Guía de ejercicios 1 - Introducción a JavaScript

¡Hola! Te damos la bienvenida a esta nueva guía de ejercicios.

¿En qué consiste esta guía?

En la siguiente guía podrás trabajar los siguientes aprendizajes:

- Ejecutar código JavaScript en la consola del navegador, por ejemplo revisar el valor de una variable con console.log()
- Cargar un script.
- Ejecutar operaciones matemáticas en la consola.
- Cargar un script.
- Cambiar el contenido de un elemento.
- Agregar eventos a elementos del HTML.
- Cambiar el estilo de un elemento.
- Cambiar el contenido o estilo en función del contenido ingresado en el prompt.
- Cambiar el estilo de un elemento en respuesta a un click.
- Cambiar el contenido de un elemento en respuesta a un cambio de un input.

¡Vamos con todo!

Tabla de contenidos

¡Comencemos!

¿Qué es JavaScript?

JavaScript es un lenguaje de programación que nos permite agregar interacción a una página web. Mientras que HTML y CSS se encargan del esqueleto y la apariencia, JavaScript se enfoca en la lógica detrás de la interfaz, permitiendo que el usuario interactúe con el sitio web y ésta reaccione.

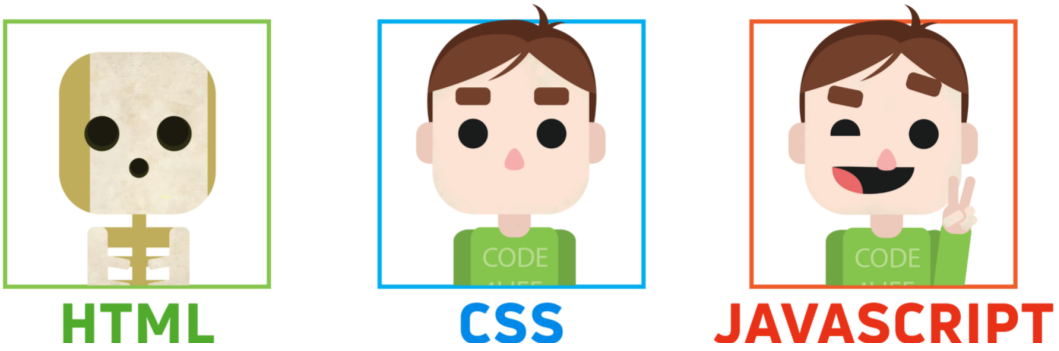

Imagen 1. Consola web
Fuente: Desafío latam

¿Qué se puede hacer con JavaScript?

Para ver ejemplos de qué se puede hacer con JavaScript, visita los siguientes sitios web y observa cómo reaccionan con movimientos del mouse y el uso del teclado:

- Kuon Yagi
- Moon Farmer
- Bruno Simon

También contamos con una serie de ejemplos de lo que podemos hacer con JavaScript disponibles en la W3School.

Comenzando con Javascript

Para tener nuestro primer encuentro con JavaScript, abriremos el navegador y accederemos al inspector de elementos. Luego, seleccionaremos la pestaña "console" del inspector de elementos y podremos ejecutar nuestro primer script.

Importante: Para este curso utilizaremos Firefox como navegador. Los elementos mostrados en el inspector de elementos cambian de navegador a navegador y la consola tiene ligeras diferencias, es posible realizar los ejercicios con Chrome o Safari, pero la ejecución será distinta.

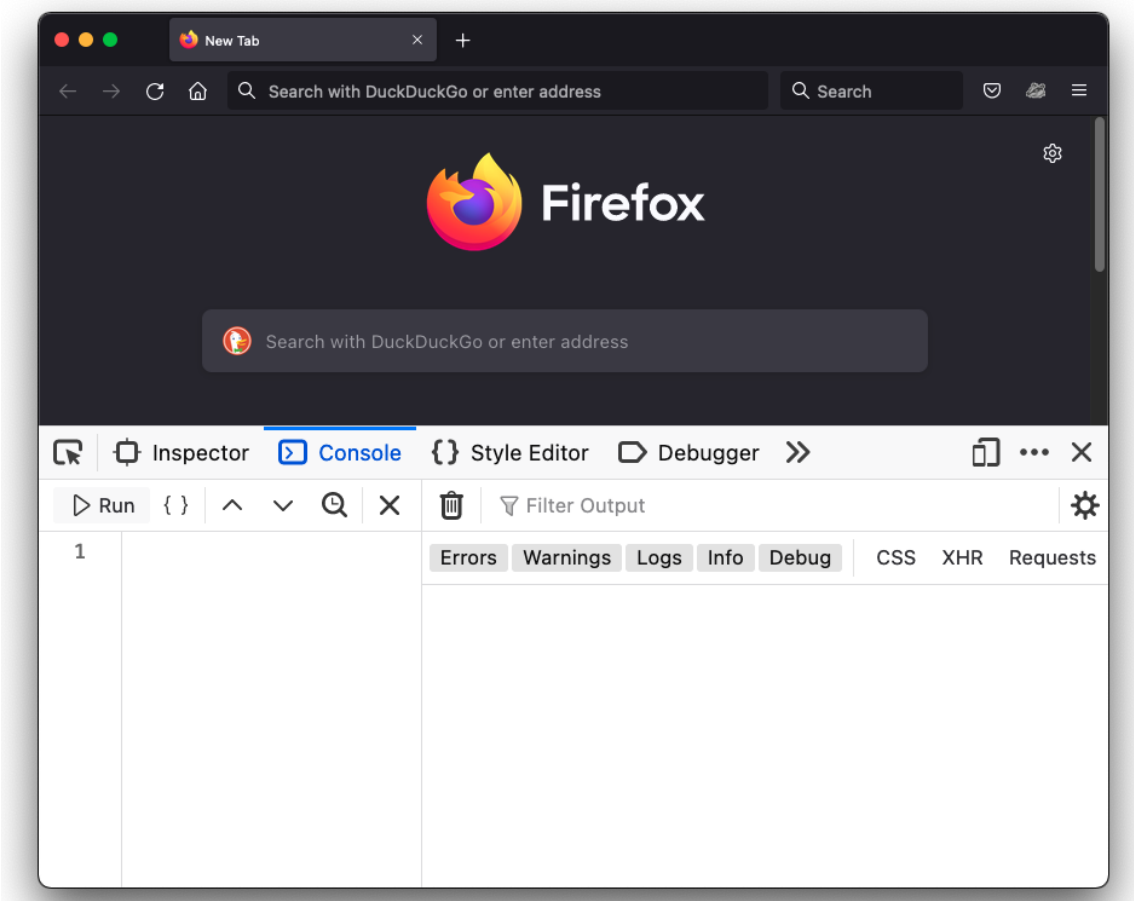

Imagen 2. Consola web
Fuente: Desafío latam

Dentro de esta consola escribe en la columna izquierda la siguiente línea:

```
alert("Esta prueba es desde la consola");
```

Ahora haz click en la opción run y verás lo siguiente:

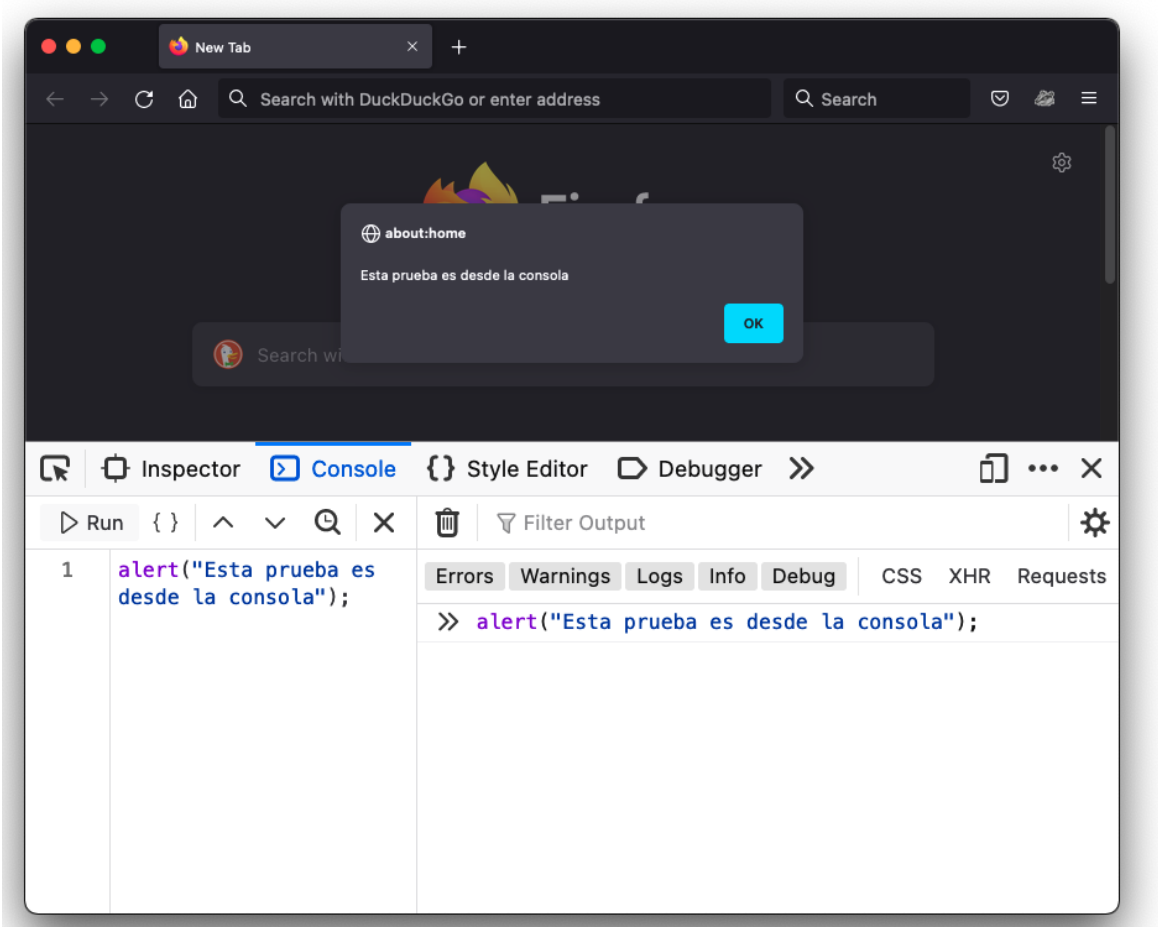

Imagen 3. Primer encuentro con JavaScript
Fuente: Desafío latam

En la misma consola podemos probar:

```
console.log("Esta prueba es desde la consola");
```

Al ejecutar el código veremos lo siguiente en la columna derecha, esto solo lo veremos si el botón de Logs está activado, en caso contrario no veremos el resultado del log.

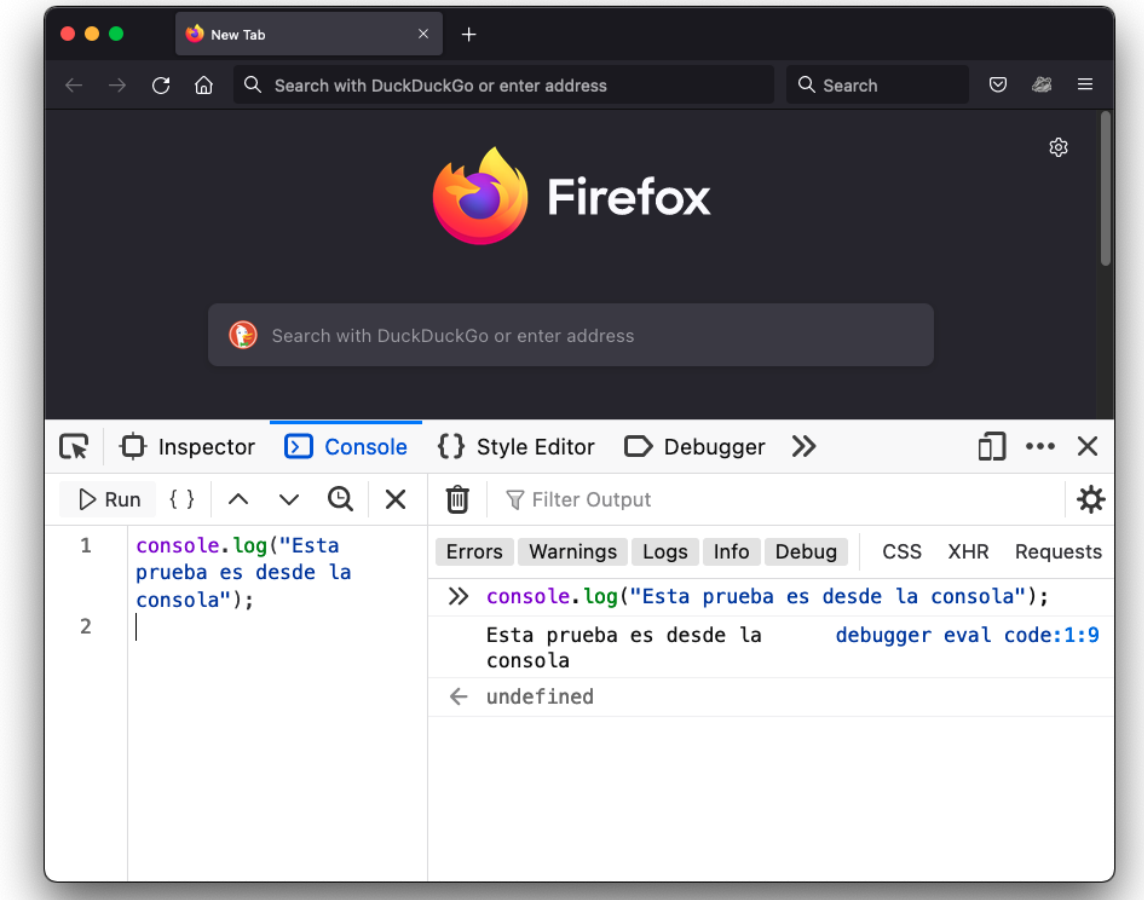

Imagen 4.Console.log
Fuente: Desafío latam

Esto es muy útil para hacer pruebas, lo ocuparemos frecuentemente a lo largo del curso.

