INTRODUCCIÓN AL DESARROLLO BACKEND

CON NODEJS

2023

ANDÉN Centro de Innovación y Emprendimientos Tecnológicos

SECRETARÍA DE EXTENSIÓN UNIVERSITARIA UTN - FRC SEU

UTN Facultad Regional Córdoba

Agencia CÓRDOBA JOVEN

CÓRDOBA entre todos

Profesor

RUBÉN ANÍBAL ROMERO

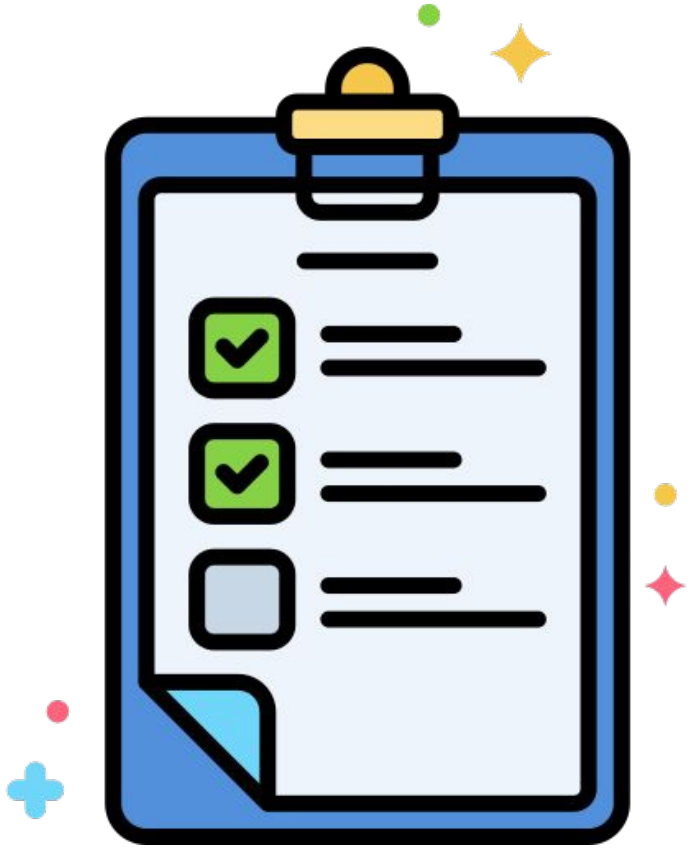
Ingeniero en Sistemas (UTN-FRC)
Especialista en Ingeniería de Negocios (UTN-FRC)
13 años de experiencia en la industria del software.
Arquitecto de Software en NaranjaX
Docente Universitario (UTN-FRC)
Docente Investigador

Objetivo del Curso

Introducir a los estudiantes en el mundo del desarrollo de aplicaciones web con un enfoque backend (en el lado del servidor) utilizando el lenguaje de programación JavaScript y la plataforma Node.js.

Cursado

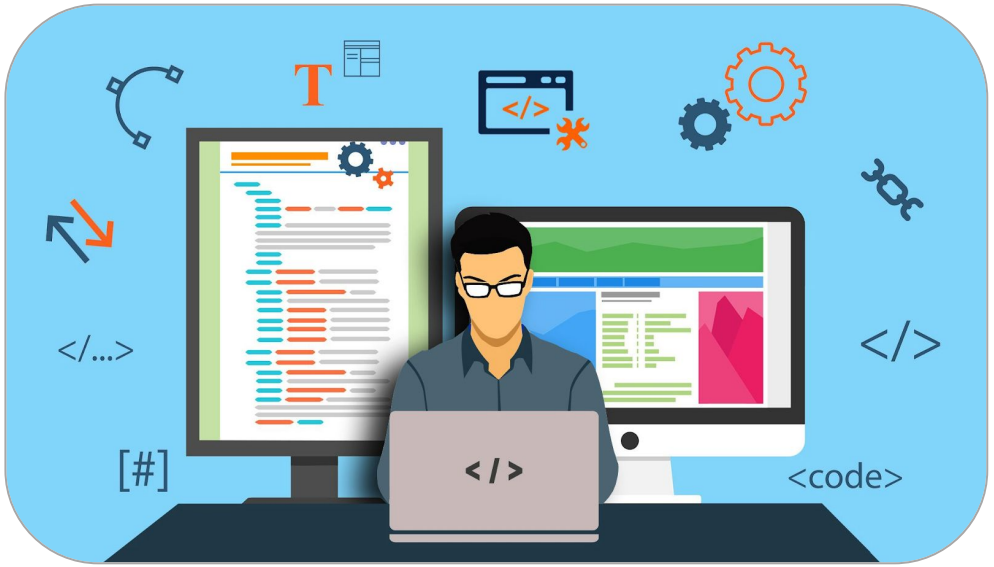
- 10 clases grabadas - 2 hs cada una.
- Cada clase tendrá contenidos con un ejemplo aplicado.
- Al finalizar la clase el alumno tendrá disponible actividad para realizar.
- Clase consulta una vez por semana. Horario y link de zoom en el aula virtual.



