Introducción a JavaScript

Introducción a JavaScript 1
- ¿Qué aprenderás? 2
- Introducción 2
- ¿Qué es JavaScript? 3
  - Algunos ejemplos de JavaScript 4
- Agregando JS 5
  - Inspector de elementos 5
  - Desde el mismo archivo HTML 6
  - Desde un archivo JS 7
- Console log 8
- Declaraciones y sintaxis 8
  - Sintaxis JavaScript 9
    - Valores 9
    - Operadores 11
    - Expresiones 11
    - Palabras Clave 11
    - Comentarios 12
- Ejercicio guiado: Primeros pasos con JavaScript 13
  - Contexto 13
- Resumen 17

¡Comencemos!

¿Qué aprenderás?

- Implementar scripts locales y externos en un documento HTML, para integrar JavaScript a un proyecto web.
- Emplear declaraciones, valores, expresiones y operadores, de acuerdo a la sintaxis de JavaScript.

Introducción

Hasta ahora, hemos ido dando estructura y estilo a nuestro proyecto. Esos cambios que fuimos agregando: los colores, imágenes y textos, los cuales hicieron que nuestro proyecto se viera cada vez más atractivo y profesional.

En las siguientes páginas, conoceremos conceptos básicos de JavaScript, los cuales nos permitirán agregar los archivos de script a nuestros proyectos y ejecutar pruebas de código de manera local en nuestro navegador, para que finalmente, logremos construir páginas dinámicas, agregando efectos visuales e interacción con el usuario.

¡Vamos con todo!

¿Qué es JavaScript?

Es un lenguaje de programación interpretado, es decir, el navegador lee directamente las instrucciones sin la necesidad de terceros.

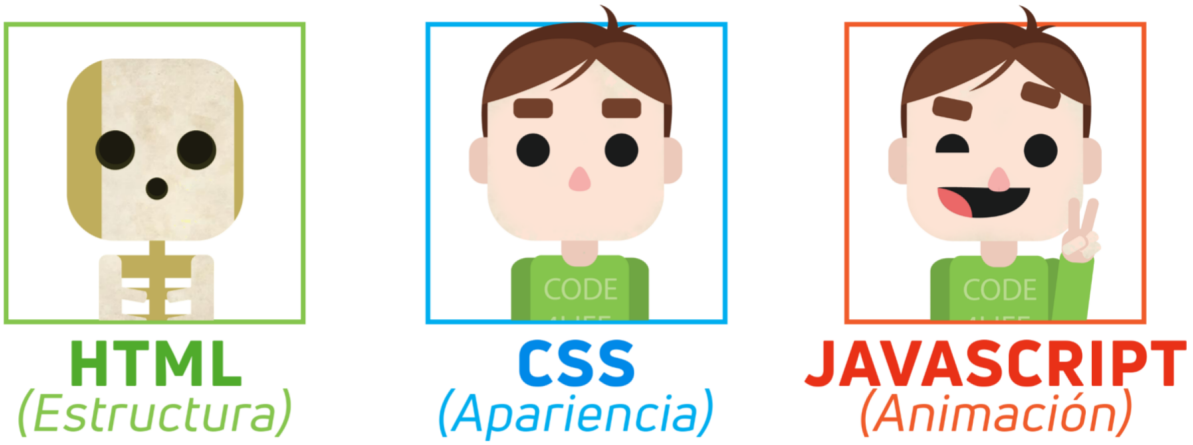

Imagen 1. Tipos de lenguajes de programación.
Fuente: Desafío Latam.

- HTML sería el esqueleto del personaje, la estructura.
- CSS serviría para darle apariencia al personaje.
- JavaScript, sería la animación del personaje, el comportamiento o la interacción que este pueda tener.

En el desarrollo de sitios web, la interactividad y el dinamismo es clave. JavaScript nos ayuda a generar interacción con el usuario, efectos de estilo dinámicos, animaciones y mucho más.

En resumen, JavaScript es un lenguaje mediante el cual se crean documentos que contienen instrucciones (scripts), las cuales se ejecutan cuando se carga una página web (al leer la página) o cuando se produce un suceso determinado (eventos, como hacer click sobre un vínculo). Estos scripts permiten crear páginas dinámicas.

Algunos ejemplos de JavaScript

En la web, puedes encontrar ejemplos de sitios utilizando JavaScript, veamos algunos:

1. filippobello.
2. promoflex.
3. legworkstudio.
4. Cualquier sitio que contenga Bootstrap está utilizando JavaScript. Este framework utiliza una biblioteca llamada jQuery, que nos permite entregar el dinamismo y los efectos visuales a sus componentes.