Introducción a variables

A la hora de programar necesitaremos ir guardando valores para usarlos más tarde, esto puede ser un valor que introdujo un usuario o proveniente de algún cálculo.

Para guardar los valores ocuparemos variables. Estas las podemos imaginar como una caja que tiene un nombre y dentro el valor guardado.

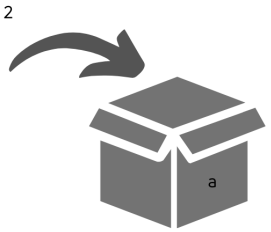

Imagen 5. Primer encuentro con JavaScript
Fuente: Desafío latam

Para definir y usar nuestra primera variable probemos escribir en la consola del inspector de elementos:

```
a = 2
alert(a + 2)
```

Al ejecutar el código (clic en run o utiliza el atajo) veremos lo siguiente:

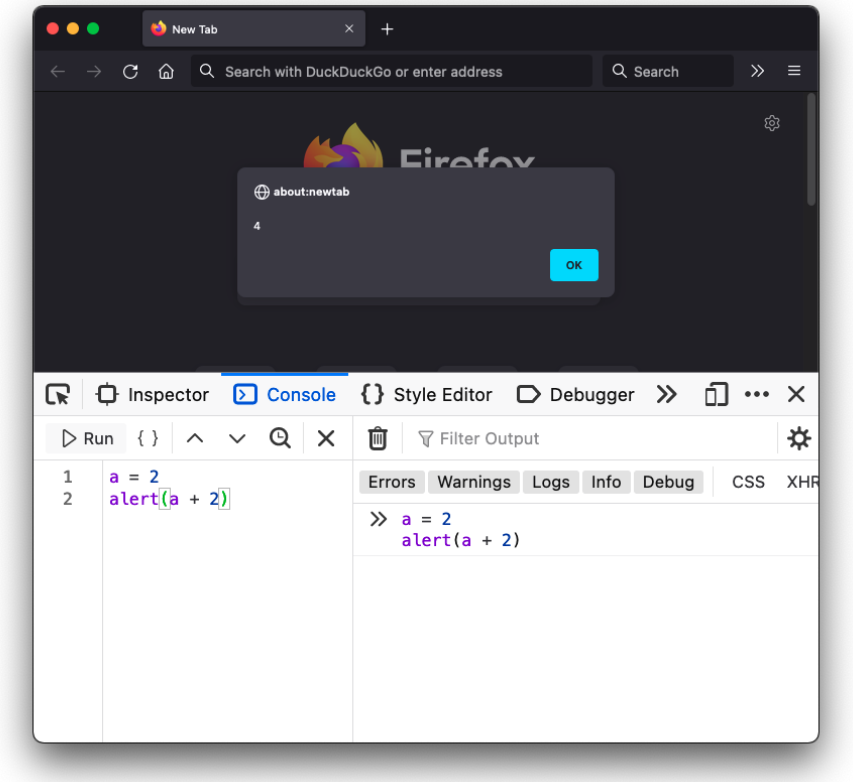

Imagen 6. Primer encuentro con JavaScript
Fuente: Desafío latam

Si la ventana izquierda de la imagen anterior no está abierta, la podemos abrir utilizando el ícono que aparece en la consola a la derecha.

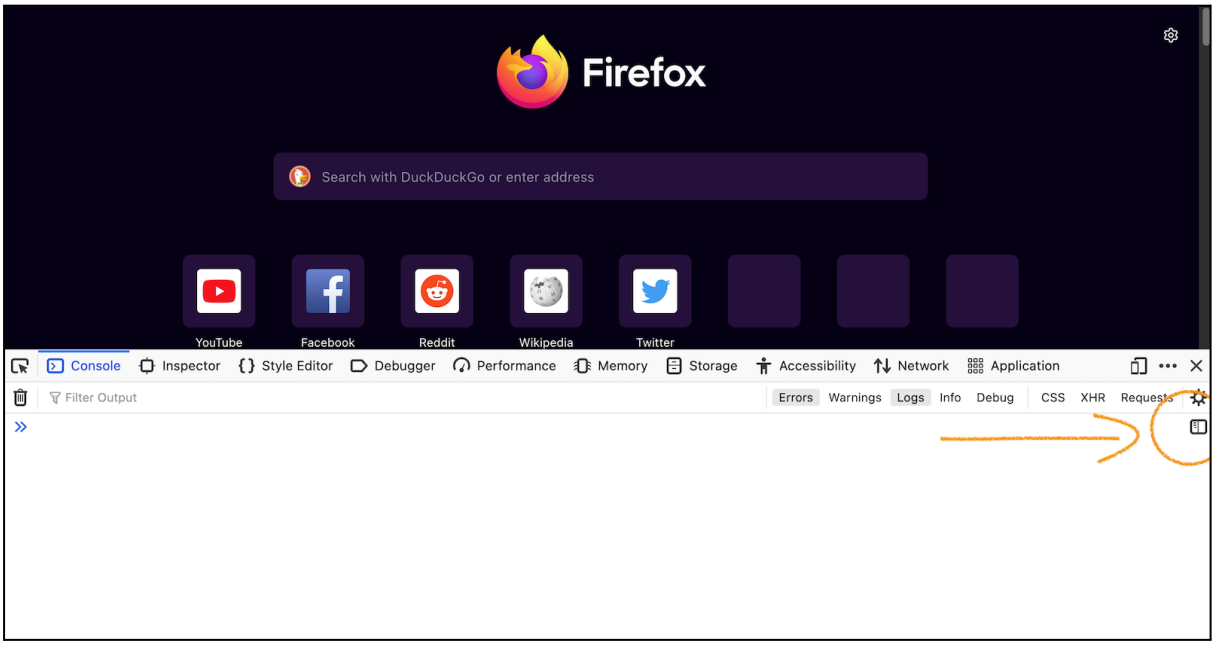

Imagen 7. Abrir el editor de código dentro de Firefox
Fuente: Desafío latam

El código anterior se lee como:

1. a = 2: Asignamos el valor 2 a la variable a
2. alert(a + 2): Mostramos en pantalla el valor dentro de la variable a sumándole 2

Por lo mismo, el resultado es 4 .

¿Por qué se llaman variables?

¿Y si son cajas que guardan valores por qué se llaman variables? El motivo es que mientras la caja conserva un nombre, el contenido dentro puede cambiar.

```
a = 2
a = 5
alert(a + 2)
```

Al ejecutarse se mostrará el valor 7 en pantalla.

Conceptos importantes en las variables

Cuando trabajamos con variables hay conceptos importantes que tenemos que tener muy en cuenta. Los dos primeros que debemos revisar son:

1. Asignar: Asignamos un valor cuando utilizamos el operador = y le damos un valor a la variable. A esto también le llamamos guardar el valor en la variable.

2. Usar: Usamos una variable cuando mostramos su valor, o la ocupamos para operar junto a otro valor, por ejemplo al sumar a + 7

Comentarios en JavaScript

Al igual que en otros lenguajes, podemos agregar comentarios en JS. Los comentarios sirven para dejar anotaciones, una instrucción comentada (envuelta en /* */) será ignorada.

```
a = 2 /* Esto sería un comentario */
/* a = 8 Esto se ignora porque está comentado */
alert(a + 2)
```

Se mostrará en pantalla el valor 4.