Clase 01 - Contenidos

- Introducción a la programación y al backend.
- Introducción a JavaScript y su uso en el backend.
- ¿Qué es Node.js y para qué se utiliza?

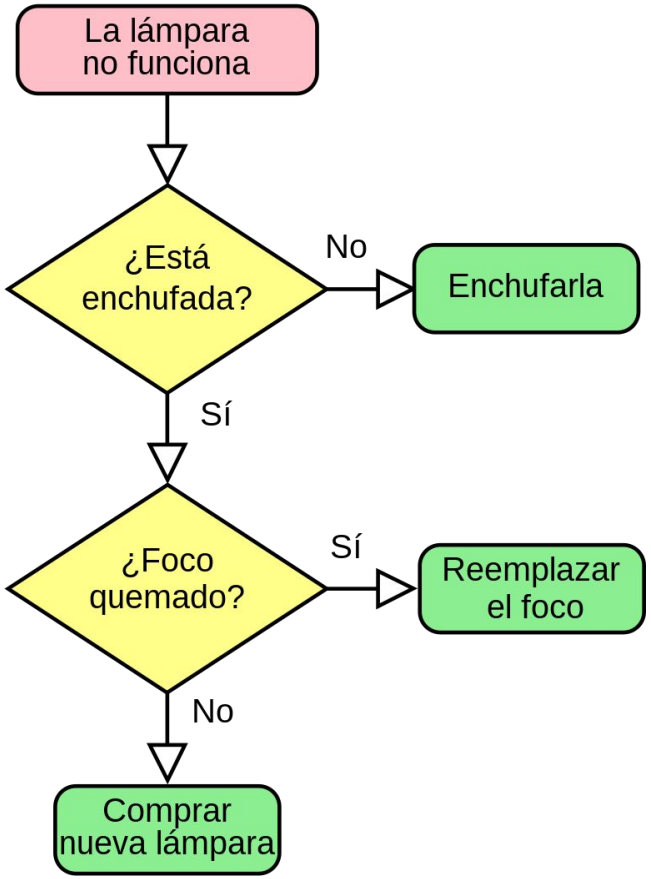
Introducción a la programación



- En la programación se utilizan **algoritmos**, que son **secuencias de pasos lógicos** que permiten resolver un problema.
- Los **lenguajes de programación** son herramientas que permiten escribir **algoritmos** que podemos escribir en un **código fuente** y que serán ejecutadas por una computadora.
- La programación es el proceso de **diseñar, escribir, probar y mantener el código fuente** de una aplicación o software.
- La programación es una **habilidad muy demandada en la industria tecnológica** y puede ser utilizada para una gran variedad de aplicaciones.

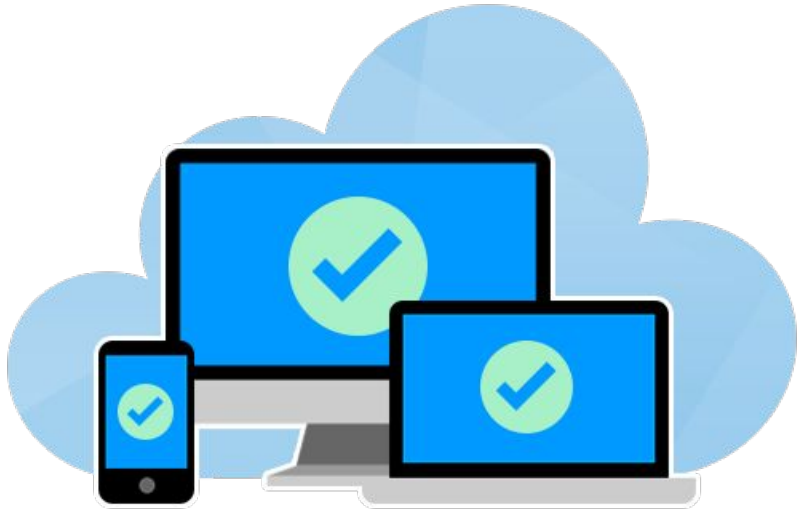
Un ejemplo de algoritmo

Un algoritmo es un conjunto de operaciones que busca resolver un problema determinado a través de secuencias lógicas.



¿Qué es una aplicación?

- Una aplicación es un programa diseñado para ejecutar una tarea o un conjunto de tareas específicas.
- Las aplicaciones pueden ser de escritorio, móviles o web.
- En general, se espera que una aplicación sea fácil de usar, eficiente y resuelva problemas cotidianos.
- Las aplicaciones pueden ser desarrolladas por empresas, equipos de programadores o individuos.



Ejemplo:
Un juego de carreras, una aplicación de mensajería, un editor de texto, una aplicación para pedir comida a domicilio, entre otros.

