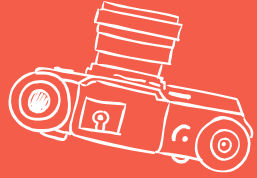







PROGRAMACIÓN IMPERATIVA EN JAVASCRIPT





PROGRAMA

- 
- 
- 
- 
- 
1. ENTORNO DE PROGRAMACIÓN
 2. PROGRAMAR EN EL EDITOR
 3. VARIABLES, CONSTANTES Y SUS TIPOS
 4. OPERACIONES BÁSICAS
 5. CONDICIONALES
 6. CICLOS
 7. FUNCIONES



REPASO DE LA CLASE ANTERIOR

1. ¿Qué es una variable? ¿Por qué son importantes?
2. Diferencias entre var y let
3. Cómo inicializar y actualizar una variable
4. Nomenclatura de nombres de variables
5. Tipos de variables: number, strings, booleanos, arreglos y objetos
6. Cómo acceder a los índices de un arreglo
7. Cómo acceder a los atributos de un objeto
8. Tipado dinámico en JavaScript
9. Constantes en JavaScript



ENTREGA PRÁCTICO NO

- fecha límite: jueves 1 de junio, 22hrs.
- via github con un tag en el repo
 - `git tag practico0`
 - `git push origin main --tags`
- las preguntas del práctico en un file respuestas.txt

GLOSARIO

<https://developer.mozilla.org/es/docs/Glossary>

TIPOS DE VARIABLES

- **number:** 3.14, -4.12 => `let pi = 3.14...;`
- **string:** “hola mundo” => `let goodbye = 'Hasta luego';`
- **booleanos:** true/false => `let iAmAlive = true;`

TIPOS DE VARIABLES

- **undefined** => `let myVar;`
- **null** => `myVar = null;`
- **symbol** => `let mySym = Symbol();`

TIPOS DE VARIABLES: SYMBOL

- `console.log(Symbol(1) === Symbol(1));`
- `console.log(Symbol('foo') === Symbol('foo'));`

TIPOS DE VARIABLES: SYMBOL

SIN SYMBOL

```
let people = [];  
people['bob'] = { name: 'Bob Smith' };  
people['bob'] = { name: 'Bob Jones' };
```

CON SYMBOL

```
let symbolPeople = [];  
symbolPeople[Symbol('bob')] = { name: 'Bob Smith' };  
symbolPeople[Symbol('bob')] = { name: 'Bob Jones' };
```

PENDIENTES DE LA CLASE PASADA: ARREGLOS

- colección ordenada
 - `let myArray = [3, 2, 7, 9];`
- tipos de datos mixtos
 - `let myMixedArray = [0, 2.3, "hello", {foo: "bar"}];`
- propiedad **length**
 - `myArray.length;` $\Rightarrow 4$
 - `myMixedArray.length;` $\Rightarrow 4$

PENDIENTES DE LA CLASE PASADA: ARREGLOS

- tamaño dinámico
 - `myMixedArray[4] = "bye";`
- indexación basada en cero
 - primer elemento: `myArray[0];`
 - ultimo elemento: `myArray[myArray.length - 1];`
- mutabilidad
 - `myMixedArray[0] = 3.44;`

4.

OPERACIONES BÁSICAS



OPERADORES ARITMÉTICOS

Operador	Nombre	Propósito	Ejemplo
+	Adición	Suma dos números juntos.	$6 + 9$
-	Resta	Resta el número de la derecha del de la izquierda.	$20 - 15$
*	Multiplicación	Multiplica dos números juntos.	$3 * 7$
/	División	Divide el número de la izquierda por el de la derecha.	$10 / 5$
%	Sobranter (también llamado módulo)	Retorna el restante después de dividir el número de la izquierda en porciones enteras del de la derecha.	$8 \% 3$ (retorna 2, como tres está dos veces en 8, quedando 2 restantes.)

OPERADORES INCREMENTO Y DECREMENTO

- incremento:

```
let incremento = 3;
```

```
incremento++;
```

- decremento:

```
let decremento = 5;
```

```
decremento—;
```

OPERADORES DE ASIGNACIÓN

Operador	Nombre	Propósito	Ejemplo	Atajo de
<code>+=</code>	Adición asignación	Suma el valor de la derecha al valor de la variable de la izquierda y retorna el nuevo valor	<code>x = 3;</code> <code>x += 4;</code>	<code>x = 3;</code> <code>x = x + 4;</code>
<code>-=</code>	Resta asignación	Resta el valor de la derecha, del valor de la variable de la izquierda y retorna el nuevo valor.	<code>x = 6;</code> <code>x -= 3;</code>	<code>x = 6;</code> <code>x = x - 3;</code>
<code>*=</code>	Multiplicación asignación	Multiplica el valor de la variable en la izquierda por el valor en la derecha y retorna el nuevo valor.	<code>x = 2;</code> <code>x *= 3;</code>	<code>x = 2;</code> <code>x = x * 3;</code>
<code>/=</code>	División asignación	Divide el valor de la variable en la izquierda por el valor de la derecha y retorna el nuevo valor.	<code>x = 10;</code> <code>x /= 5;</code>	<code>x = 10;</code> <code>x = x / 5;</code>

OPERADORES DE COMPARACIÓN

Operador	Nombre	Propósito	Ejemplo
<code>===</code>	Igual estricto	Comprueba si los valores izquierdo y derecho son idénticos entre sí	<code>5 === 2 + 4</code>
<code>!==</code>	Igual no-estricto	Comprueba si los valores izquierdo y derecho no son idénticos entre sí	<code>5 !== 2 + 3</code>
<code><</code>	Menor que	Comprueba si el valor izquierdo es menor que el derecho.	<code>10 < 6</code>
<code>></code>	Mayor que	Comprueba si el valor izquierdo es mayor que el derecho.	<code>10 > 20</code>
<code><=</code>	Menor o igual a	Comprueba si el valor izquierdo es menor o igual que el derecho.	<code>3 <= 2</code>
<code>>=</code>	Mayor o igual a	Comprueba si el valor izquierdo es mayor o igual que el derecho..	<code>5 >= 4</code>

MANEJO DE STRINGS

- `toLowerCase` \Rightarrow `'Hola'.toLowerCase();`
- `toUpperCase` \Rightarrow `'Hola'.toUpperCase();`
- `replace` \Rightarrow `'Hola Mundo'.replace('Mundo', 'Amigo');`
- `split` \Rightarrow `'Hola Mundo'.split(' ');`

MANEJO DE STRINGS

- `startsWith` `=> 'Hola Mundo'.startsWith('Hola');`
- `endsWith` `=> 'Hola Mundo'.endsWith('Mundo');`
- `trim` `=> ' Hola Mundo '.trim();`
- `charAt` `=> 'Hola'.charAt(2);`

STRINGS Y NUMBERS

1. `"hello" + 123;`
2. `"19" + "95";`
3. `Number("1234");`
4. `let myNumber = 123; myNumber.toString();`
5. `parseInt("50"); parseInt("50.55");`
6. `parseFloat("50"); parseFloat("50.55");`
7. `String(43);`