Creando un script

Crear un script es una forma de decir que vamos a agregar código JavaScript en un archivo en lugar de hacerlo en la consola. Para lograrlo utilizaremos la etiqueta <script> dentro del documento HTML.

En el mundo de la programación, crear un script es crear un programa que será llevado a cabo por un intérprete en lugar de un procesador. Javascript es un lenguaje interpretado donde el intérprete es el navegador, por lo mismo usualmente se dice crear un script en lugar de decir crear un programa.

¿Dónde agregar la etiqueta script?

Se puede agregar dentro del head, o en el body como cualquier etiqueta de HTML. Por ahora lo agregaremos al final justo antes de cerrar el body.

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Document</title>
  </head>
  <body>
    <p>
      Lorem ipsum dolor sit amet consectetur adipisicing elit. Architecto
      magnam, repudiandae sint optio, dolorem assumenda dignissimos a amet
      doloribus eaque voluptatem blanditiis laboriosam quis mollitia? Beatae
      eius enim minima id?
    </p>
    <script> alert("hola") </script>
  </body>
</html>

```

Imagen 8. Página web con Javascript
Fuente: Desafío latam

Al recargar la página, deberíamos ver la alerta con el texto "hola". Ahora probemos reemplazando el script anterior con el siguiente script:

```
<script>
numero1 = 100
numero2 = 50
alert(numero1 + numero2)
</script>
```

Al recargar la página y deberíamos ver la alerta mostrando 150

Cargando el script desde otro archivo

Es posible escribir el código JavaScript en un archivo independiente, para esto tenemos que seguir los siguientes pasos.

1. Crear un archivo con extensión .js Por ahora lo haremos en la misma carpeta donde se encuentra el archivo index.html
2. Utilizar la etiqueta script dentro del archivo html indicando el nombre del archivo.

Realizando el proyecto desde cero.

1. Creamos la carpeta suma
2. Creamos el archivo index.html
3. Creamos el archivo suma.js

Dentro del archivo suma.js agregamos el siguiente código:

```
/* suma.js */
numero1 = 100
numero2 = 50
alert(numero1 + numero2)
```

Dentro del archivo index.html agregamos el siguiente código:

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Script Externo</title>
</head>
<body>
  <script src="suma.js"></script>
</body>
</html>
```

Al recargar la página deberíamos ver la alerta con el número 150.

Actividad 1: Creando un Script

Realiza los siguientes pasos utilizando como base el ejercicio anterior:

1. Crea el script resta.js donde se asignen números a dos variables y se muestre la resta de ellas. Luego modifica el HTML para que el script haga referencia al archivo resta.js y pruébalo recargando la página web en el navegador. Para restar utiliza el operador -

2. Esta vez partiendo desde cero, intenta hacerlo de memoria, crea una nueva página web con la base de HTML y crea el script multiplicacion.js donde se asignen números a dos variables y se muestre la multiplicación de ellas. Carga el script al final del index.html antes del cierre del body. Para multiplicar utiliza el operador *

3. Otra vez partiendo desde cero, crea una nueva página web con la base de HTML y crea el script division.js donde se asignen números a dos variables y se muestre la división de ellas. Carga el script al final del index.html antes del cierre del body. Para dividir utiliza el operador /

Ahora que ya hemos creado un script, la siguiente definición nos hará mucho sentido. Un programa o script es una secuencia de "instrucciones" para ser ejecutadas por un computador.

Aprendizajes a verificar

- Ejecutar código JavaScript en la consola del navegador, por ejemplo revisar el valor de una variable con console.log()
- Diferenciar los conceptos de definir una variable o asignarle un nuevo valor.
    - ¿Qué tienes que escribir si te piden definir una variable a con el valor 8?
    - ¿Qué tienes que escribir si te piden asignar un nuevo valor a la variable a con el valor 15?
- Crear un script y cargarlo en el HTML
- Ejecutar operaciones matemáticas en JavaScript

Guardando texto en variables

En JavaScript además de guardar números en variables podemos guardar textos, para lograrlo es importante envolver el texto en comillas. Por ahora probemos en la consola o en un script nuevo lo siguiente.

```
contenido = "hola"
alert(contenido)
```

Estos textos en JavaScript reciben formalmente el nombre de **strings** (o cadenas de caracteres), de ahora en adelante nos referiremos a ellos como strings.

Variables versus strings

Las comillas son necesarias para evitar confundir un string con una variable:

```
contenido = "hola"
alert(contenido)
alert("contenido")
```

Si ejecutamos este código veremos que el primer alert muestra el texto "hola" mientras que el segundo muestra el texto "contenido" esto nos indica la diferencia entre la variable contenido y "contenido".

Concatenando strings

Los strings funcionan distintos a los números, cuando intentamos sumar dos strings obtendremos como resultado la concatenación de estos.

```
var1 = "Hola"
var2 = " Camila"
alert(var1 + var2)
```

Si ejecutamos este código veremos el texto "Hola Camila"

Funciones y Métodos

Las funciones y métodos son herramientas que podemos utilizar para realizar diversos tipos de acciones. Por ahora las trataremos como sinónimos, aunque más adelante aprenderemos que son cosas distintas.

JavaScript dispone de varios métodos que podemos ejecutar según necesitemos. Entre los muchos mencionados tenemos:

- console.log()
- alert()
- prompt()

Sabemos que son funciones o métodos porque vienen acompañados de paréntesis. Los elementos que agreguemos dentro de los paréntesis reciben el nombre de argumentos.

alert()

al ser ejecutado el método alert hace aparecer una ventana emergente, el texto mostrado será el argumento que le entreguemos.

```
alert("Todos pueden")
```

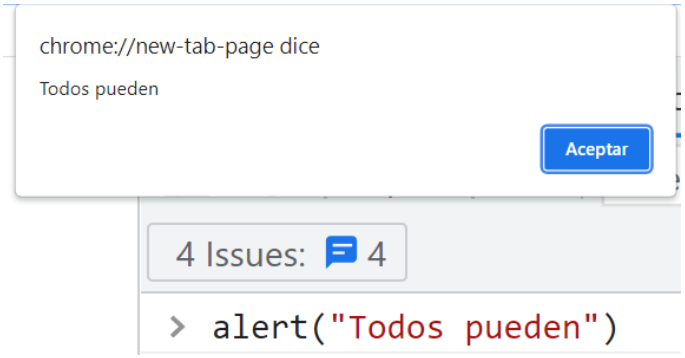

Imagen 8. alert()
Fuente: Desafío latam

¿Qué es un argumento?

