

Variables

Módulo 01

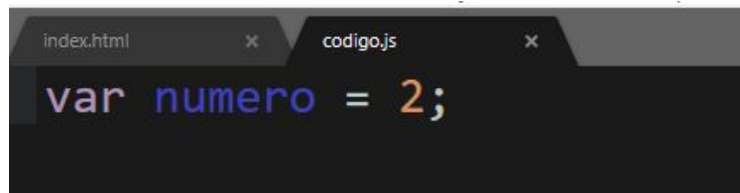
Variables: Introducción

Las variables en los lenguajes de programación en general , no solamente en JS, tienen la misma lógica que en una operación matemática.

Una variable es un elemento que se emplea para almacenar, guardar en un cajón la información y utilizarla nuevamente (por eso el nombre de variable).

La verdad es que sin variables cualquier programa es inútil y sin sentido por eso es interesante conocerlas.

Las variables se crean o definen mediante la palabra **reservada var**.

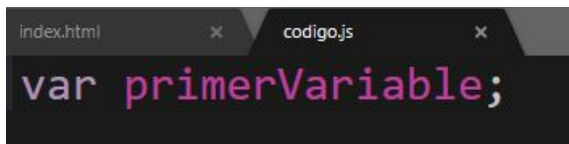
A screenshot of a code editor with two tabs: 'index.html' and 'codigo.js'. The 'codigo.js' tab is active and shows the JavaScript code 'var numero = 2;'. The word 'var' is in light blue, 'numero' is in dark blue, and '2' is in orange. The background is dark grey.

```
var numero = 2;
```

Variables: Reglas de Nomenclatura

Las **reglas para poder nombrar una variable** son las siguientes,

- Los nombres de variables puede contener **letras, números, _ (underscore) y signo de dólar (\$)**
- **No se puede comenzar con un número**
- Los nombres son **caseSensitive** (esto significa que no es lo mismo A qué a)
- Hay **palabras reservadas** que no las podés usar para nombres de variables, no te preocupes a medida que vayamos avanzado vas a saber cuales son

A screenshot of a code editor with two tabs: 'index.html' and 'codigo.js'. The 'codigo.js' tab is active, showing the code 'var primerVariable;' in a dark theme with syntax highlighting (keywords in blue, identifiers in pink, and punctuation in white).

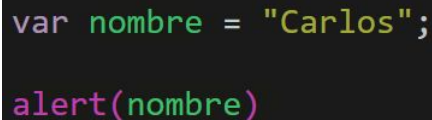
Variables: Tipos, string

String son fundamentalmente cadenas de texto, se escriben entre comillas dobles o comillas simples. Vamos a probarlo en nuestro **codigo.js** , por ejemplo,

A screenshot of a code editor with two tabs: 'index.html' and 'codigo.js'. The 'codigo.js' tab is active and shows the code `var nombre = "Carlos";` with syntax highlighting: 'var' is purple, 'nombre' is green, '=' is white, '"Carlos"' is yellow, and ';' is white.

```
var nombre = "Carlos";
```

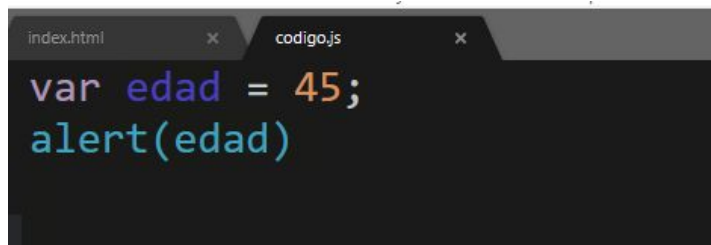
Por supuesto, que nada de esto tiene sentido si yo no puedo mostrar la información, por lo tanto, vamos a mostrar el nombre del **empleado a través de una ventana de alerta**, cuándo se muestra el dato, no es necesario como en el primer ejemplo **ponerlo entre comillas** ya que la variable es un objeto y no requiere de las mismas.

A screenshot of a code editor showing the code `var nombre = "Carlos";` followed by `alert(nombre)` on the next line. The syntax highlighting is consistent with the previous block: 'var' is purple, 'nombre' is green, '=' is white, '"Carlos"' is yellow, ';' is white, 'alert' is purple, and '(nombre)' is green.

```
var nombre = "Carlos";  
alert(nombre)
```

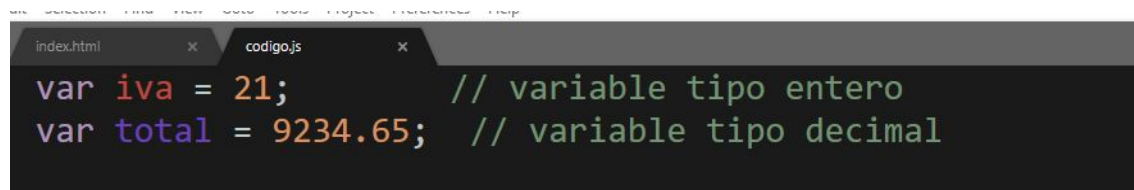
Variables: Tipos, numérica

Cualquier expresión numérica, esta puede ser float (decimal) o entera. En nuestro archivo, trabajaremos de la siguiente manera,



```
index.html x  codigo.js x
var edad = 45;
alert(edad)
```