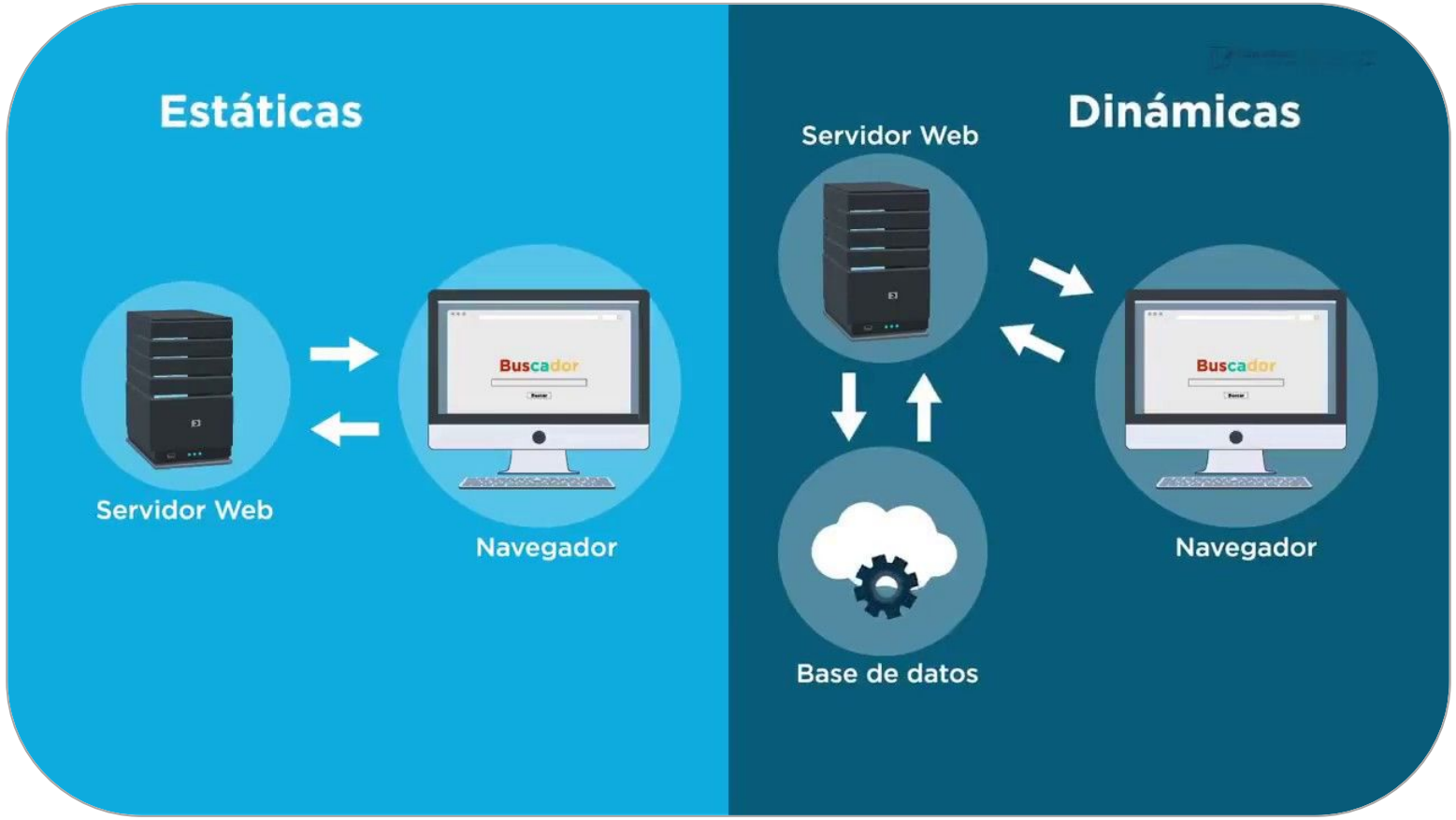
¿Qué es una aplicación web?

- Una aplicación web es una aplicación que se ejecuta en un navegador web y que permite realizar diversas tareas y actividades a través de internet.
- Las aplicaciones web son accesibles desde cualquier dispositivo con conexión a internet, lo que las hace muy convenientes para el usuario.

