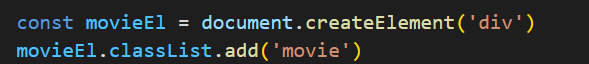
Texto

Descripción generada automáticamente con confianza media

1. Cada vez que se recorra la información de la película vamos a crear una película con la estructura que habíamos realizado en html
2. Para consumir la información debemos desestructurar los datos de las películas



1. Crear una etiqueta para ser insertada en el DOM
2. Agregar una clase movie para darle el formato que se asignó para esta sesión en el archivo css



1. En el contenido del <div> que creamos vamos a enviar la estructura de la película

Texto

Descripción generada automáticamente

Para finalizar debemos agregar la etiqueta al archivo HTML



Organizar el contenido para consumir la información de las películas

La estructura completa para obtener la información de las películas

Texto

Descripción generada automáticamente

Resultado en el navegador

Pantalla de juego de video

Descripción generada automáticamente con confianza media

Para darle funcionalidad al buscador, debemos agregar una función para escuchar cuando se ejecute el evento de submit.

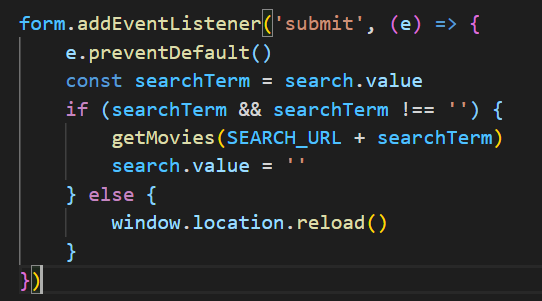
Para evitar que la aplicación se ejecute de nuevo, vamos a prevenir que ese evento se ejecute



Capturamos la información de ese input de búsqueda



Validamos si tenemos información para hacer la búsqueda, de lo contrario se vuelve a recargar la pagina



Realizar las validaciones respectivas en la App

Instalar sweetalert a la aplicación, ir a la siguiente página <https://sweetalert2.github.io/#download>

Texto

Descripción generada automáticamente con confianza media

La instalación la vamos a realizar por medio del CDN

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

Agregar esta etiqueta al archivo HTML

Texto

Descripción generada automáticamente

En la función donde obtenemos la data, es conveniente validar si la información se genera correctamente, si surge un error se activa una alerta de la librería de sweetalert

Pantalla de computadora con letras

Descripción generada automáticamente con confianza media

Interfaz de usuario gráfica, Aplicación, Teams

Descripción generada automáticamente

**Ejercicio:**

1. Consumir el API de Rick and Morty e imprimirlas como se ve en la imagen.

<https://rickandmortyapi.com/api/character>

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

2). Consumir el API de Pokemon <https://pokeapi.co/>