DEV.F::

Clase 3

Arreglos y ciclos en JavaScript

Temas de la clase (180 min)

Ciclo While (x-x min)

Bucles

¿Qué es un while y cómo funciona?

Estructura

Ventajas y desventajas

Ciclo For (x-x min)

¿Qué es un ciclo For?

Estructura

Arrays (x-x min)

if

if y else

else if

Entregable: Clasificación de Frutas 🍓 🍒 🥑

DEV.F::

Objetivos de la clase

Al finalizar esta clase deberías ser capaz de entender qué son los ciclos y entender cómo se diferencía un while de un for.

También es importante que manipules los arrays combinado con ciclos, para eso el ejercicio de las frutas.

DEV.F::

Bucle

Un **bucle** es una estructura de control que **repite instrucciones**.

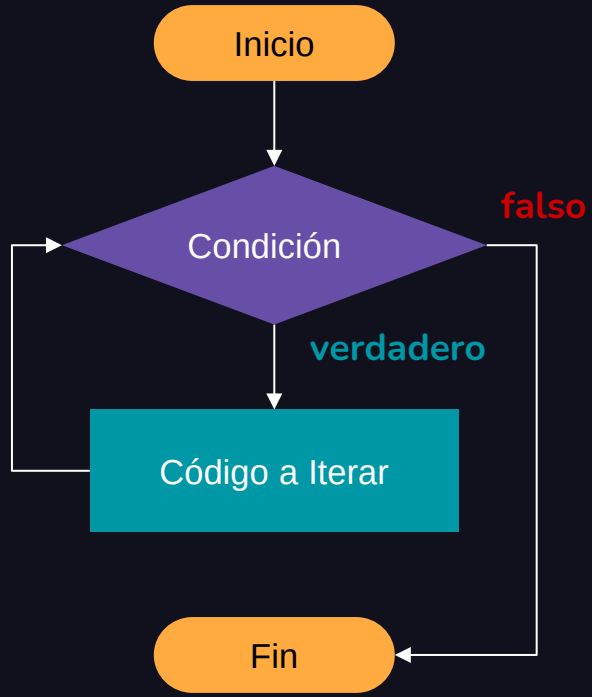
Un bucle entonces nos permite repetir un bloque de instrucciones determinado hasta que se cumpla cierta condición.

Cada repetición suele llamarse <u>iteración</u>

Ciclo While

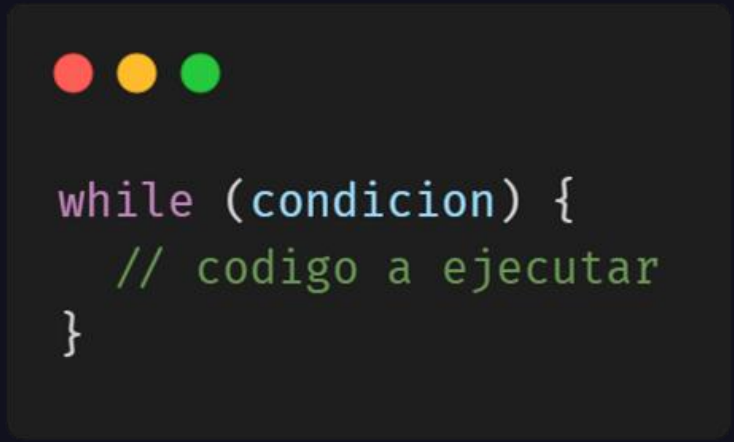
La idea principal del ciclo while es: MIENTRAS se cumpla la condición REALIZAR estas acciones. Cuando la condición deje de cumplirse salimos del bucle y continúa el flujo del programa.

Muy importante: En el ciclo while la condición es lo primero que se evalúa, antes de ejecutar el código a iterar.



Estructura

Usamos la palabra reservada ***while***, seguida de la condición entre paréntesis ***()*** y finalmente colocamos el código que se repetirá entre llaves ***{}***



Importante: Necesitamos en el código a iterar insertar una variable de control que nos permita salir eventualmente el ciclo while. En caso contrario nuestro programa se quedará ciclado “infinitamente”.