Cuando se llama a una función o método, se pueden pasar uno o varios valores entre paréntesis separados por comas. Estos valores que pasamos reciben el nombre de argumentos.

console.log()

Este método muestra en la consola lo que le entreguemos como argumento.

```
console.log("Desafío Latam")
```

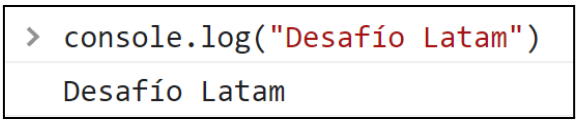

Imagen 9. console.log()
Fuente: Desafío latam

prompt()

Con el método prompt el usuario puede guardar un valor en una variable.

```
nombre = prompt("Ingrese su nombre")
```

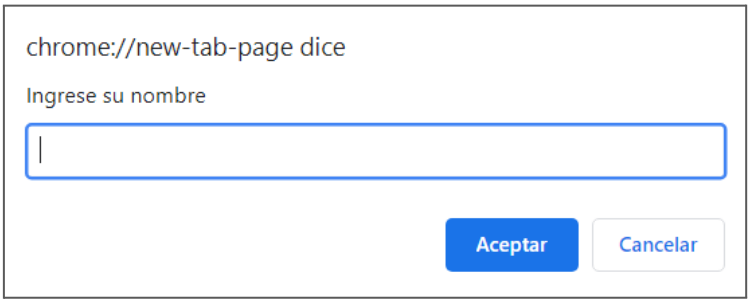

Imagen 10. prompt()
Fuente: Desafío latam

Luego podemos utilizar el texto ingresado de diversas formas.

```
nombre = prompt("Ingrese su nombre")
alert("Hola " + nombre)
```

Conceptos claves sobre métodos y funciones

Podemos pensar en los métodos y funciones como cajas negras (no sabemos exactamente lo que pasa adentro). Estas cajas pueden recibir valores de entrada y algunas de ellas entregan un valor de salida

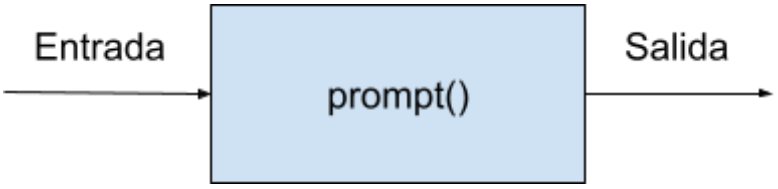

Imagen 11. Entrada y salida, prompt()
Fuente: Desafío latam

Cuando un método tiene un valor de salida, hablaremos de que **devuelve un valor**. Por ejemplo, prompt devuelve un string con el valor que introdujo el usuario, mientras que la entrada de prompt permite verificar el texto que aparece en la ventana.

Aprendizajes a verificar

Revisa particularmente que puedas transformar a código cada una de las siguientes acciones.

- Guardar en una variable el string "hola".
- Concatenar dos strings.
- Guardar en una variable un valor devuelto por el método prompt y utilizarlo dentro de un alert.

El método Number()

En algunas situaciones vamos a necesitar convertir un texto en un número, el siguiente caso ilustra perfectamente la situación.

```
num1 = prompt("Ingrese el primer número")
num2 = prompt("Ingrese el segundo número")
resultado = num1 + num2
alert(resultado)
```

Si el usuario ingresa en los prompts el número 1 y luego 2 tendrá como resultado 12 y no 3, esto se debe a que los textos se concatenan. El método prompt devuelve siempre un string y si queremos trabajar los datos como números primero debemos transformarlos, para esto existe el método number.

El método number recibe como argumento un string y devuelve un número.

```
num1 = prompt("Ingrese el primer número")
num2 = prompt("Ingrese el segundo número")
resultado = Number(num1) + Number(num2)
alert(resultado)
```

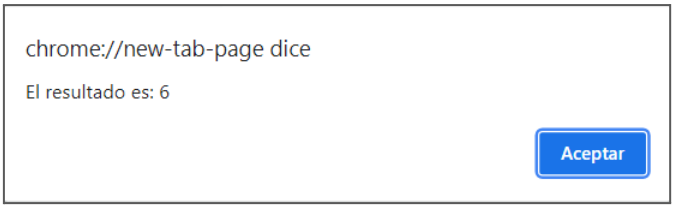

Imagen 12. El método Number()
Fuente: Desafío latam

¡Ahí lo tenemos! Ahora sí obtenemos el resultado esperado gracias a la conversión de las cadenas de texto a tipo *number*.

Sumando string con número

Al sumar un string con un número obtendremos un string. Prueba en la consola escribiendo 1 + '1' y obtendrás como resultado 11. JavaScript es un lenguaje que puede transformar a la fuerza los tipos de datos para realizar algunas operaciones. Esta característica recibe el nombre formal de **coerción**, ya que el dato se transforma de forma implícita.

Podemos evitar muchos problemas transformando los datos al tipo correcto antes de operar con ellos, si vamos a sumar utilicemos Number.

Tip: Anota la palabra **coerción** y su significado, ya que aparecerá en las preguntas finales!!

Manipulando el DOM

El DOM, por sus siglas Document Object Model, se refiere a un objeto con el que podemos seleccionar y manipular los elementos de una página web. Podemos cambiar dinámicamente el contenido, estilos e incluso el valor de sus atributos.

Introducción a QuerySelector

Hay varios métodos para seleccionar elementos del DOM, en esta guía trabajaremos con .querySelector. Para probarlo crearemos una página nueva donde utilizaremos querySelector para tomar un elemento de la página web y cambiar su contenido vía JavaScript.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <h1></h1>
  <script>
```

```
    titulo = document.querySelector("h1")
    titulo.innerHTML = "Este contenido fue agregado con JavaScript"
  </script>
</body>
</html>
```

Al abrir la página con el navegador deberíamos poder ver lo siguiente:

Imagen 13. El método *querySelector*
Fuente: Desafío latam

Puede sorprenderte que en la página web no se alcanza a ver el *h1* vacío, esto es porque la ejecución del script es prácticamente inmediata.

Actividad 2: Agregar nuevo contenido

1. Crea una página web básica con un párrafo con contenido Lorem Ipsum. Es importante que sea un párrafo y no otro tipo de etiqueta.
2. Al final de la página, antes del cierre del body agrega la etiqueta <script>
3. Dentro del script agrega:

    a. `parrafo = document.querySelector("p")`
    b. `parrafo.innerHTML = "Este contenido fue agregado con JavaScript"`

4. Al abrir la página web deberías ver el nuevo contenido.

Buscando por clase y por id

El argumento que utilizamos en el método querySelector no es más que un selector de elementos tal y como lo hacemos en CSS, es decir, que podemos apuntar a un elemento por clase, id o incluso especificidad.

Por ejemplo, dado el siguiente código HTML:

```
<nav class="menu">Menú</nav>
<section id="Hero-Section">
```

```
    ...
</section>
<footer>Pie de página</footer>
```

Podemos crear variables con la referencia de estos elementos ocupando el *querySelector*.