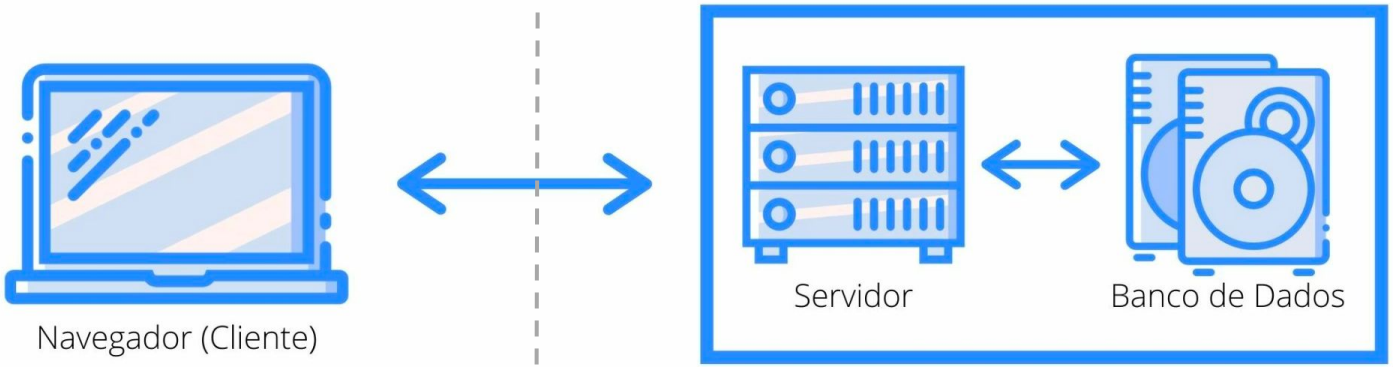
Ejemplo:
Desde aplicaciones simples como una calculadora o una lista de tareas hasta aplicaciones más complejas como un sistema de comercio electrónico (MercadoLibre).

¿Cómo funciona una aplicación web?

Aplicacion Web Estático vs Dinámica



Desarrollador Aplicación Web



Desarrollador FullStack

Desarrollador Frontend

- Se ocupa de la interfaz de usuario de la aplicación.
- Se asegura de que la aplicación sea fácil de usar, accesible y atractiva visualmente.
- Usa herramientas como HTML, CSS y JavaScript para crear la interfaz de usuario de la aplicación.

Desarrollador Backend:

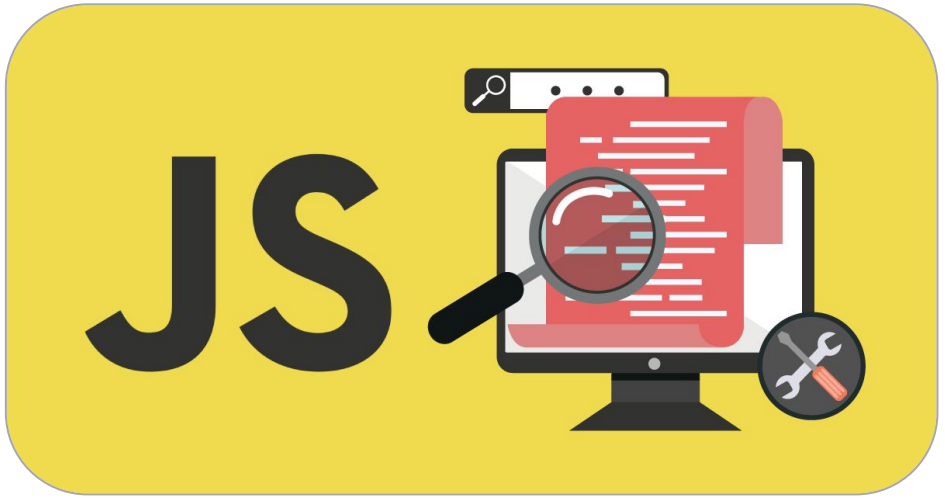
- Se ocupa de la lógica de negocio de la aplicación y la gestión de datos.
- Crea y mantiene la API y se encarga de integrar la aplicación con otras bases de datos y servicios externos.
- Usa herramientas como Node.js y Express.js.

¿Qué es el backend?

- El backend es la parte de una aplicación web que se encarga del procesamiento de datos y los algoritmos (lógica) de la aplicación.
- La lógica de la aplicación también podemos llamarla lógica de negocio.

¿Qué es JavaScript (js)?

- JavaScript es un **lenguaje de programación** que se utiliza para conectar el frontend con el backend.
- JavaScript **funciona en el navegador** web del usuario y puede interactuar con el contenido de una página web.



¿Por qué es importante JavaScript?

- Lenguaje de programación versátil y dinámico.
- Ampliamente utilizado en el desarrollo de aplicaciones web.
- Permite conectar frontend con backend y crear efectos visuales en el frontend.
- Utilizado tanto en el frontend como en el backend (con Node.js para crear aplicaciones escalables).

Sintaxis básica de JavaScript

- Javascript utiliza **variables** para almacenar datos y operadores para realizar operaciones matemáticas y lógicas.
- Las declaraciones de control de flujo, como "if" y "else", permiten tomar decisiones y ejecutar diferentes acciones en función de las condiciones.

La **sintaxis** se refiere a las reglas y convenciones que se utilizan para escribir un código en un lenguaje de programación.

Nodejs - Javascript para Backend

- Entorno de ejecución de JavaScript fuera del navegador (RE - Runtime Environment)
- Basado en el motor V8 de Google Chrome
- Orientado a aplicaciones de servidor y herramientas de desarrollo.
- Gestión de paquetes a través de npm (Node Package Manager).

