**JavaScript**

🔤 **STRING**

|  |  |  |
| --- | --- | --- |
| **Método / Propiedad** | **Ejemplo** | **Qué hace** |
| **.length** | "hola".length | Devuelve el número de caracteres del string. |
| **.toLowerCase()** | "Hola".toLowerCase() | Convierte todo el texto a minúsculas. |
| **.toUpperCase()** | "hola".toUpperCase() | Convierte todo el texto a mayúsculas. |
| **.includes(valor)** | "aeiou".includes("e") | Verifica si el string contiene el valor indicado. |
| **charAt(posición)** | "hola".charAt(1) | Devuelve el carácter en la posición especificada. |
| **string[i]** | "hola"[0] | Forma alternativa de acceder al carácter en una posición. |
| **.trim()** | " hola ".trim() | Elimina espacios al inicio y al final del texto. |
| **.replace(a, b)** | "hola".replace("a", "o") | Reemplaza una parte del texto por otra. |
| **.split(separador)** | "a,b,c".split(",") | Divide un string en un array, usando el separador. |

📚 **ARRAY**

|  |  |  |
| --- | --- | --- |
| **Método / Propiedad** | **Ejemplo** | **Qué hace** |
| **.push(valor)** | arr.push("Mónica") | Agrega un elemento al final del array. |
| **.pop()** | arr.pop() | Elimina el último elemento del array. |
| **.shift()** | arr.shift() | Elimina el primer elemento |
| **.** **unshift(valor)** | arr.unshift(1) | Agrega al inicio del array |
| **.length** | arr.length | Devuelve la cantidad de elementos del array. |
| **.filter(fn)** | arr.filter(x => x > 10) | Crea un nuevo array con los elementos que cumplan la condición. |
| **.includes(valor)** | arr.includes("Luis") | Devuelve true si el valor existe en el array. |
| **.forEach(fn)** | arr.forEach(x => console.log(x)) | Itera cada elemento del array. |
| **.map(fn)** | arr.map(x => x \* 2) | Crea un nuevo array aplicando la función a cada elemento. |

🔢 **NUMBER**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operador / Método** | **Ejemplo** | **Qué hace** |  |
| **parseInt()** | parseInt("25") | Convierte un string a número entero. |  |
| **parseFloat()** | parseFloat("3.14") | Convierte un string a número decimal. |  |
| **% (módulo)** | 5 % 2 | Devuelve el residuo de la división. |  |
| **contador++** | contador++ | Suma 1 al valor. |  |
| **Math.round()** | Math.round(2.6) | Redondea al entero más cercano. |  |
| **Math.floor()** | Math.floor(2.9) | Redondea hacia abajo. |  |
| **Math.ceil()** | Math.ceil(2.1) | Redondea hacia arriba. |  |
| **isNaN(valor)** | isNaN("hola") | Retorna true si el valor no es un número. |  |

🔁 **CONTROL DE FLUJO**

|  |  |  |
| --- | --- | --- |
| **Estructura** | **Ejemplo** | **Qué hace** |
| **if / else** | if (x > 0) { ... } else { ... } | Ejecuta bloques según una condición. |
| **for** | for (let i = 0; i < 5; i++) | Repite el bloque varias veces según condición. |
| **while** | while (x < 10) | Repite el bloque mientras se cumpla la condición. |
| **do...while** | do { ... } while (condición) | Igual que while pero se ejecuta al menos una vez. |
| **switch / case** | switch (opción) { case "a": ... } | Evalúa múltiples opciones. |
| **break** | break; | Sale del bucle o switch. |
| **continue** | continue; | Salta a la siguiente iteración del bucle. |

🖥️ **CONSOLA Y ENTRADA**

|  |  |  |
| --- | --- | --- |
| **Método** | **Ejemplo** | **Qué hace** |
| **console.log()** | console.log("Hola") | Imprime texto o variables en la consola. |
| **prompt()** | prompt("Tu nombre:") | Solicita al usuario una entrada. |
| **alert()** | alert("Hola") | Muestra un cuadro de alerta al usuario. |
| **confirm()** | confirm("¿Estás seguro?") | Muestra un cuadro con aceptar/cancelar. |

