**2) Formas de enlazar JavaScript.**

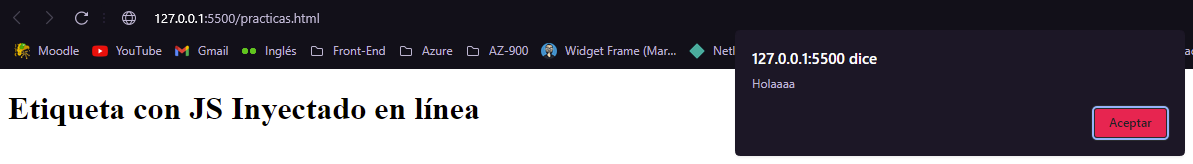
* **En línea.**

Una de las formas de trabajarlo en línea, sería a través de eventos.

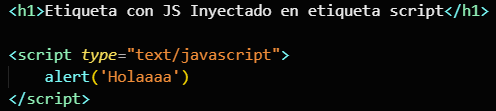
**NOTA:** No es recomendable por temas de modularización.



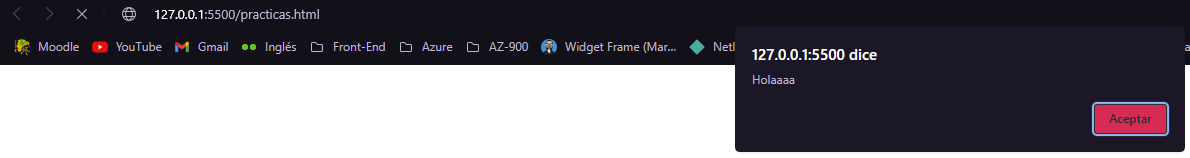
**Salida:**



* **Como contenido en la etiqueta <script>**

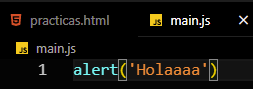
****

**Salida:**

****

* **Como contenido en un archivo de formato .js**

Esta sería la forma más recomendada, aunque también depende de la situación.



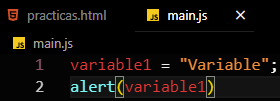


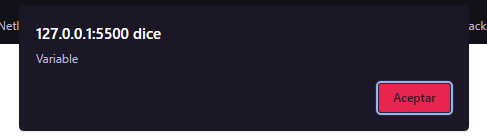
**Salida:**



**3) Variables y constantes.**

Una variable, es un espacio que nosotros reservamos en memoria.





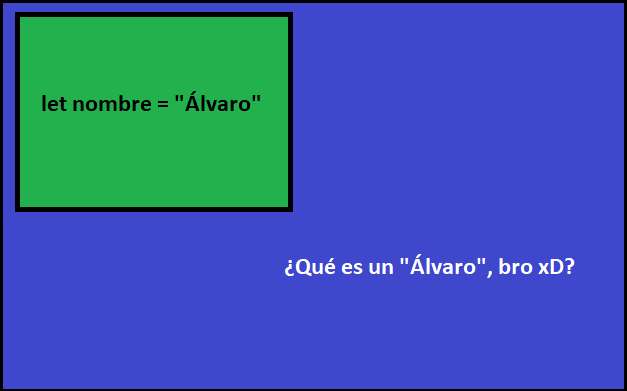
Las variables pueden ser declaradas de las siguientes 3 formas

* **var:** Var nos permite tener un alcance global de la variable en todo el código.

**Ejemplo:** var numero = 15;

* **let:** Let lo que hace es limitar el alcance de la variable a un bloque de código o lo que se conoce como **Scope.**

**Ejemplo:** let nombre = “Álvaro”;

****

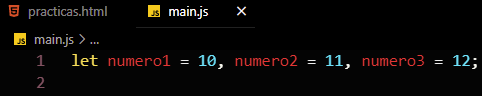
* **const:** Const se usa para constantes, es decir valores que una vez asignados NO pueden cambiar su valor durante todo el flujo del programa.

**Ejemplo:** const pi = 3.1416

**NOTA:** const se tiene que inicializar al momento de su declaración.

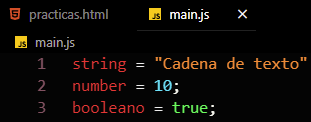
**3.1) Creación de múltiples variables.**

Para crear muchas variables de un mismo tipo lo haremos de la siguiente manera.