```
menu = document.querySelector(".nav")
heroSection = document.querySelector("#Hero-Section")
pieDePagina = document.querySelector("footer")
```

Ahora que disponemos de estas variables en nuestro código, podríamos interactuar con los elementos modificando su contenido, sus estilos o atributos.

Cambiando contenido

Utilizando querySelector y .innerHtml podemos seleccionar elementos que estén anidados o dentro de un contenedor, por ejemplo si tenemos el siguiente párrafo:

```
<p>Hola, mi nombre es <span class="nombre">____</span>, mucho gusto</p>
```

Podemos ocupar el *querySelector* para hacer una referencia del span que está dentro del párrafo y luego cambiarle el contenido:

```
var nombreSpan = document.querySelector("p .nombre")
nombreSpan.innerHTML = "John"
```

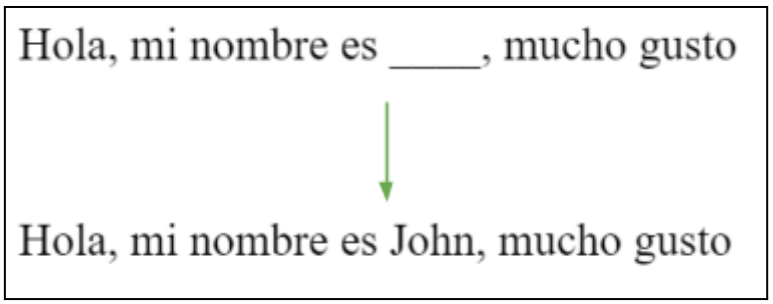

Imagen 14. Cambiando el contenido de un span
Fuente: Desafío latam

Actividad 3: Rellenando un contrato de arriendo

Realicemos un ejercicio en donde podamos completar el contenido de un párrafo.

1. En un archivo base HTML:
   a. Utiliza la etiqueta *p* escribe la porción del contrato de arriendo usando etiquetas *span* en los espacios por llenar.
   b. Agrega identificadores a las etiquetas *span* con un nombre que haga referencia a su contenido.
   c. Agrega la etiqueta script al final del documento html (antes de </html>).

2. Dentro la etiqueta script previamente agregada:
   a. Utiliza prompt y guarda la información ingresada por el usuario en las variables info1 e info2.
   b. Utiliza el método querySelector para encontrar los elementos que vas a modificar y guarda esa información en las variables ele1 y ele2
   c. Utiliza ele1.*innerHTML(info1)* para cambiar el valor

3. Observa el resultado en el navegador

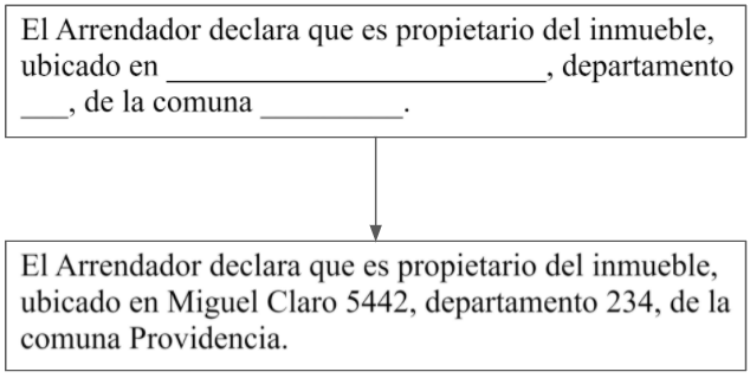

Imagen 15. Resultado de Rellenando un contrato de arriendo
Fuente: Desafío latam

Modificando estilos con JavaScript

Previamente, aprendimos que podemos modificar el contenido de una etiqueta HTML utilizando JavaScript, a continuación aprenderemos a cambiar los estilos.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <h1> Desafío Latam </h1>
  <script>
    titulo = document.querySelector("h1")
    titulo.style.color = "#7ba238"
  </script>
</body>
</html>
```

Al abrir la página en el navegador veremos lo siguiente:

Imagen 16. Modificando el color de texto de un título
Fuente: Desafío latam

utilizando .style podemos cambiar todas las propiedades de CSS. Pero es importante considerar que existe una pequeña diferencia entre cómo llamamos las propiedades de CSS en JavaScript a cómo lo escribimos directamente en un documento CSS.

En CSS la propiedad border-radius la escribimos separando las palabras con un guion medio.

```
border-radius: 50%
```

No obstante, en JavaScript cuando queramos ocupar una propiedad de CSS que utilice dos o más palabras, deberemos recurrir al Camelcase, es decir, que tendremos que a partir de la segunda palabra, todas las restantes deberán empezar por una letra mayúscula.

A continuación algunos ejemplos de cómo se definirían propiedades CSS desde JavaScript con Camel Case:

```
imagen.style.borderRadius = "50%"
```

```
texto.style.fontFamily = "Fantasy"
```

```
texto.style.fontSize= "10px"
```

Actividad 4: Redondeando una imagen

Realicemos un ejercicio en donde redondeemos una imagen con CSS aplicando un *border-radius* definido por el usuario.

1. En un archivo HTML nuevo, crear una base con una etiqueta *img* que ocupe una imagen con dimensiones cuadradas, puedes utilizar la siguiente: https://placedog.net/500/500

2. Agregar la etiqueta script al final de la página, antes del cierre del body, dentro de la etiqueta script.

    a. Utilizar el método querySelector para seleccionar la etiqueta *img*.

    b. Utiliza el *prompt* para solicitar al usuario el valor de la propiedad *border-radius*.

    c. A través del atributo *style* accede a la propiedad *borderRadius* y asigna el porcentaje ingresado por el usuario. Para hacerlo sencillo ingrésalo directamente con la unidad de medida, o sea, el usuario ingresará 50%.

3. Observa el resultado en el navegador.

Imagen 17. Modificando estilos con JavaScript
Fuente: Desafío latam

Manipulando el DOM en respuesta a un evento

En HTML podemos ejecutar JavaScript directamente en función de ciertos eventos, por ejemplo que el usuario haga clic en algún elemento.

Cambiando en respuesta a un clic

Para lograrlo vamos a crear una página web con un botón, que al hacer clic en el botón se ejecute código JavaScript, en este caso el código seleccionará el elemento body y cambiará el color de fondo a negro.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <button onclick="
    body = document.querySelector('body')
    body.style.backgroundColor='black'
  "> A negro </button>
</body>
</html>
```

Algunas observaciones importantes del código mostrado:

1. El código JavaScript se escribe directamente sin utilizar la etiqueta script.
2. Todo el código se escribe dentro de comillas dobles, por lo mismo no podemos utilizarlas dentro, utilizaremos comillas simples en su lugar.

Vamos a hacer un ejercicio similar, ahora tendremos 2 botones y una imagen, el primero agrega bordes redondeados a la imagen, el segundo le quitará los bordes redondeados.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <img id="imagen-a-trabajar" src="https://placedog.net/500
  " alt="">

  <button onclick="
    element = document.querySelector('#imagen-a-trabajar')
    element.style.borderRadius='30%'
  "> Agregar Bordes </button>

  <button onclick="
  element = document.querySelector('#imagen-a-trabajar')
  element.style.borderRadius='0'
  "> Quitar Bordes </button>
</body>
</html>
```

Veamos otro ejemplo similar: en esta ocasión tendremos una lista de elementos, al hacer click en el elemento aparecerá tachado. La propiedad en css para esto es text-decoration con el valor "line-through" pero lo haremos utilizando JS.