Veamos el siguiente ejemplo, puedes visitar la página en este link:

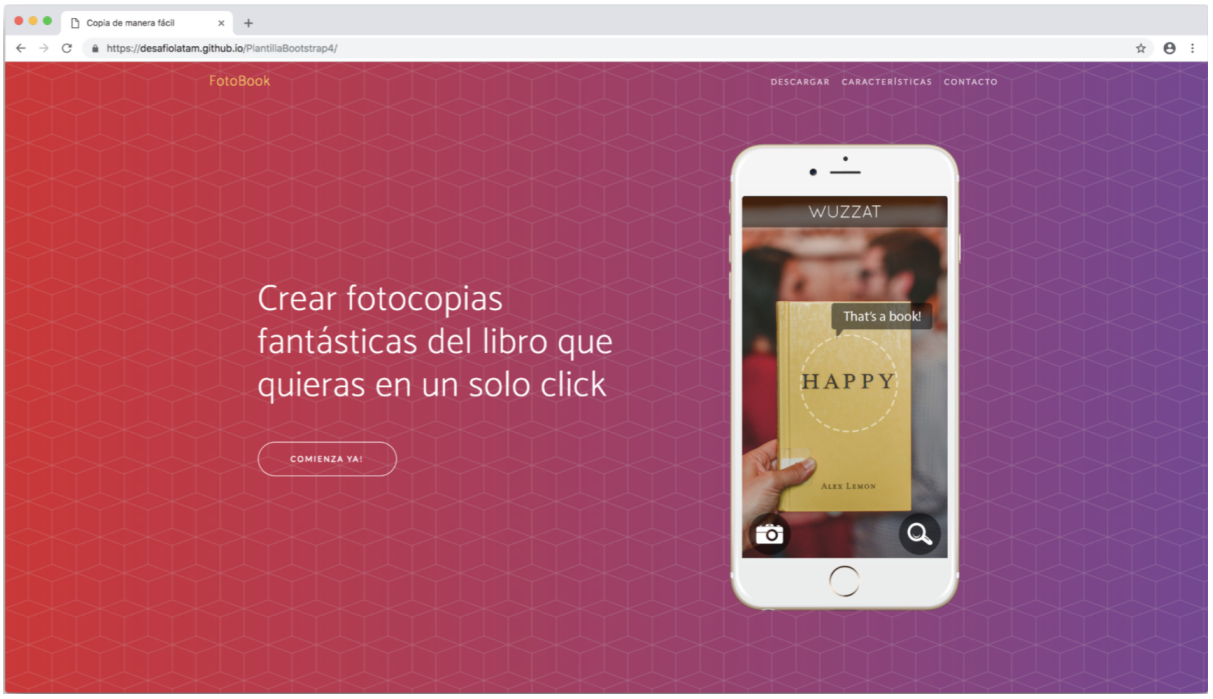

Imagen 2. Ejemplo de Bootstrap con JavaScript y jQuery.
Fuente: Desafío Latam.

En esta página, todos los efectos, cambios de color en los botones y los desplazamientos se dan gracias a los elementos de JavaScript y jQuery que contiene Bootstrap, según lo aprendido en la unidad anterior.

Agregando JS

Para comenzar a trabajar con JavaScript, veamos cómo integrar los archivos de script en nuestros proyectos. Existen diversas formas, las cuales revisaremos a continuación:

a. Inspector de elementos

La primera forma nos será muy útil al momento de realizar pruebas sencillas de código. Para eso, debemos abrir el navegador e ingresar al inspector de elementos.

En la pestaña `console`, podremos ejecutar código JavaScript. Por ejemplo, al ejecutar el código: `alert("Esta prueba es desde la consola");` se mostrará lo siguiente:

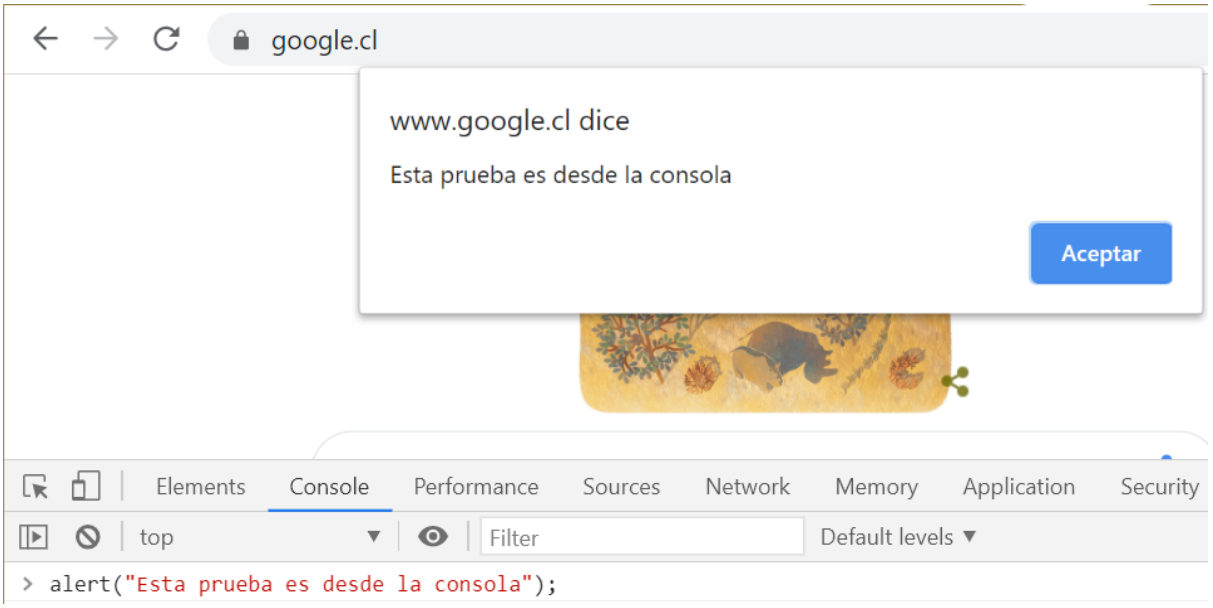

Imagen 3. Mensaje de alerta en navegador.
Fuente: Desafío Latam.

Como vemos, esto arrojará un mensaje de alerta en nuestro navegador con el mensaje descrito.

b. Desde el mismo archivo HTML

Si bien la ejecución por consola es útil para realizar pruebas, sabemos que esta información no se guarda en ningún archivo. Para hacer esto, una de las opciones que tenemos es incluir el código en una etiqueta `<script> ... <script>`. Esta podemos colocarla tanto dentro de la etiqueta `<head>` como de la etiqueta `<body>`.

Como vimos en la unidad de Bootstrap, el documento se carga desde arriba hacia abajo, por lo tanto, nos será conveniente poner esta etiqueta al final del `<body>` para que la página en total se pueda cargar más rápido. Por ejemplo:

```
<script>
  alert("Esta prueba es desde la etiqueta script");
</script>
```

Este código, ejecutado en el navegador, dará el siguiente resultado:

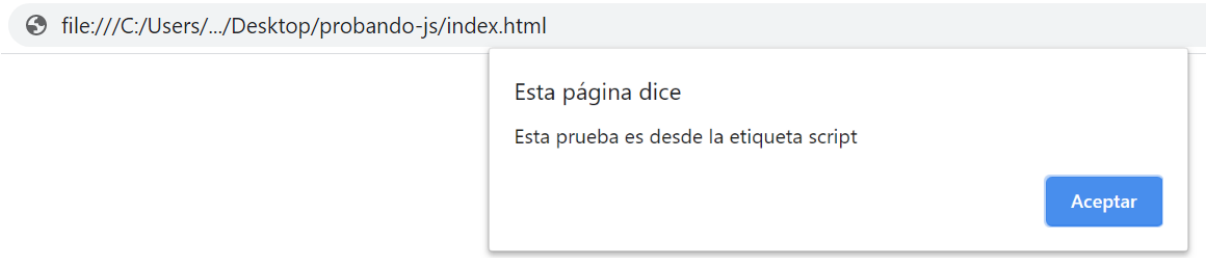

Imagen 4. Mensaje de alerta en archivo HTML.
Fuente: Desafío Latam.

c. Desde un archivo JS

También podemos escribir JavaScript en un archivo externo al HTML. Para vincularlo a la etiqueta `<script>` le debemos añadir el atributo `src` con valor de la ruta del archivo. Este puede ser un archivo que tengamos dentro de la carpeta de nuestro proyecto, como también un archivo llamado desde el CDN.

Por ejemplo, el siguiente código buscará el archivo script.js en la ruta assets/js del proyecto.

En el documento HTML, escribimos el atributo `src` con el valor de la ruta del archivo:

```
<body>
  <script src="assets/js/script.js"></script>
</body>
```

Al ejecutar el código anterior, muestra una alerta en el navegador:

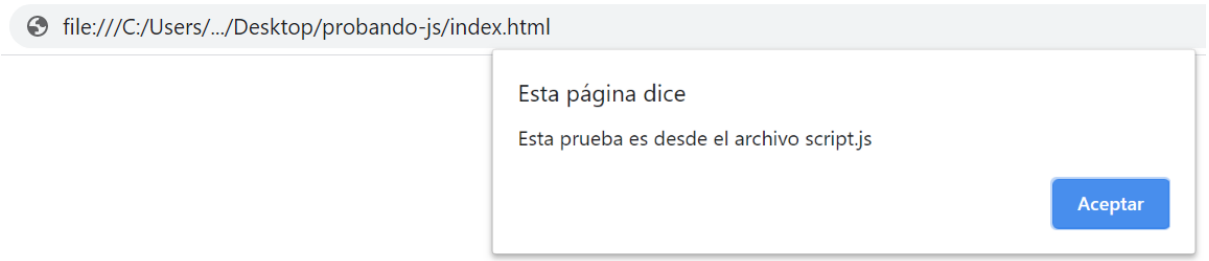

Imagen 5. Mensaje de alerta vinculado a etiqueta script.
Fuente: Desafío Latam.

Esta última es la que más utilizaremos, ya que hacerlo en un archivo externo al HTML nos permitirá reutilizarlo en otros archivos HTML del sitio sin tener que escribirlo una y otra vez en la etiqueta `<script>`.

Console log

Existe otra función que aprenderemos, el llamado `console.log();`. Con él se puede imprimir lo que necesitemos, pero en la consola. Veamos un ejemplo:

```
console.log("Estamos probando mensajes en la consola");
```

Nos mostrará el siguiente mensaje:

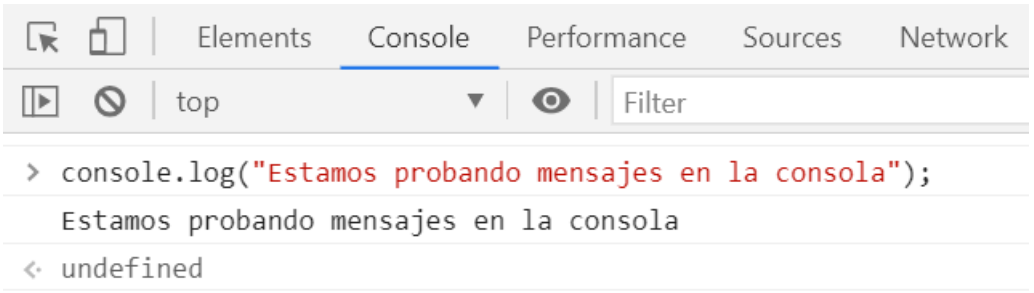

Imagen 6. Utilizando la función console.log().
Fuente: Desafío Latam.

Observemos que lo que despliega no es una ventana emergente, sino que el mensaje se despliega en la misma consola.

La función `console.log();` nos servirá más adelante para hacer debug. Por ahora, nos ayudará a mostrar diversos resultados.

Declaraciones y sintaxis

Antes de seguir escribiendo código JavaScript, debemos entender lo que estamos haciendo:

- Un programa es una lista de "instrucciones" para ser "ejecutadas" por un computador.
- En un lenguaje de programación, estas instrucciones de programación se denominan declaraciones.
- Un script de JavaScript es una lista de declaraciones.

Es decir, en los ejemplos anteriores, hicimos dos declaraciones que el navegador ejecuta:

1. Realizar una alerta al usuario con el texto `"Esta prueba es desde el archivo script.js"`.

2. Imprimir un texto en consola con el texto `"Estamos probando mensajes en la consola"`.

Las declaraciones de JavaScript se componen de:

- Valores;
- Operadores;
- Expresiones;
- Palabras clave;
- Comentarios.

Sintaxis JavaScript

La sintaxis de JavaScript tiene sus propias reglas, pero dependerá del tipo de declaración que corresponda. Los punto y coma (;) sirven para separar las declaraciones de JavaScript. Si bien ahora ya no son 100% obligatorios, nos ayudarán a ordenar nuestro propio código y a evitar errores.

Es importante tener en cuenta los siguientes puntos en JavaScript:

- Es sensible a las mayúsculas y minúsculas;
- No toma en cuenta los espacios en blanco ni los saltos de línea.

Valores

Respecto a la declaración de tipo `"Valor"` existen dos tipos:

1. Literales;
2. Variables.

Los valores de tipo literales solo se escriben:

```
100

0.5

"Hola a todos"
```

Como se escriba, dependerá del tipo de dato, lo cual veremos en profundidad más adelante, pero es necesario comprender desde ya, que los números se pueden escribir enteros o en decimales y que los textos (`strings`) están siempre entre comillas simples o dobles.