¿Por qué utilizar node.js en el back-end?

- **Mismo lenguaje** JavaScript en el cliente y en el servidor
- **Muy Eficiente** porque utiliza modelo de E/S sin bloqueo y basado en eventos, lo que permite manejar un gran número de conexiones concurrentes sin agotar los recursos del servidor.
- **Altamente escalable**: manejar grandes cantidades de tráfico y conexiones.
- **Amplia variedad de bibliotecas y módulos**
- **Comunidad de desarrolladores muy activa**

Variables / Constantes

- Una variable es un **contenedor** que se utiliza para **almacenar datos**. Se puede pensar en una variable como una caja con un nombre en la que puedes guardar un valor. Estos valores pueden ser números, cadenas de texto, objetos, arreglos u otros tipos de datos.
- Las **Variables** se crean con **let:**

```
let apellidoYNombre = "Juan Lopez";
let cantidadDisponible = 15;
cantidadDisponible = 30;
```

Buena práctica:
- Siempre utilizaremos **let** en vez de **var**.
- Usar camelCase para los nombres.

- Las **Constantes** se crean con **const** y son variables de sólo lectura (no se pueden modificar).

```
const PI = 3.141592;
const EDAD = 23;
```

Buena práctica:
- Usar mayúscula para los nombres.

- Las variables creadas sin un valor aún contienen el valor **undefined**.

Javascript: Tipos de datos

Javascript: Operadores aritméticos

+	suma
-	resta
*	producto
%	módulo
++	incremento
--	decremento

```
let a = 10;
let b = 5;

// Suma
let suma = a + b; // resultado: 15

// Resta
let resta = a - b; // resultado: 5

// Multiplicación
let multiplicacion = a * b; // resultado: 50

// División
let division = a / b; // resultado: 2

// Módulo (resto de la división)
let modulo = a % b; // resultado: 0

// Incremento
let incremento = a++; // resultado: 10 (a = 11 después del incremento)

// Decremento
let decremento = b--; // resultado: 5 (b = 4 después del decremento)
```

Ejemplo de código JavaScript

Aquí se muestra un ejemplo básico de código JavaScript para sumar dos números:

```
1   let numero1 = 5;
2   let numero2 = 7;
3   const resultado = numero1 + numero2;
4   console.log(resultado);
```

Explicación:
Este código define dos variables, "numero1" y "numero2", suma sus valores y almacena el resultado en la variable "resultado". Finalmente, se imprime el resultado en la consola con el comando "console.log()".

Javascript: Operadores lógicos

&&	and
||	or
!	not
?:	operador ternario

```
var edad = 25;
var tieneLicencia = true;
var esFeriado = true;
var tienePermiso = false;
var esNoche = false;

// Operador && (and): Ambas condiciones deben ser verdaderas para que se cumpla
if (edad >= 18 && tieneLicencia) {
  console.log("Puede conducir un automóvil.");
} else {
  console.log("No cumple los requisitos para conducir un automóvil.");
}

// Operador || (or): Al menos una de las condiciones debe ser verdadera para que se cumpla
if (esFeriado || tienePermiso) {
  console.log("Puede salir de vacaciones.");
} else {
  console.log("No puede salir de vacaciones.");
}

// Operador ! (not): Invierte el valor de verdad de una condición
if (!esNoche) {
  console.log("Es de día.");
} else {
  console.log("Es de noche.");
}

// Operador ternario
var mensaje = (edad >= 18) ? "Es mayor de edad" : "Es menor de edad";
console.log(mensaje);
```