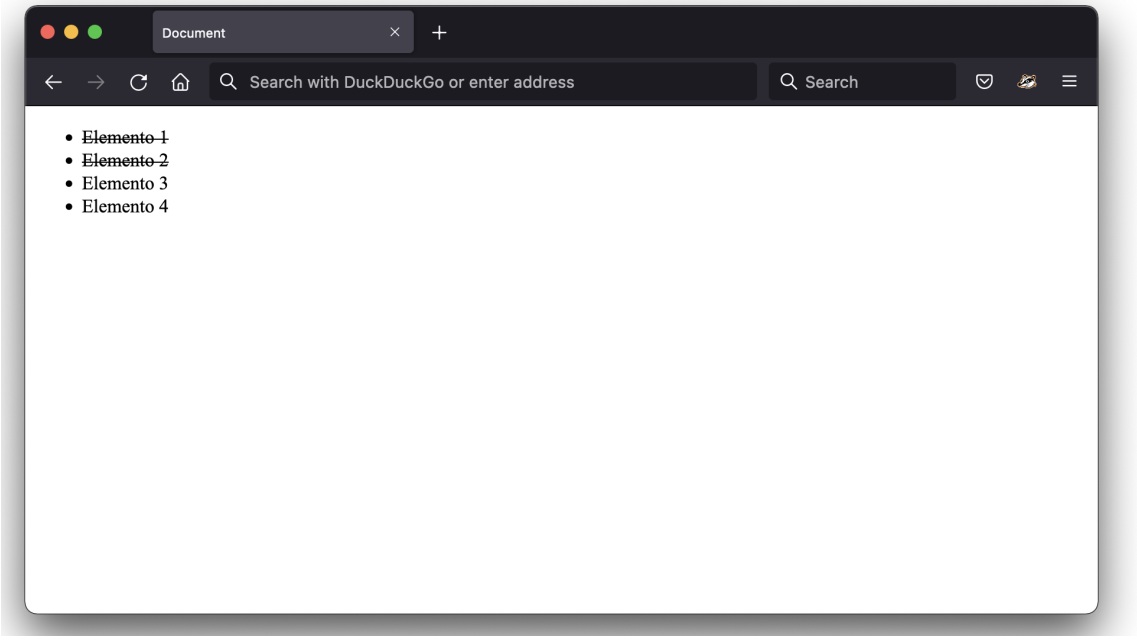

Imagen 18. Elemento tachado.
Fuente: Desafío latam

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <ul>
    <li id="ele1" onclick="
      ele = document.querySelector('#ele1')
      ele.style.textDecoration = 'line-through';
    "> Elemento 1</li>
    <li id="ele2" onclick="
    ele = document.querySelector('#ele2')
    ele.style.textDecoration = 'line-through';
  "> Elemento 2</li>
    <li> Elemento 3</li>
    <li> Elemento 4</li>
  </ul>
</body>
</html>
```

Actividad 5: Práctica

1. Agrega a la página anterior el evento click al elemento 3 y al elemento 4.

Cambiando en respuesta al cambio de un input

También podemos ejecutar JavaScript cuando un elemento de un formulario cambia. Un caso potente de uso de esto son los selectores. Con ellos podríamos cambiar el color de fondo de la página, el color de un producto o mostrar una foto distinta dependiendo de la opción que se seleccione.

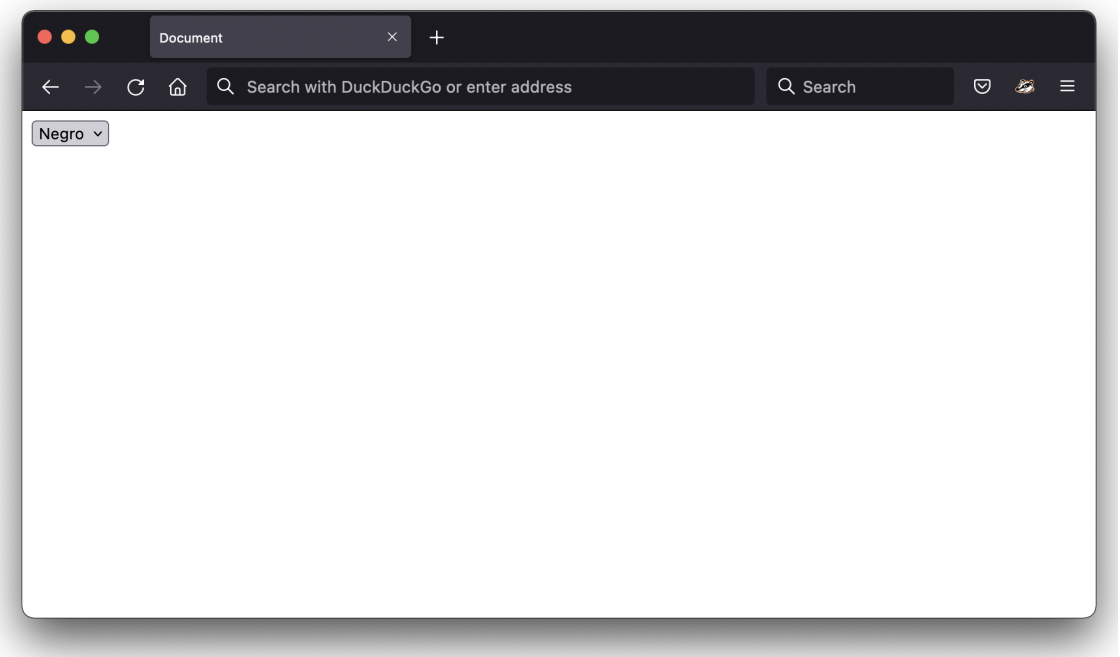

Imagen 19. Selector.
Fuente: Desafío latam

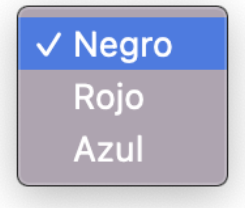

Imagen 20. Selector.
Fuente: Desafío latam

Nuestro primer ejemplo será un selector que muestra 3 colores. Al momento de escoger uno se mostrará una alerta con el valor seleccionado.

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta http-equiv="X-UA-Compatible" content="IE=edge">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>Document</title>
</head>
<body>

 <select id="selector" onchange="
   element = document.querySelector('#selector')
   alert(element.value)
 ">
   <option value="black">Negro</option>
   <option value="red">Rojo</option>
   <option value="blue">Azul</option>
 </select>

</body>
</html>
```

Un selector es solo una etiqueta html que se compone de *<select>* y por cada opción hay que agregar un *<option>* Cada opción *<option>* tiene un texto que se muestra y un valor (value), el valor es lo que maneja internamente indicando que realmente seleccionó el usuario.

En términos de JavaScript lo único nuevo consiste en utilizar onchange en lugar de onclick y que al momento de hacer el alert hacemos element.value puesto que ahí está guardado el valor que necesitamos.

Ahora utilicemos lo aprendido anteriormente para cambiar el color de la página al momento de escoger una opción.

```
  <select id="selector" onchange="
    element = document.querySelector('#selector')
    body = document.querySelector('body')
    body.style.backgroundColor = element.value
  ">
    <option value="black">Negro</option>
    <option value="red">Rojo</option>
    <option value="blue">Azul</option>
  </select>
```

Lo único nuevo en este código es que estamos seleccionando dos elementos distintos, (el selector y el body) y estamos ocupando el valor de la opción seleccionada para cambiar el color de fondo del body.