****

**4) Tipos de datos.**

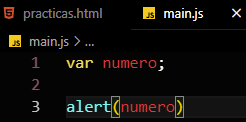
Los principales tipos de datos (primitivos) serían los siguientes.



**5) Casos especiales de datos.**

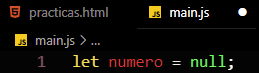
Para empezar, son 3 tipos de datos que nos indican que hay un variable no definida o hay un error.

* **Undefined:** Es un tipo de dato que nos indica que la variable existe en el programa, pero no tiene valor.

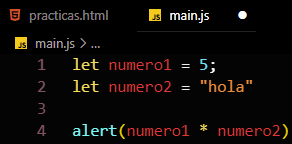
****

****

* **Null:** Esto nos indica que la variable está vacía, pero hay mucha diferencia con undefined y se declara intencionalmente.

****

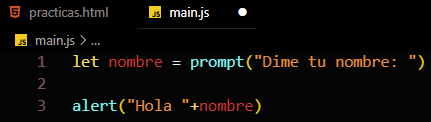
* **NaN (Not a Number):** Nos indica que estamos tratando de hacer una operación matemática con algo que no es un número.

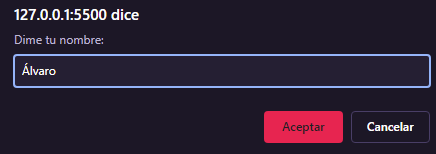
****

****

**6) Pruebas con prompt.**

Prompt, es una de las Apis de JavaScript que nos permite darle un dato a través de una entrada de texto.

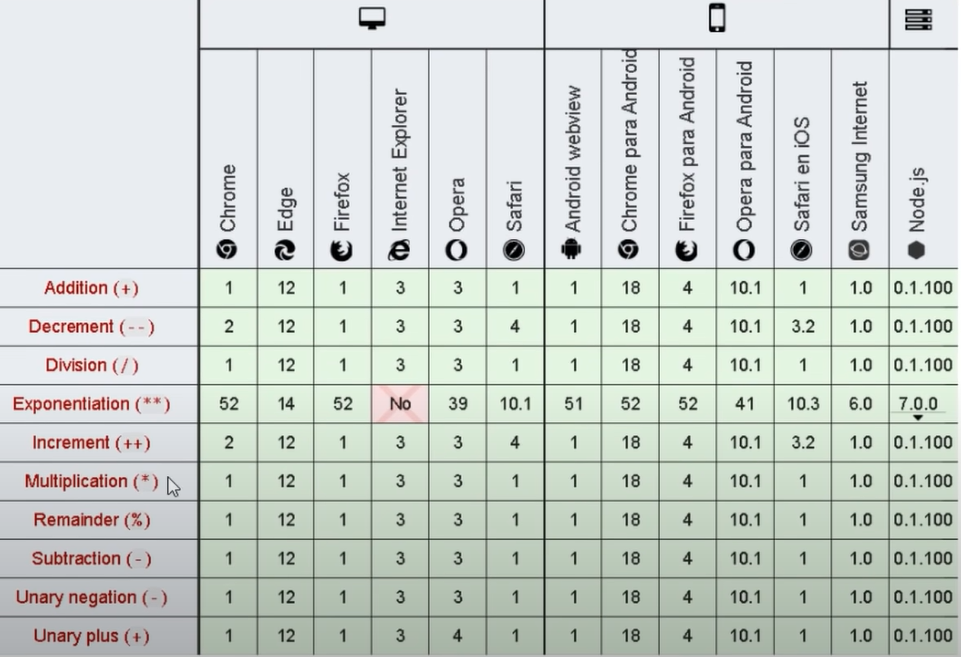
****

****

****

**7) Operadores en JavaScript.**

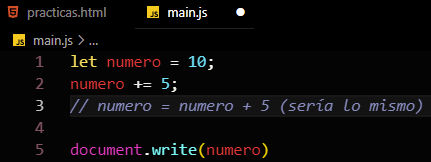
* **Operadores aritméticos:** Son los que toman valores numéricos (literales o variables) como sus operandos y retornan un valor numérico único.