Operadores de comparación

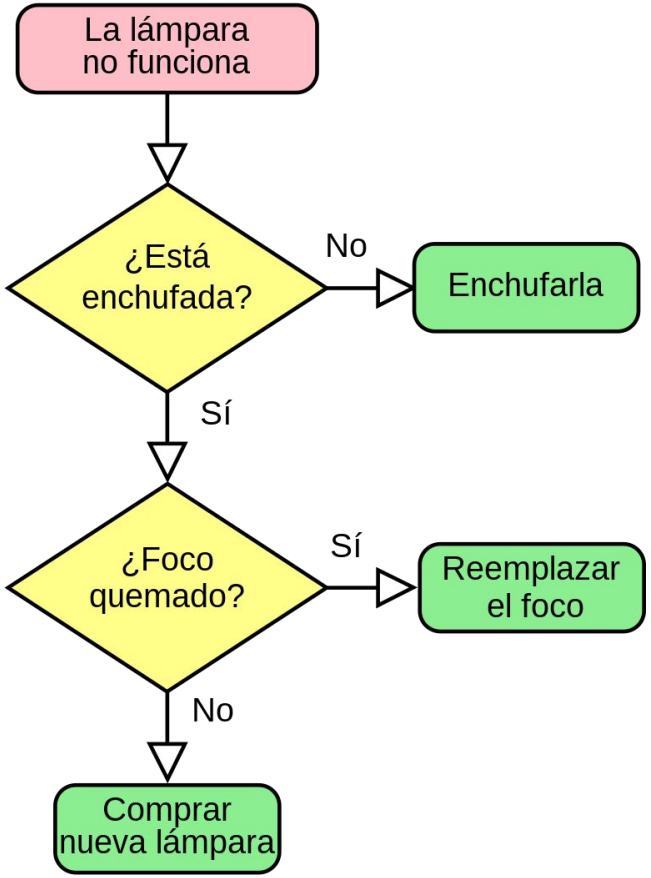
==	igualdad
!=	desigualdad
>	mayor
<	menor
>=	mayor o igual
<=	menor o igual
===	estrictamente iguales
!==	no estrictamente iguales

```
var numero1 = 10;
var numero2 = 5;

// Operador de igualdad (==)
console.log(numero1 == numero2);  // Resultado: false

// Operador de desigualdad (!=)
console.log(numero1 != numero2);  // Resultado: true

// Operador estrictamente igual (===)
console.log(numero1 === numero2);  // Resultado: false

// Operador estrictamente desigual (!==)
console.log(numero1 !== numero2);  // Resultado: true

// Operador mayor que (>)
console.log(numero1 > numero2);  // Resultado: true

// Operador menor que (<)
console.log(numero1 < numero2);  // Resultado: false

// Operador mayor o igual que (>=)
console.log(numero1 >= numero2);  // Resultado: true

// Operador menor o igual que (<=)
console.log(numero1 <= numero2);  // Resultado: false
```

Sentencias condicionales: if / if..else

```
if (condicion) {
    código que se ejecuta si la condición es verdadera
}
```

```
if (condicion) {
    código que se ejecuta si la condición es verdadera
} else {
    código que se ejecuta si la condición es falsa
}
```

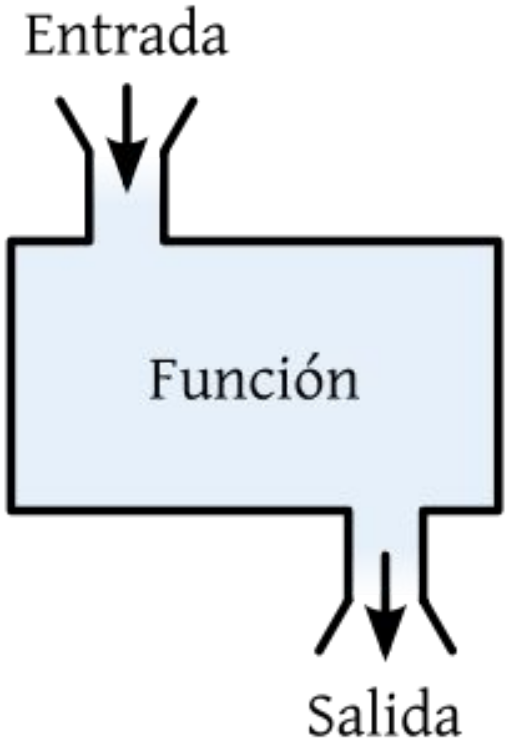


Javascript: Funciones

Una función es un bloque de código reutilizable que realiza una tarea específica cuando se invoca o se llama. Una función puede recibir datos de entrada, llamados argumentos o parámetros, realizar operaciones en esos datos y devolver un resultado.

```
function nombre(par1, par2, par3) {
    /* código */
}
```

```
function sumar(numero1,numero2){
    return numero1+numero2;
}
```



Javascript: Funciones

- Para invocar una función se utiliza el operador ()

```
let resultadoSuma = sumar(2,3);
```