Un ejercicio similar podría ser mostrar en un texto la opción seleccionada. Para mostrar un texto dependiendo de la opción que mostremos, imaginemos que estamos haciendo un quiz y hay 4 alternativas.
Para lograrlo necesitamos un select con las alternativas y un elemento donde mostraremos el resultado:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <p> ¿Qué lenguaje de programación estamos aprendiendo? </p>
  <select id="selector" onchange="
    elemento_seleccionado = document.querySelector('#selector')
    elemento_resultado = document.querySelector('#resultado')
    elemento_resultado.innerHTML = 'La alternativa es ' +
elemento_seleccionado.value
  ">
    <option value="incorrecta">Java</option>
    <option value="correcta">Javascript</option>
    <option value="incorrecta">Python</option>
    <option value="incorrecta">Ruby</option>
  </select>
  <span id="resultado"> </span>

</body>
</html>
```

Esto sirve para dinámicas entretenidas en una página web, pero si el resultado es sensible a trampas, por ejemplo, es de un concurso o para tomar pruebas, necesitaremos que la respuesta no está en el lado de front de forma que un usuario avanzado no pueda obtenerla simplemente con el inspector de elementos.

Lógica para resolver problemas

Los ejercicios de modificación del DOM en función de un evento tienen una estrategia general para resolverse.

1. Agregamos todos los elementos necesarios en el HTML
2. Identificar cuál es el elemento es el disparador
    a. un `select`?
    b. un `input`?
    c. Otro
3. Identificar cuál es el evento
    a. ¿`onchange`?
    b. ¿`onclick`?
4. ¿Necesitamos algún valor de algún elemento de la página web?
    a. Seleccionamos el elemento y lo guardamos en una variable
    b. ¿Es algo directo en el html?
        i. Lo podemos obtener con `nombreVariable.innerhtml`
    c. ¿Es un valor asociado a una propiedad del css?
        i. Lo podemos obtener con `nombreVariable.style.propiedadCSS`
    d. ¿Es un valor en un input del formulario?
        i. Lo podemos obtener con nombreVariable.value
    e. ¿Qué debemos modificar del DOM?
        i. Seleccionamos el elemento que queremos modificar utilizando querySelector y lo guardamos en una variable
5. Modificamos un elemento
    a. ¿Qué elemento tenemos que modificar? Lo seleccionamos con `document.querySelector` y guardamos en una variable
    b. ¿Es html? Utilizamos `nombreVariable.innerHTML`
    c. ¿Es CSS? Utilizamos `nombreVariable.style.propiedadCSS`
    d. ¿Es un input de un formulario? Utilizamos `nombreVariable.value`

Actividad 6: Redondeando una imagen

Pongamos a prueba nuestra lógica para resolver problemas. Nos piden una página web con un input y un botón, el usuario escribirá un color sobre el input, al presionar el botón se debe cambiar el color de fondo de la página. El color será introducido en inglés (ejemplo: `red`, `blue`, etc.) o en hexadecimal (`#ff0000`).

Reflexiona: Antes de ver el paso a paso, responde las preguntas de la lógica para resolver problemas con este ejercicio.

1. ¿Agregamos los elementos necesarios en el html?

En este caso sería la base de html, un input y botón

2. ¿Cuál es el disparador?

Aquí el disparador es el botón que al hacerle click realiza una acción, en este caso cambiar el color de fondo de la página.

3. ¿Cuál es el evento?

El clic sobre el botón

4. ¿Necesitamos algún valor de la página web?

Sí. Necesitamos el color escrito por el usuario. Lo debemos obtener del input, por lo que tenemos que seleccionar el elemento con document.querySelector() y luego con nombreVariable.value

5. Modificamos un elemento

Nos piden cambiar el color de la página web, por lo tanto, tenemos que cambiar el estilo de body.

Luego pasamos lo anterior a código

Intenta escribir el código antes de ver la solución

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta http-equiv="X-UA-Compatible" content="IE=edge">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>Document</title>
</head>
<body>
 <input id="input-1" type="text">
 <button id="btn-1" onclick="
   input = document.querySelector('#input-1')
   body = document.querySelector('body')
   body.style.backgroundColor = input.value
 ">
 Cambiar </button>
 </body>
</html>
```

Actividad 7: Otro ejemplo

Probemos con otro ejemplo, crearemos una página con un contador de clicks, para eso tendremos un párrafo con el número cero, luego crearemos un botón que aumentará la cuenta en un uno cada vez que es presionado.

Reflexiona: Antes de ver el paso a paso, responde las preguntas de la lógica para resolver problemas con este ejercicio.

1. ¿Agregamos los elementos necesarios en el html?

En este caso sería la base de html, un párrafo con el texto cero y un botón

2. ¿Cuál es el disparador?

Aquí el disparador es el botón que al hacerle click realiza una acción, en este caso incrementar en uno el número en el párrafo.

3. ¿Cuál es el evento?

El clic sobre el botón

4. ¿Necesitamos algún valor de la página web?

Si. Necesitamos el número escrito en el párrafo, para obtenerlo ocuparemos `.innerHTML` recordemos también que el contenido dentro del párrafo es un string y no un número, así que necesitaremos convertir en número utilizando `Number()`

5. Modificamos un elemento

Incrementamos el valor en 1 y lo agregamos al html, ocupando `elemento.innerHTML = nuevo_valor`

Luego pasamos lo anterior a código.

Intenta escribir el código antes de ver la solución.

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta http-equiv="X-UA-Compatible" content="IE=edge">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>Document</title>
</head>
<body>
 <p id="p-1"> 0 </p>
 <button onclick="
   parrafo = document.querySelector('#p-1')
   cuenta = Number(parrafo.innerHTML)
   cuenta = cuenta + 1
   parrafo.innerHTML = cuenta
 "> Incrementar </button>
</body>
</html>
```

Otras formas de obtener un elemento del DOM

Así como existe document.querySelector existen otras formas de obtener un elemento del DOM, una muy usada es `document.getElementById('id-del-elemento')` o

`document.getElementsByClassName('clase')`. A lo largo del curso utilizaremos principalmente `document.querySelector`

Preguntas de entrevista laboral

Contesta con tus palabras las siguientes preguntas, anota las respuestas y luego revisa en la guía las definiciones para evaluar si las aprendiste correctamente.

1. ¿Qué es el DOM?
2. ¿Cómo se puede agregar una clase a un elemento del DOM?
3. ¿Cómo se puede cambiar un estilo a un elemento del DOM?
4. ¿Qué se obtiene al sumar 2 + '2'?
5. ¿Qué se obtiene al sumar 2 + 3 + '2'?
6. ¿Qué es la coerción?

Resumen

- JavaScript es un lenguaje de programación que le agrega interactividad a un sitio web.
- Podemos escribir código JavaScript directamente en la consola de cualquier navegador.
- Se debe respetar la sintaxis para que nuestros programas se ejecuten correctamente.
- Existen varios tipos de datos que se definen según la forma en la que se escriben.
- Los métodos son funciones que tienen diferentes objetivos y que podemos ejecutar en diferentes situaciones según sea la necesidad.
- La concatenación consiste en unir varias cadenas de texto para formar una frase compuesta.
- El método *Number()* se utiliza para formatear una cadena de texto a un número y pueda ser usado en una operación aritmética.
- JavaScript puede manipular los elementos del DOM para acceder o modificar sus atributos.
- A través del atributo *style* podemos definir propiedades de CSS para cambiar los estilos de un elemento, considerando siempre escribirlas con Camelcase.
- A través de los atributos onclick y onchange de HTML podemos agregar JavaScript directamente en respuesta a un clic o el cambio de un input.