**Tabla de Métodos y Propiedades para Objetos en JavaScript**

|  |  |  |
| --- | --- | --- |
| **Método / Propiedad** | **Ejemplo** | **¿Qué hace?** |
| **objeto.propiedad** | persona.nombre | **Accede a una propiedad de forma directa.** |
| **objeto["propiedad"]** | persona["edad"] | **Accede a una propiedad con nombre dinámico.** |
| **objeto.propiedad = valor** | persona.edad = 30 | **Modifica o crea una propiedad.** |
| **delete objeto.propiedad** | delete persona.nombre | **Elimina una propiedad del objeto.** |
| **for...in** | for (let clave in objeto) | **Recorre todas las propiedades enumerables del objeto.** |
| **Object.keys(objeto)** | Object.keys(libro) | **Devuelve un array con las claves del objeto.** |
| **Object.values(objeto)** | Object.values(libro) | **Devuelve un array con los valores del objeto.** |
| **Object.entries(objeto)** | Object.entries(libro) | **Devuelve un array de arrays: cada uno contiene [clave, valor].** |
| **hasOwnProperty("clave")** | persona.hasOwnProperty("edad") | **Verifica si el objeto tiene esa propiedad directamente (no heredada).** |
| **typeof valor** | typeof persona.edad | **Determina el tipo de dato de una propiedad.** |
| **Object.assign()** | Object.assign(destino, fuente) | **Copia propiedades de uno o más objetos a otro.** |
| **JSON.stringify(objeto)** | JSON.stringify(persona) | **Convierte el objeto en un string JSON.** |
| **JSON.parse(cadena)** | JSON.parse('{"nombre": "Sam"}') | **Convierte una cadena JSON en un objeto.** |

📚 **Tabla de Métodos Esenciales de Arrays en JavaScript**

|  |  |  |
| --- | --- | --- |
| **Método** | **Ejemplo** | **¿Qué hace?** |
| **push()** | arr.push(4) | **Agrega un elemento al final del array.** |
| **pop()** | arr.pop() | **Elimina el último elemento del array.** |
| **shift()** | arr.shift() | **Elimina el primer elemento del array.** |
| **unshift()** | arr.unshift(0) | **Agrega un elemento al inicio del array.** |
| **forEach()** | arr.forEach(item => console.log(item)) | **Ejecuta una función por cada elemento del array.** |
| **map()** | let dobles = arr.map(num => num \* 2) | **Crea un nuevo array con los resultados de aplicar una función.** |
| **filter()** | let mayores = arr.filter(num => num > 5) | **Crea un nuevo array con los elementos que cumplen una condición.** |
| **find()** | let encontrado = arr.find(num => num === 3) | **Devuelve el primer elemento que cumple con la condición.** |
| **includes()** | arr.includes(3) | **Retorna true si el valor está en el array.** |
| **indexOf()** | arr.indexOf(3) | **Devuelve el índice de la primera aparición del elemento.** |
| **slice()** | arr.slice(1, 3) | **Devuelve una copia parcial del array (no lo modifica).** |
| **splice()** | arr.splice(1, 2) | **Elimina o reemplaza elementos dentro del array (sí lo modifica).** |
| **concat()** | arr.concat(otroArray) | **Une dos arrays y devuelve un nuevo array.** |
| **join()** | arr.join(", ") | **Une todos los elementos del array en un string.** |
| **reduce()** | arr.reduce((acc, val) => acc + val, 0) | **Reduce el array a un solo valor acumulando resultados.** |
| **reverse()** | arr.reverse() | **Invierte el orden del array (lo modifica).** |
| **sort()** | arr.sort((a, b) => a - b) | **Ordena los elementos del array (cuidado: lo modifica).** |

**Git and GitHub**

**🧭 Comandos básicos de Git**

|  |  |
| --- | --- |
| **Comando Git** | **Descripción** |
| **git init** | Crea un nuevo repositorio Git local. |
| **git status** | Muestra el estado actual del repositorio. |
| **git add <archivo>** | Agrega archivos al área de preparación (staging). |
| **git add .** | Agrega todos los archivos modificados al staging. |
| **git commit -m "mensaje"** | Guarda los cambios con un mensaje. |
| **git log** | Muestra el historial de commits. |
| **git diff** | Muestra diferencias entre archivos. |

**🔁 Trabajando con ramas (branches)**

|  |  |
| --- | --- |
| **Comando Git** | **Descripción** |
| **git branch** | Lista todas las ramas locales. |
| **git branch <nombre>** | Crea una nueva rama. |
| **git checkout <rama>** | Cambia a otra rama. |
| **git switch <rama>** | Alternativa moderna a checkout. |
| **git merge <rama>** | Fusiona una rama con la rama actual. |
| **git branch -d <rama>** | Elimina una rama local. |