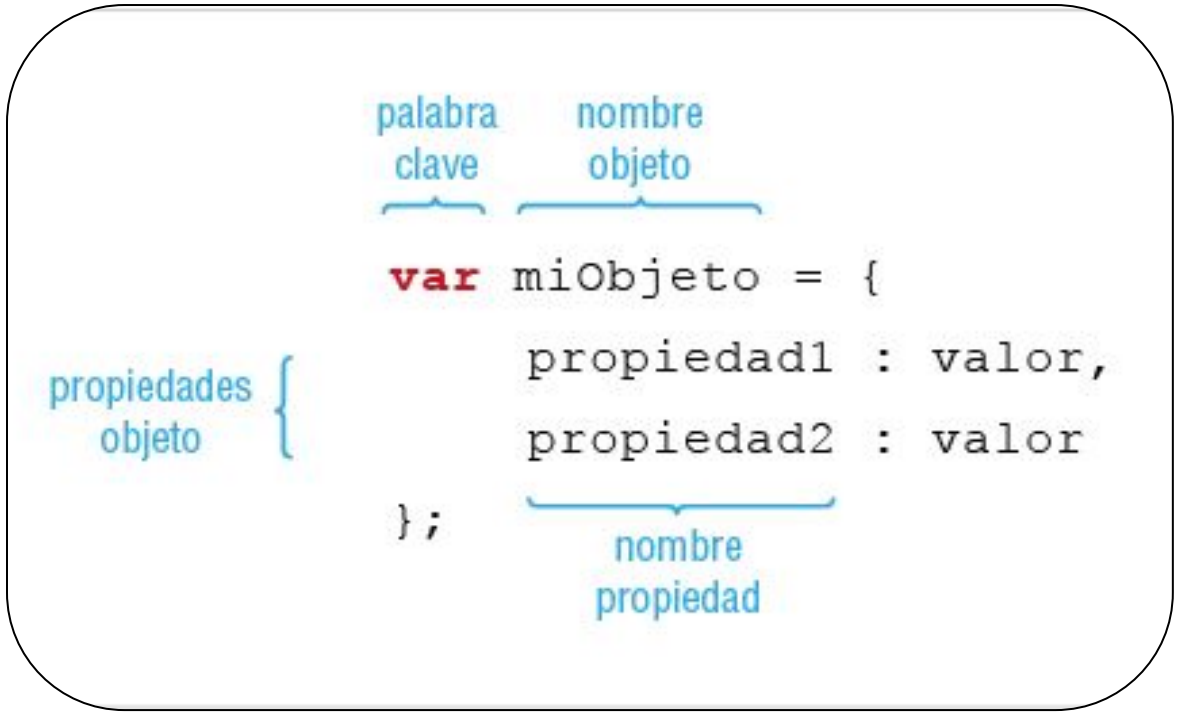
- Se puede almacenar una función en una variable sin invocarla.

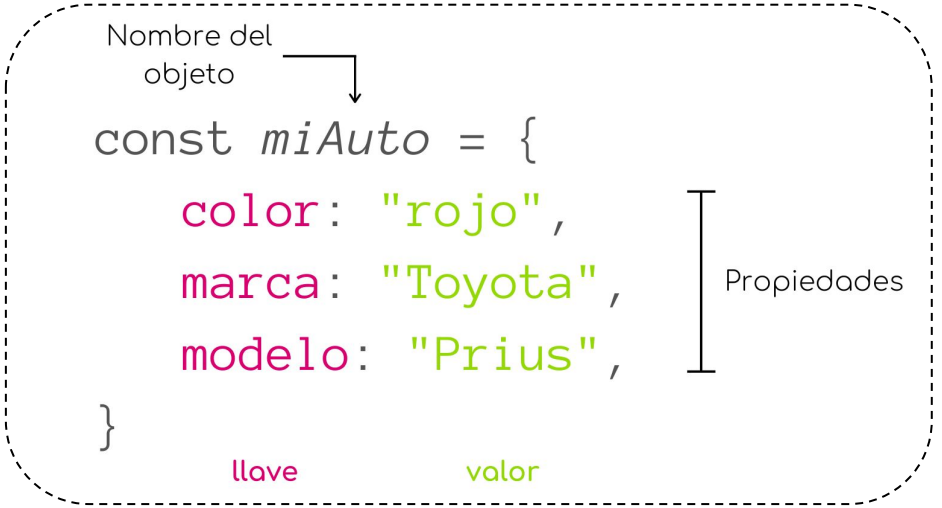
```
let c = sumar;
d = c(5,6);  /* d contendrá 11 */
```

- Funciones flecha (arrow function)

```
sumar = (a, b) => { return a + b; }

sumar = (a, b) => a + b;
```

Javascript: Objetos





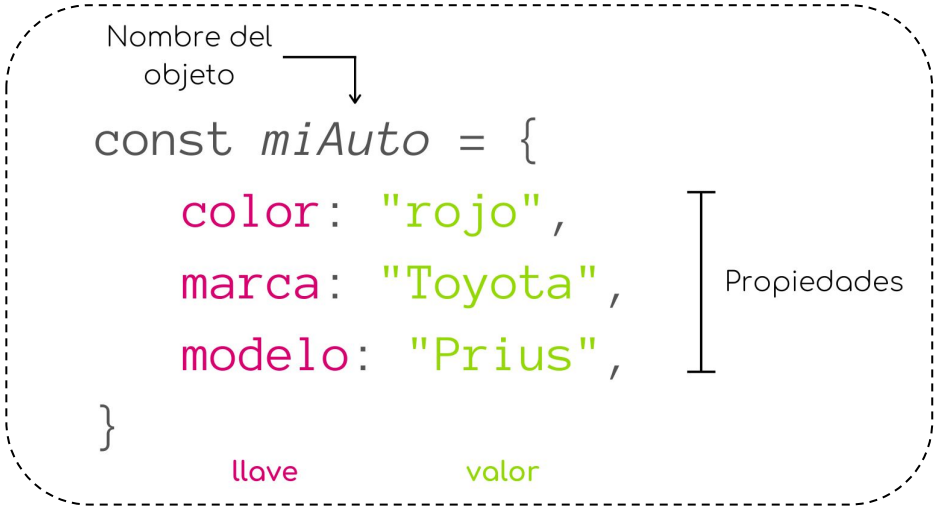
Javascript: Objetos

Se puede acceder a las propiedades mediante la **notación punto:**

```
let m = miAuto.marca;
auto.marca = "Fiat";
```