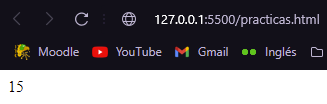
****

* **Operadores de asignación:** Asigna un valor al operando de la izquierda basado en el valor del operando de la derecha.

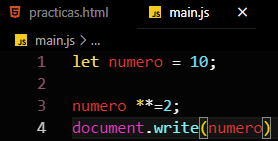


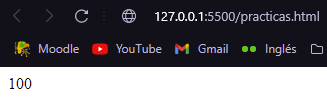
**Ejemplo.**





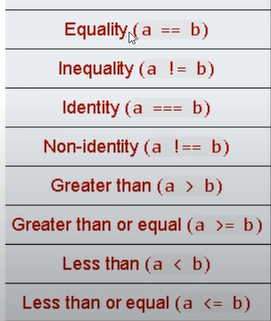
**Ejemplo exponenciales.**

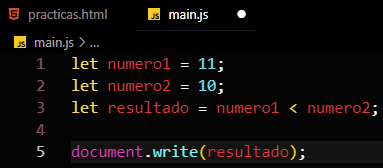




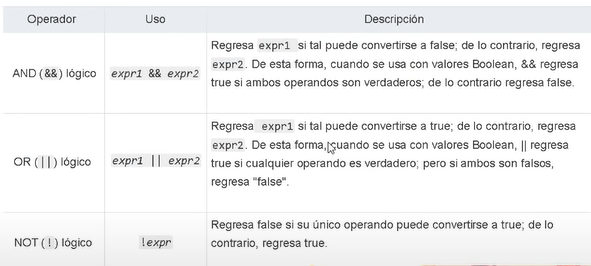
El **document.write** lo que hace es escribir directo en la página.

* **Operadores de comparación:** Los operadores de comparación, comparan dos expresiones y devuelven un valor booleano que representa la relación de sus valores.

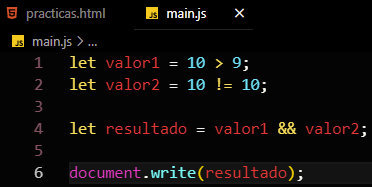
****

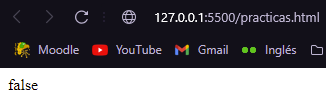
****

**Operadores lógicos:** Los operadores lógicos nos devuelven un resultado a partir de que se cumpla o no una condición, el resultado es booleano y sus operandos son valores lógicos o asimilables a ellos.

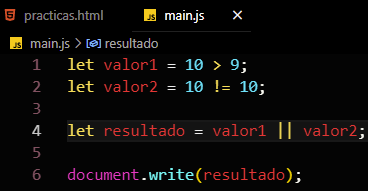


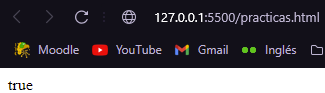
**Ejemplo 1:** En AND se tienen que cumplir las dos.



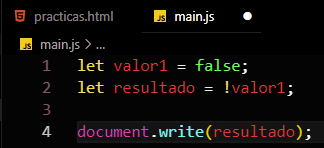


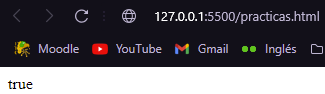
**Ejemplo 2:** En OR se tiene que cumplir solo una.





**Ejemplo 3:**  Niega el valor.

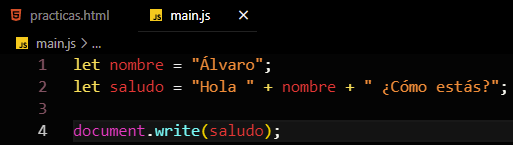


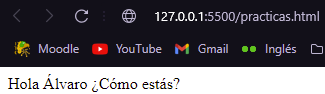


**8) Concatenación.**

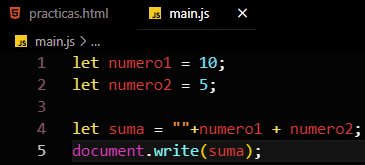
La concatenación en palabras fáciles es unir Strings.