Los valores de tipo variable se escriben así:

```
var numero = 100;

var numero2 = 0.5;

var frase = "Hola a todos";
```

Una variable se emplea para almacenar y hacer referencia a un valor. Para crear una variable se debe respetar la siguiente sintaxis:

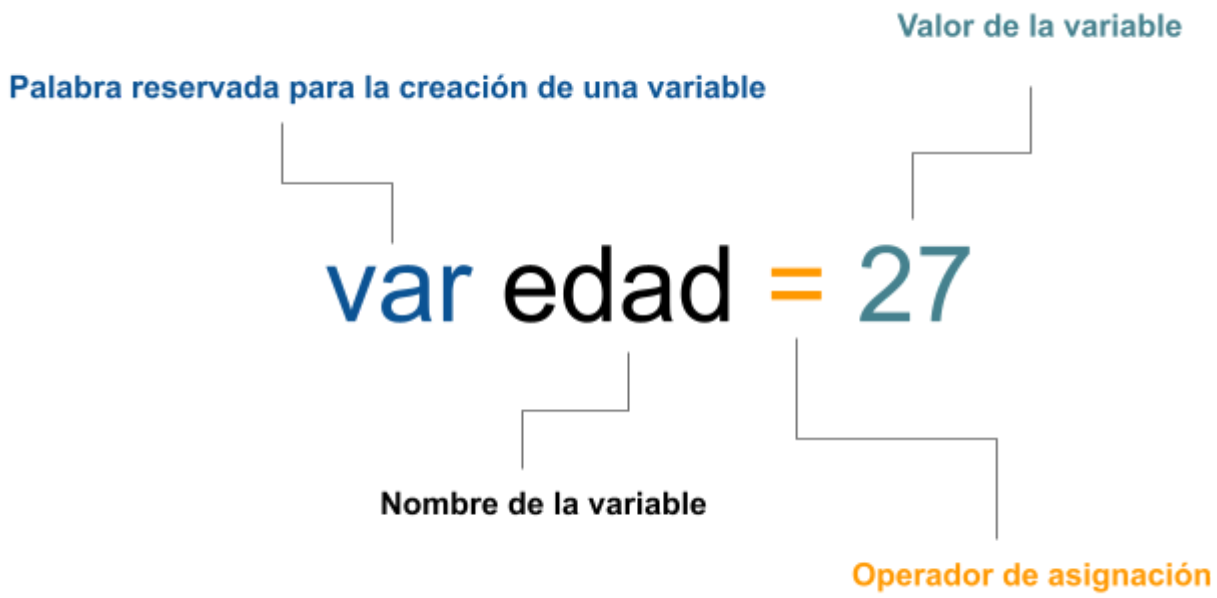

Imagen 7. Sintaxis de una variable
Fuente: Desafío Latam

Operadores

Los operadores son símbolos que ocupamos para interactuar con las variables y según el carácter o combinación de caracteres que utilicemos, podremos aplicar diferentes acciones.

Veremos 3 tipos:

1. Operadores de asignación.
2. Operadores aritméticos.
3. Operadores de comparación.

Estos operadores los veremos más adelante, en mayor profundidad.

Expresiones

Una expresión es cualquier unidad de código válida que se resuelve en un valor. Es una combinación de valores, variables y operadores, que computa un valor. El cómputo se llama evaluación. Una expresión puede contener distintos tipos de valores (literales o variables) y estos pueden ser de distintos tipos de datos. En los siguientes casos, la expresión se evalúa como 7:

```
x = 7

3 + 4
```

Palabras Clave

Las palabras clave son identificadores que tienen un significado especial en JavaScript. Las declaraciones de JavaScript a menudo comienzan con una palabra clave para identificar la acción que se realizará.

Muchas de las palabras claves están reservadas, lo que significa que **no pueden ser utilizadas como variables, etiquetas o nombres de funciones**.

En la siguiente imagen, podemos ver algunas de las palabras reservadas. Para consultar la lista actualizada, puedes ir al siguiente enlace.