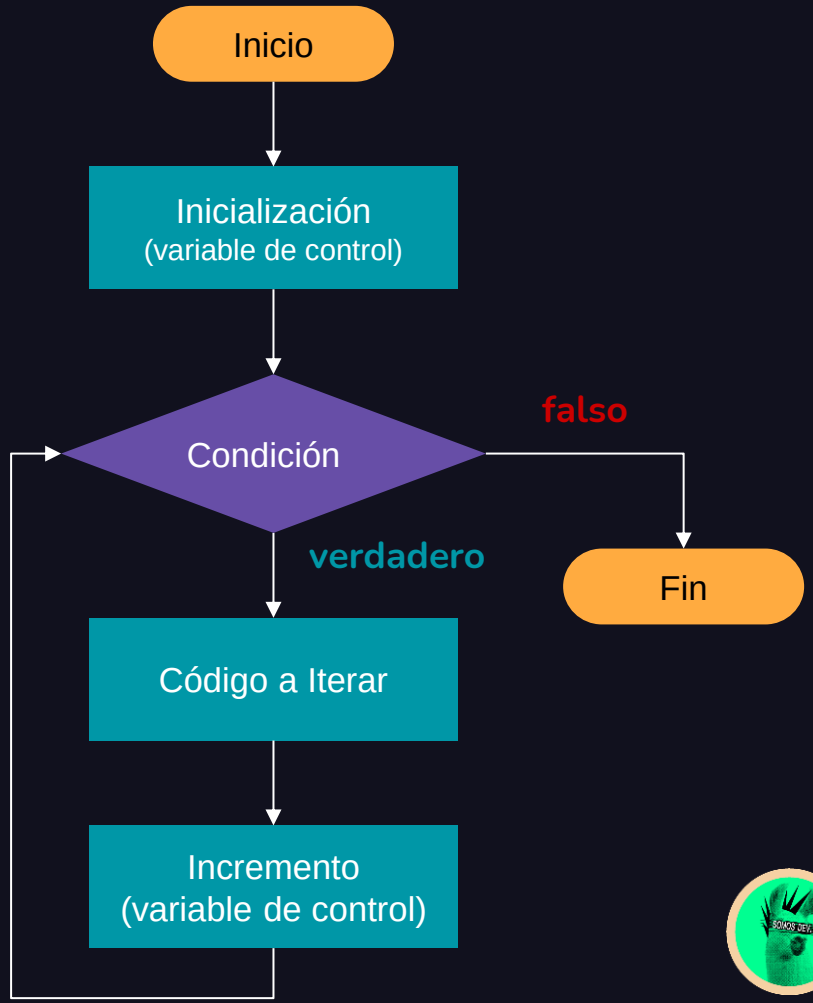
Casos de uso del ciclo While

Contar hasta un número específico: Como en los ejemplos anteriores, se puede usar para contar, realizar operaciones repetitivas y más.

Leer datos hasta que se cumpla una condición: Ideal para recibir entradas del usuario.

Ciclo For

Un **bucle for** es un bucle que **repite** el bloque de instrucciones **un número predeterminado de veces.**



DEV.F::

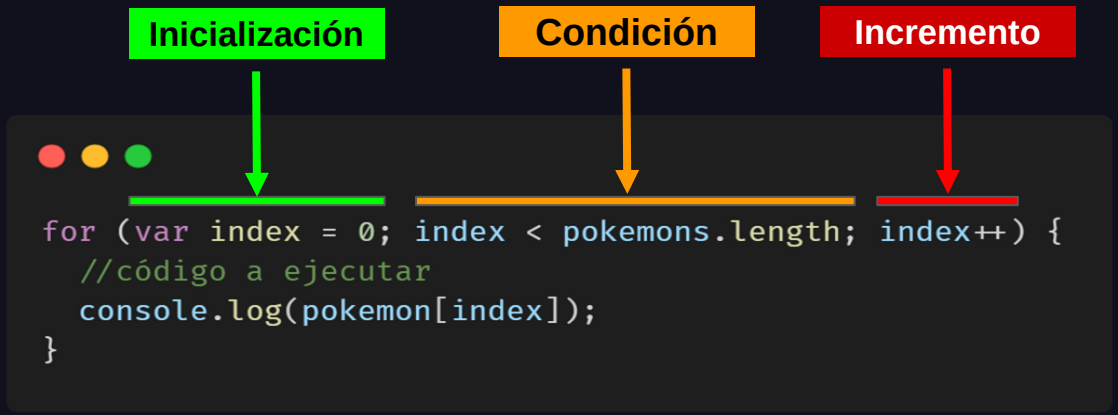
Contadores y acumuladores

En muchos programas se necesitan variables que cuenten cuántas veces ha ocurrido algo (contadores) o que acumulen valores (acumuladores).

Se entiende por **contador** una variable que lleva la cuenta del número de veces que se ha cumplido una condición.

Se entiende por **acumulador**, una variable que acumula el resultado de una operación.

Sintaxis Ciclo for



Inicialización: Se ejecuta una vez al inicio del ciclo. Generalmente se usa para declarar e inicializar una variable contadora.

Condición: Mientras la condición se cumpla, se ejecutará el código dentro de las llaves { }.

Incremento: Se ejecuta después de cada iteración, se utiliza para modificar la variable contadora.

Casos de uso del ciclo for

Procesamiento de Listas: Es común en el desarrollo web y la programación en general para procesar listas de datos, como resultados de consultas a bases de datos.

Generación de Reportes: Se utiliza para iterar sobre datos y generar reportes, resúmenes o análisis.

Arrays

Arreglos o Matrices (Arrays)

Los Arreglos **“son una manera ordenada”** de almacenar una lista de elementos de datos bajo un solo nombre de **variable**, pudiendo acceder a cada elemento individual de la lista.

Un arreglo se representa con corchetes [ ], dentro se coloca el contenido. Cada elemento es separado por coma.

```
const alumnosMali = []; // arreglo vacio

alumnosMali = ['Willinton','Laura','Jorge','Luis','Rick'];
```

DEV.F::

¿En dónde se usan los arrays?