Si queremos distinguir **decimales de enteros** lo haremos de la siguiente forma,



```
index.html x  codigo.js x
var iva = 21;           // variable tipo entero
var total = 9234.65;    // variable tipo decimal
```

Variables: Tipos, numérica


También existen otras variables que más adelante profundizaremos tales como,

- **Boolean** : Expresiones booleanas **TRUE** o **FALSE**.
- **Undefined** : Toda variable declarada sin valor o cualquier propiedad interna no existente de un objeto
- **Object** : Vector asociativo en n dimensiones. El mismo se inicializa de manera literal con {}(llaves) y contiene en su interior pares de índices asociados a valores por el operador : (dos puntos) separados por , (coma). Ej.: {nombre:"Educacion IT"}.
- **Array** : Objeto especializado en poder tener además de su comportamiento habitual la habilidad de guardar datos de manera secuencial, es decir bajo **índices numéricos** auto incrementales.

Output de Javascript : document.write()

Hay variadas formas de mostrar nuestra información en **JS**, por ejemplo, hemos visto cómo a través de una ventana de alerta podemos hacerlo.

Sin embargo existen otras maneras de interactuar con nuestro **HTML**, por ejemplo con **document.write()**, que nos permite escribir directamente en nuestro documento, para lograr tal fin, generamos las siguientes líneas de código en nuestro **archivo.js**,

A screenshot of a code editor with two tabs: 'index.html' and 'codigo.js'. The 'codigo.js' tab is active, showing the code `document.write('Hola estoy dentro de mi documento')` in a dark-themed editor with syntax highlighting.

El resultado será el siguiente,

Hola estoy dentro de mi documento

Output de Javascript : document.write()

Si queremos generar un mejor formato, podemos **implementar etiquetas de HTML**, por ejemplo, nuestra intención es que este **texto sea un título**, lo haremos de la siguiente forma,

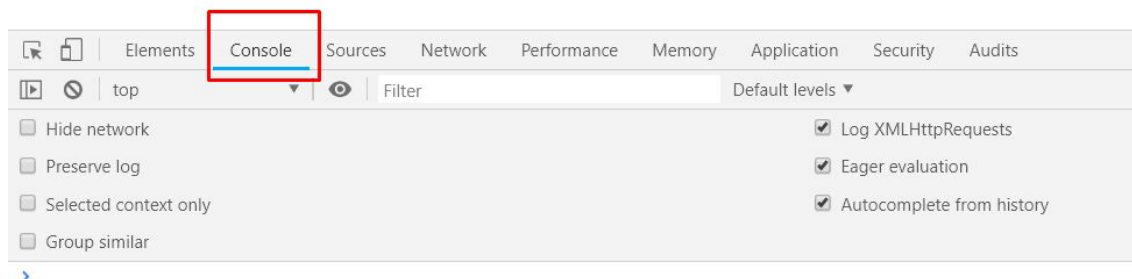
```
document.write('<h1>Hola estoy dentro de mi documento</h1>')
```

Podemos también implementar , varias líneas dentro de nuestro archivo con diferentes etiquetas válidas de HTML,

```
document.write('<h1>Hola estoy dentro de mi documento</h1>')
document.write('<p>Este es un párrafo con<strong>negrita</strong></p>')
```


Output de Javascript : console.log()

Esta forma de generar contenido se hace a través de la consola, a la cual podemos acceder presionando la **tecla f12** desde en nuestro navegador,

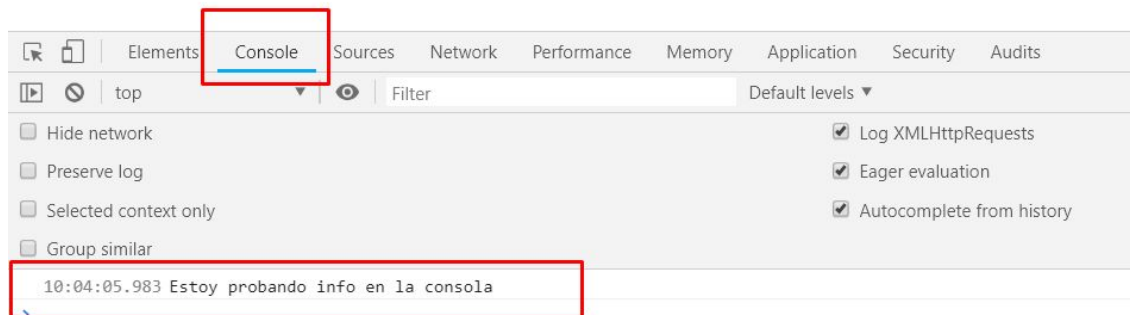


La consola, nos permite conocer errores, datos y demás cuestiones sumamente importantes para el **trabajo con JS**, pero puntualmente en este caso la utilizaremos para generar información de la siguiente manera,

```
index.html x  codigo.js x
console.log('Estoy probando info en la consola')
```

Output de Javascript : console.log()

El resultado del slide anterior será el siguiente,



Revisión

- Repase los conceptos básicos de un **lenguaje de programación**.
- Trabaje con variables en sus **diferentes tipos**
- Implemente una **ventana de alerta de forma externa a través de un string**
- Muestre datos a través de **document.write()**
- Trabaje con **console.log()**.
- Aplique todas las propiedades en el **proyecto integrador**
- Realice las preguntas necesarias al/la docente antes de continuar



¡Muchas gracias!

¡Sigamos trabajando!