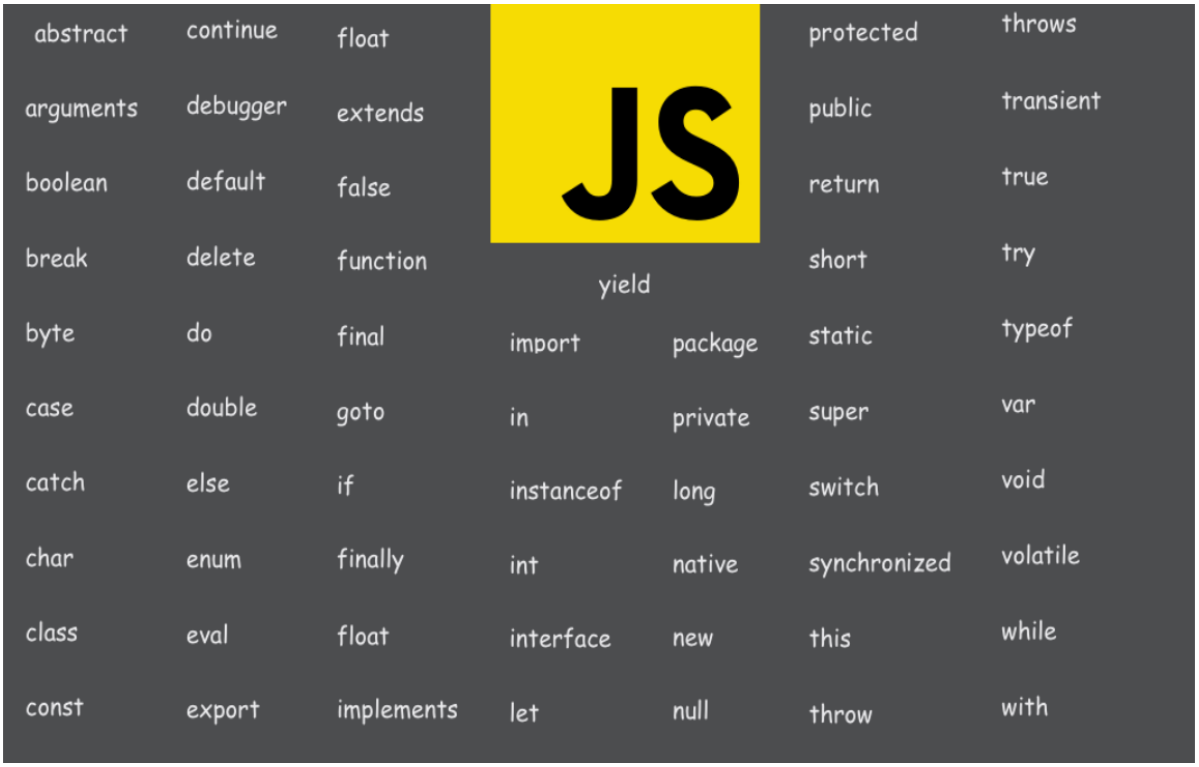

Imagen 8. Palabras reservadas.
Fuente: Codewordjs.com

Comentarios

- No todas las declaraciones de JavaScript son "ejecutadas";
- El código después de `//` o entre `/*` y `*/` se trata como un comentario;
- Los comentarios son ignorados, y no serán ejecutados;
- Los comentarios son muy útiles para documentar el código.

Antes de continuar:

¿Existe algún concepto que no hayas comprendido?

Vuelve a revisar los conceptos que más te hayan costado antes de seguir adelante.

Ejercicio guiado: Primeros pasos con JavaScript

Contexto

Como hemos visto, JavaScript nos permite agregar lógica y dinamismo a nuestros sitios. En el siguiente ejercicio, iremos experimentando con JavaScript, para ir teniendo mayor familiaridad con su declaración y sintaxis. De esta manera, iremos adquiriendo práctica, para agregar aspectos dinámicos a nuestros sitios web.

Lo primero que haremos será crear un proyecto nuevo, donde realizaremos las pruebas pertinentes para conocer el lenguaje de JavaScript, paso a paso.

Como ya lo hemos hecho durante el módulo crearemos un directorio de carpetas, con sus archivos ordenados y clasificados.

Sigue atentamente el paso a paso que te presentamos a continuación:

- ***Paso 1:*** crea una nueva carpeta donde más nos acomode. Esta se llamará `probando-js`.
- ***Paso 2:*** entra a VSC. Crea el `index.html`, en la raíz de la carpeta `probando-js`.
- ***Paso 3:*** crea la carpeta assets presionando "botón derecho" y luego nueva carpeta.
- ***Paso 4:*** dentro de assets crea la `carpeta js`.

Hacer este ejercicio por cada proyecto ayudará a tener una estructura de carpetas ordenada, como se muestra a continuación:

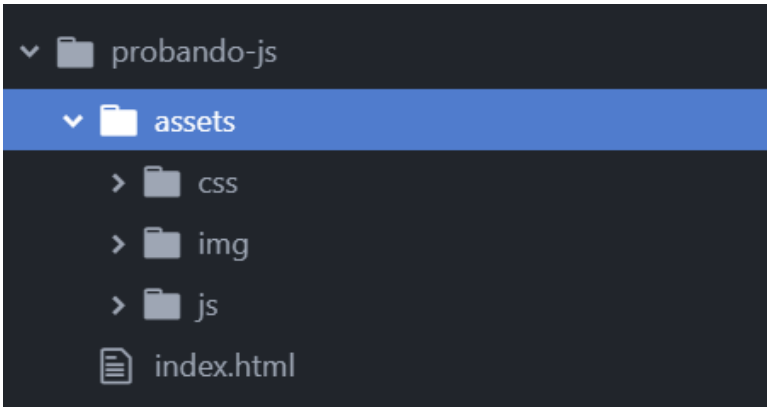

Imagen 9. Estructura de carpetas.
Fuente: Desafío Latam.