**🔄 Trabajar con GitHub (remoto)**

|  |  |
| --- | --- |
| **Comando Git** | **Descripción** |
| **git remote add origin <url>** | Conecta el repositorio local con GitHub. |
| **git push -u origin main** | Sube los cambios a GitHub y configura la rama remota por defecto. |
| **git push** | Envía commits al repositorio remoto. |
| **git pull** | Trae y fusiona cambios del repositorio remoto. |
| **git clone <url>** | Clona un repositorio remoto a tu equipo local. |

**💡 Utilidades adicionales**

|  |  |
| --- | --- |
| **Comando Git** | **Descripción** |
| **git config --global user.name** | Configura tu nombre de usuario para todos los proyectos. |
| **git config --global user.email** | Configura tu correo para todos los proyectos. |
| **git stash** | Guarda cambios no confirmados para aplicarlos después. |
| **git restore <archivo>** | Revierte un archivo a su última versión confirmada. |
| **git reset --hard** | Elimina todos los cambios no confirmados. |

**Tabla de Comandos y Funciones esenciales de Git & GitHub**

|  |  |  |  |
| --- | --- | --- | --- |
| **Categoría** | **Comando / Método** | **Ejemplo** | **¿Qué hace?** |
| **Inicialización** | git init | git init | Crea un nuevo repositorio Git en el directorio actual. |
| **Configuración** | git config | git config --global user.name "Sam" | Configura el nombre de usuario. |
| **Seguimiento** | git add | git add index.html | Agrega archivos al área de preparación (staging). |
| **Confirmación** | git commit | git commit -m "Mensaje" | Guarda una instantánea del estado actual. |
| **Estado** | git status | git status | Muestra los archivos modificados y su estado. |
| **Ver historial** | git log | git log | Muestra el historial de confirmaciones. |
| **Conexión remota** | git remote add | git remote add origin URL | Conecta el repo local con uno en GitHub. |
| **Enviar cambios** | git push | git push origin main | Sube los cambios al repositorio remoto. |
| **Traer cambios** | git pull | git pull origin main | Descarga y fusiona cambios del repositorio remoto. |
| **Clonar repo** | git clone | git clone URL | Crea una copia local de un repo remoto. |

**💻 Tabla de Atajos y Funciones Esenciales de VS Code**

|  |  |  |  |
| --- | --- | --- | --- |
| **Categoría** | **Atajo / Función** | **Ejemplo / Tecla** | **¿Qué hace?** |
| Navegación | **Ir a archivo** | Ctrl + P | Abre búsqueda rápida para ir a un archivo. |
| Navegación | **Ir a línea** | Ctrl + G | Saltar a una línea específica. |
| Navegación | **Ir a definición** | F12 | Salta a donde está definida una función o variable. |
| Búsqueda | **Buscar** | Ctrl + F | Busca texto dentro del archivo actual. |
| Búsqueda | **Buscar en todos los archivos** | Ctrl + Shift + F | Busca texto en todo el proyecto. |
| Edición | **Seleccionar todo** | Ctrl + A | Selecciona todo el contenido del archivo. |
| Edición | **Duplicar línea** | Shift + Alt + ↓ / ↑ | Duplica la línea actual hacia abajo o arriba. |
| Edición | **Eliminar línea** | Ctrl + Shift + K | Elimina la línea donde está el cursor. |
| Edición | **Comentario** | Ctrl + / | Comenta o descomenta la línea seleccionada. |
| Edición múltiple | **Editar múltiples líneas** | Alt + Click | Agrega cursores múltiples para editar a la vez. |
| Organización | **Formatear código** | Shift + Alt + F | Aplica sangría y acomoda el código automáticamente. |
| Organización | **Renombrar variable** | F2 | Renombra de forma inteligente en todo el archivo. |
| Terminal | **Nueva terminal integrada** | Ctrl + Ñ (o Ctrl + `) | Abre una terminal en la parte inferior. |
| Archivo | **Crear nuevo archivo** | Ctrl + N | Crea un archivo nuevo. |
| Archivo | **Abrir carpeta** | Ctrl + K, Ctrl + O | Abre una carpeta de proyecto. |
| Panel lateral | **Mostrar / Ocultar panel lateral** | Ctrl + B | Muestra u oculta el panel lateral izquierdo. |
| Extensiones | **Abrir marketplace** | Ctrl + Shift + X | Abre la vista de extensiones. |
| Vista dividida | **Dividir editor** | Ctrl + \ | Divide la vista del editor en dos. |
| Explorador | **Mostrar barra de archivos** | Ctrl + Shift + E | Muestra el explorador de archivos. |
| Temas | **Cambiar tema** | Ctrl + K luego Ctrl + T | Permite cambiar el tema de color de VS Code. |