O **usando corchetes** [ ]

```
let m = miAuto["marca"];
auto["marca"] = "Fiat";
```



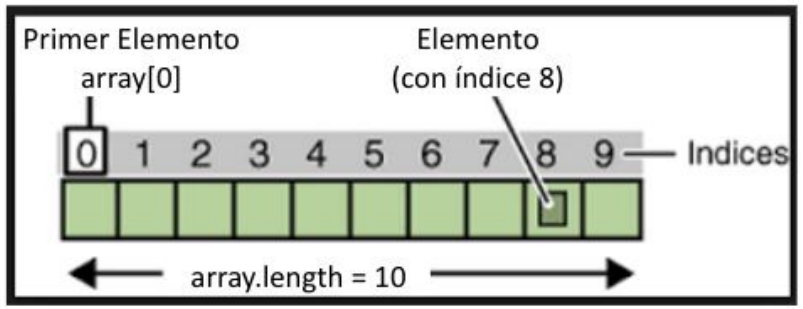
Javascript: Arreglos

Sintaxis:

```
let nombre-array = [item1, item2, ...];
```

Ejemplo:

```
let colores = ["Verde","Rojo","Azul"];
```



Acceder a los elementos de un array:

```
let color1 = colores[0];
colores[2] = "Amarillo";
```

Los **arreglos**, también conocidos como arrays, son estructuras de datos que permiten almacenar y organizar múltiples valores en una sola variable. Los arreglos son colecciones ordenadas, lo que significa que los elementos se almacenan en una secuencia y se accede a ellos mediante un índice numérico.

Sentencias iterativas: for

```
for (sentencia1; sentencia2; sentencia3) {
    código
    //break;
    //continue;
    }
```

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

Sentencias iterativas: for in

```
/* Recorre las propiedades de un objeto */

for (let propiedad in Persona) {
  console.log(Persona[propiedad]);
}
```

Javascript: JSON

JSON (se pronuncia Yeison) viene de **J**ava**S**cript **O**bject **N**otation

- Es un formato de intercambio de datos entre aplicaciones independiente del lenguaje
- Es auto-descripto


```
{
    "clientes":[
                {"nombre":"Juan", "apellido":"Perez"},
                {"nombre":"Pedro", "apellido":"Garcia"},
                {"nombre":"Maria", "apellido":"Sanchez"}
              ]
}
```

Bueno, Vamo a Codea!!!!

- Utilizando la herramienta javascript-tester, ingresando al siguiente sitio:
  https://codebeautify.org/javascript-tester

- Crear un programa en JavaScript que permita realizar un registro de estudiantes. El programa debe tener las siguientes funcionalidades:
  1. Permitir al usuario ingresar la cantidad de estudiantes que desea registrar.
  2. Solicitar al usuario que ingrese los nombres y edades de los estudiantes.
  3. Almacenar la información de cada estudiante en un objeto con las propiedades nombre y edad.
  4. Guardar cada objeto del estudiante en un array.
  5. Mostrar en pantalla la lista de estudiantes registrados, mostrando el nombre y la edad de cada uno.

Actividad 1: Javascript

- Utilizando la herramienta javascript-tester, ingresando al siguiente sitio: https://codebeautify.org/javascript-tester
- Escribe una función llamada calcularPromedio que tome como parámetro un arreglo de números y calcule el promedio de esos números. La función debe retornar el promedio calculado:
- Instrucciones:
  1. Crea una función llamada calcularPromedio con un parámetro llamado **numeros**.
  2. Dentro de la función, declara una variable llamada suma y asígnale el valor inicial de 0.
  3. Utiliza la sentencia iterativa **for** para recorrer cada número en el arreglo **numeros**.
  4. En cada iteración, suma el número actual al valor de suma.
  5. Después de recorrer todos los números, calcula el promedio dividiendo suma entre la longitud del arreglo numeros.
  6. Retorna el promedio calculado.
  7. Entrega el codigo fuente resultante en la actividad del aula virtual.

```
1   const calcularPromedio = (numeros) => {
2     let suma = 0;
3
4     for (let i = 0; i < numeros.length; i++) {
5       suma += numeros[i];
6     }
7
8     let promedio = suma / numeros.length;
9     return promedio;
10  }
11
12  // Prueba de la función
13  let numeros = [5, 8, 2, 10, 4];
14  let resultado = calcularPromedio(numeros);
15  console.log(resultado);
```

MUCHAS GRACIAS

ANDÉN Centro de Innovación y Emprendimientos Tecnológicos

SECRETARÍA DE EXTENSIÓN UNIVERSITARIA UTN - FRC SEU

UTN Facultad Regional Córdoba

Agencia CÓRDOBA JOVEN

CÓRDOBA entre todos