- ***Paso 5:*** ahora al index.html, dale la estructura base de un archivo HTML.

```
<!DOCTYPE html>
<html lang="es" dir="ltr">
 <head>
   <meta charset="utf-8">
   <title>Probando JS</title>
 </head>
 <body>

 </body>
</html>
```

- ***Paso 6:*** guarda y abre el archivo con el navegador Chrome. Por ahora, no mostrará nada, ya que no hay contenido, pero luego deberás ir refrescando el sitio con el código que agregues.

- ***Paso 7:*** ahora, te invitamos a experimentar JavaScript con la consola del navegador, ingresa al inspector de elementos y dirígete a la pestaña "Console".

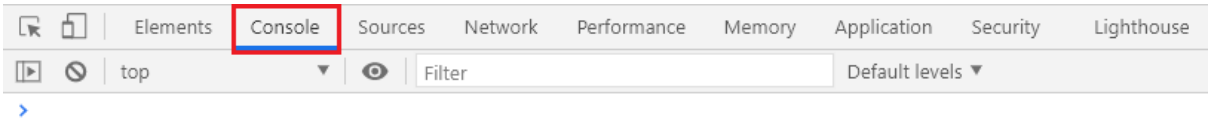

Imagen 10. Consola Chrome.
Fuente: Desafío Latam.

- ***Paso 8:*** escribe en la consola lo siguiente `alert("Estoy ejecutando la función alert");` presiona enter para ejecutarlo. Podremos observar que nos aparece la alerta:

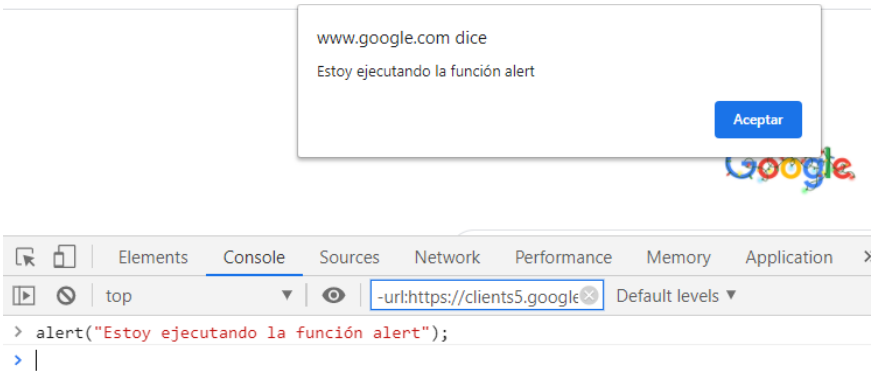

Imagen 11. Función alert().
Fuente: Desafío Latam.

Como puedes observar, se ejecuta el script inmediatamente y nos muestra un mensaje a través del navegador.

- ***Paso 9:*** Observa ahora cómo se comporta `console.log`. Escribe en la consola lo siguiente: `console.log("Estoy ejecutando la función console.log");` presiona enter para ejecutarlo. Podremos observar que aparece el mensaje, pero esta vez en nuestra consola:

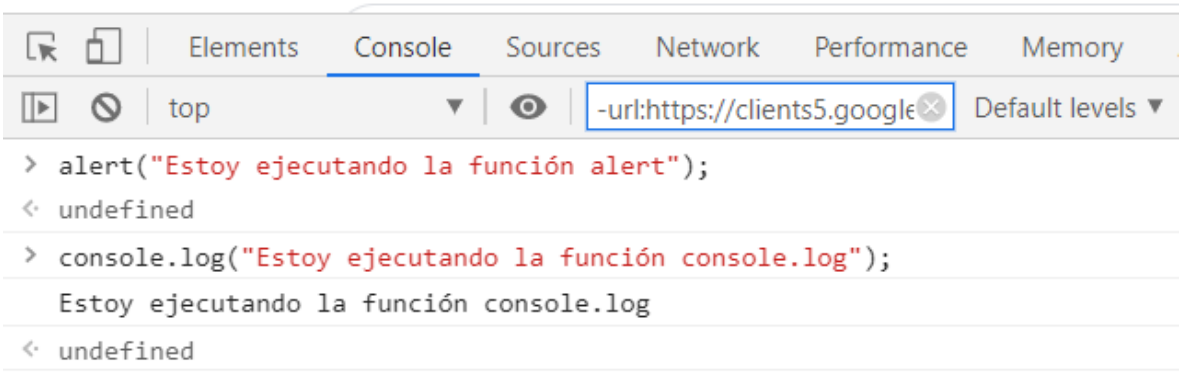

Imagen 12. Función console.log().
Fuente: Desafío Latam.