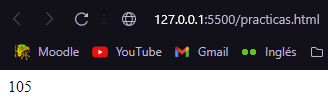
* **Con el signo +**



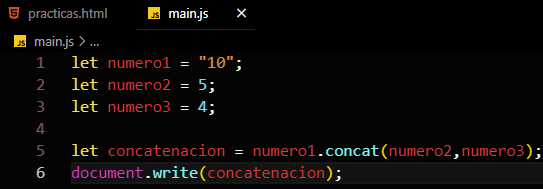


* **Forzar String (“” + 1 + 1)**

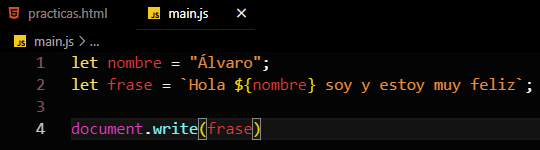
****

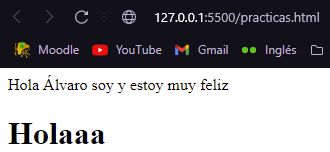
****

* **Concat:** (El concat solo funciona con String al principio, ya que es un método de cadenas)

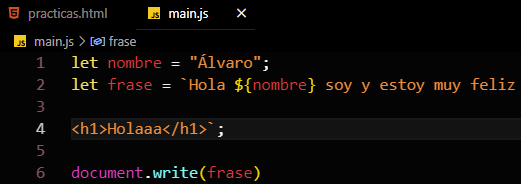
****

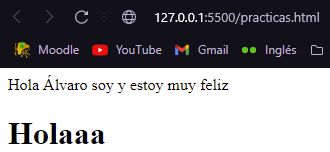
* **Backtiks y la variable entre ${}:** Esta sería la mejor forma de concatenar y la más recomendada.

****

****

De igual forma, las Backtiks nos permiten escribir código HTML de manera simple y rápida.





**9) Camel case.**

Camel case, es una forma de escribir código, y hace referencia a escribir la primera letra en minúscula y cada nueva palabra en mayúscula.

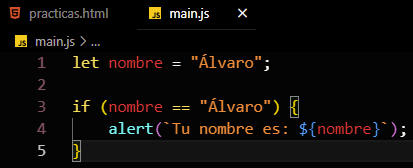
**Ejemplo:** let **n**ombre**E**mpleado = “Pedro”;

**Ejemplo 2:** document.get**B**y**F**irst();

**10) Condicionales.**

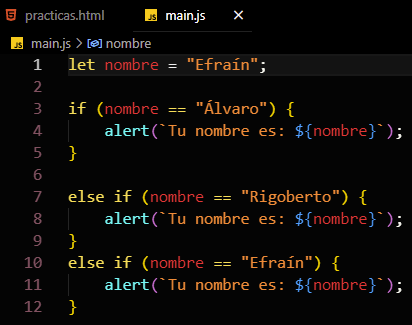
Un condicional es una sentencia que nos permite validar expresiones.

* **If:** Por su traducción podemos tomarlo como un: “Si esto se cumple, haz esto.” Y es la condición principal.



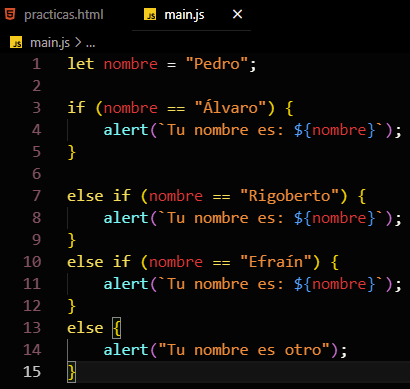


* **else if:** Podemos verlo como un: “Entonces si esto se cumple, haz esto otro.” Y podemos tener tantos como sean necesarios.





* **else:** Podemos verlo como la última instancia, ya que si no se cumple ninguna de las anteriores va a llegar a esta





**Capítulo 2.**

**11) Arrays.**

Los arrays ya no son un dato primitivo, ya son objetos. En estos podemos guardar más de un dato e incluso de diferentes tipos. Existen varias formas de crear un array, pero hay 2 las cuales son más comunes.