Almacenamiento de Listas: como nombres de usuarios, productos, etc.

Manejo de Datos: para guardar datos de formularios, respuestas de API...

Manipulación de Datos: para que se puedan crear operaciones como ordenar, filtrar o transformar colecciones de datos

Matrices y Estructuras de Datos Complejas: por ejemplo matrices multidimensionales

DEV.F::

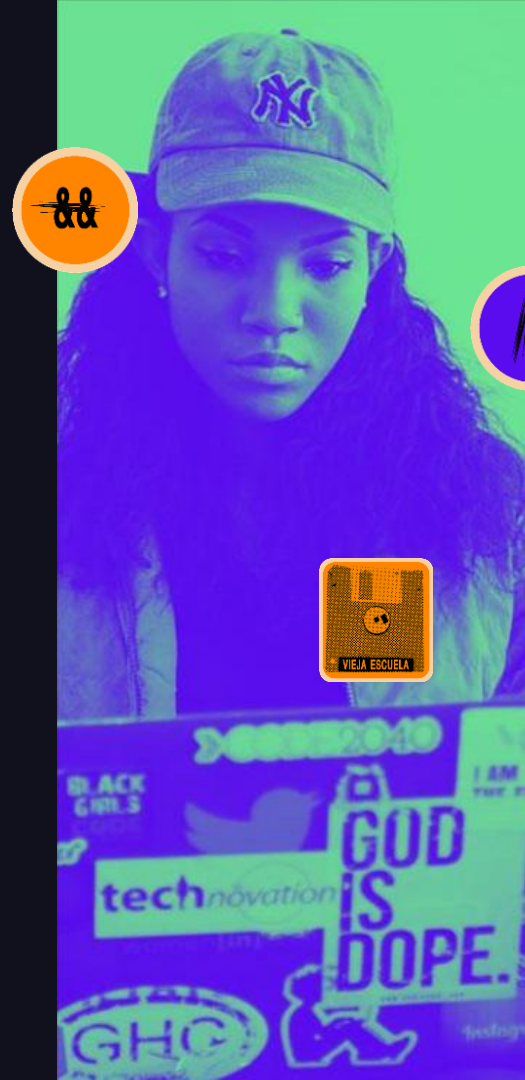
Características de un array

Los arrays tienen las siguientes características:

Índices: Cada elemento tiene una posición numerada que comienza en 0.

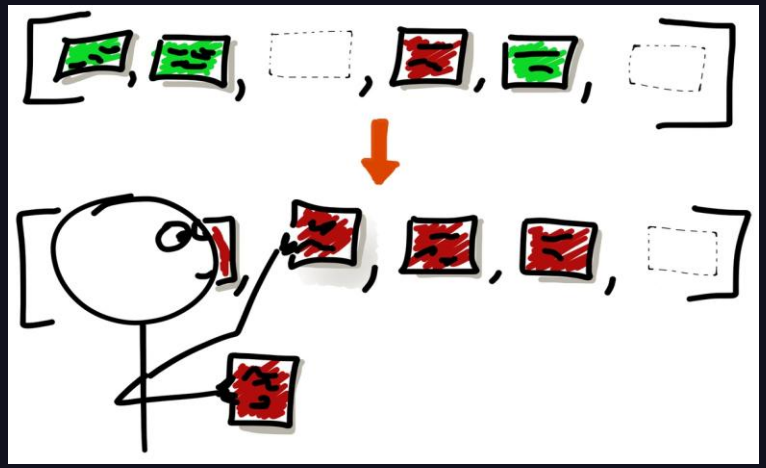
Tipo de datos: Pueden almacenar múltiples valores, y en JavaScript pueden contener diferentes tipos de datos.

Acceso a elementos: Se puede acceder a un elemento usando su índice, por ejemplo, array[0] devuelve el primer valor.



Métodos en los arrays

Los métodos de los arrays son acciones o funciones especiales que podemos usar en los arreglos para modificarlos, recorrerlos o hacer cálculos con sus elementos.



DEV.F::

Algunos métodos básicos en los arrays

1. push() → Agregar un elemento al final
2. pop() → Quitar el último elemento
3. unshift() → Agregar un elemento al inicio
4. shift() → Quitar el primer elemento
5. sort() → Ordenar un array
6. map() → Modificar todos los elementos

Podríamos presentar todos los métodos existentes, pero vale la pena mejor entender cómo trabajar bien con estos.

DEV.F::

Iterar un array

Iterar un arreglo significa **recorrer cada uno de sus elementos**, uno por uno, para hacer algo con ellos, como mostrarlos, modificarlos, sumarlos, etc. Es como revisar una lista de tareas e ir marcando cada una a medida que la completas.

Existen diferentes maneras de recorrer un array, pero de momento solo veremos cómo hacerlo con un *“for”*.

DEV.F::

Ejemplo de iterar un array con for

```
const animales = ["🐶", "🐱", "🐰", "🦁"];

for (let i = 0; i < animales.length; i++) {
    console.log(animales[i]);
}
```

Cada iteración del ciclo for imprime un elemento del arreglo en la consola:

🐶

🐱

🐰

🦁

DEV.F::