- ***Paso 10:*** realiza este mismo ejercicio, pero desde el archivo index.html. Dentro del tag `<body>` agrega dos líneas, una para la función `alert()` y otra para `console.log()`, como lo muestra el siguiente ejemplo:

```
<!DOCTYPE html>
<html lang="es" dir="ltr">
 <head>
   <meta charset="utf-8">
   <title>Probando JS</title>
 </head>
 <body>
   <script>
     alert("Estoy ejecutando la función alert");
     console.log("Estoy ejecutando la función console.log");
   </script>
 </body>
</html>
```

- ***Paso 11:*** guarda y abre el archivo en el navegador. Observa que inmediatamente muestra la alerta.

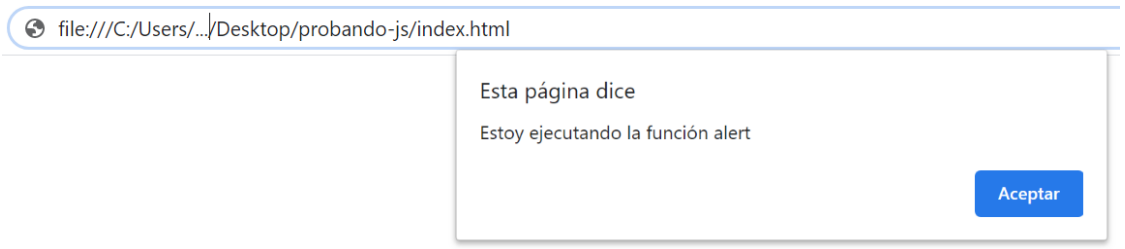

Imagen 13. Integrando JS desde el archivo HTML.
Fuente: Desafío Latam.

Una vez que haces clic en aceptar, verifica la consola. Observa que también está ahí el mensaje. Este último método es muy útil para realizar debug de nuestros programas.

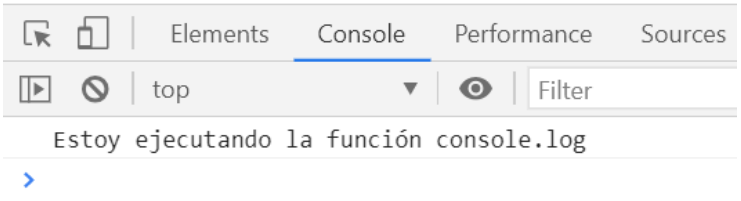

Imagen 14. Integrando JS desde el archivo HTML.
Fuente: Desafío Latam.

- ***Paso 12:*** ahora haz lo mismo, pero desde el archivo script.js. Elimina el código que acabas de agregar y agrega las referencias al archivo externo:

```
<!DOCTYPE html>
<html lang="es" dir="ltr">
 <head>
   <meta charset="utf-8">
   <title>Probando JS</title>
 </head>
 <body>
   <script src="assets/js/script.js"></script>
 </body>
</html>
```

En el archivo script.js escribe el siguiente código:

```
alert("Estoy ejecutando la función alert");
console.log("Estoy ejecutando la función console.log");
```

Observa que no es necesario volver a agregar la etiqueta `<script>`, solo agregaremos aquí el código JavaScript que queramos ejecutar.

Si guardas y vuelves a cargar el archivo, verás que mantiene el mismo comportamiento que la prueba anterior, pero ahora has llamado a un archivo como referencia desde el documento HTML.

Esta última forma es la que utilizaremos para nuestros proyectos.

Resumen

- Javascript es un lenguaje de programación interpretado, es decir, el navegador lee directamente las instrucciones sin la necesidad de terceros.
- Existen al menos 3 formas de trabajar con JavaScript: a través del inspector de elementos, desde el mismo archivo HTML y desde un archivo JS.
- La función `console.log();` permite imprimir lo que necesitemos, pero en la consola.
- Los punto y coma (;) sirven para separar las declaraciones de JavaScript.
- JavaScript es sensible a las mayúsculas y minúsculas y no toma en cuenta los espacios en blanco ni los saltos de línea.
- Una variable se emplea para almacenar y hacer referencia a un valor.
- Los operadores son símbolos con los que se pueden realizar operaciones con variables, valores literales o variables y valores literales.
- Una expresión es cualquier unidad de código válida que se resuelve en un valor.
- Las palabras clave son identificadores (token) que tienen un significado especial en JavaScript.
- Un comentario es el código después de `//` o entre `/ *` y `* /` el cual no es ejecutado.