

# DISEÑO DE SITIOS WEB DINÁMICOS

## FUNDAMENTOS DE JAVASCRIPT



### TEMA 01:

JavaScript: Entorno de trabajo, sintaxis básica, variables, constantes, tipos de datos primitivos.

### DOCENTE:

MIRIAM HUACCHA HERRERA

2025-II

### PROGRAMA DE ESTUDIOS:

II DESARROLLO DE SISTEMAS DE INFORMACIÓN

# JavaScript

Es un lenguaje de programación que se usa principalmente para hacer páginas web interactivas.

Se ejecuta en el navegador (como Chrome o Firefox), aunque también se puede usar en servidores.



# JavaScript (JS)

Es un lenguaje de programación interpretado, de alto nivel y multiparadigma, utilizado principalmente para dar interactividad y dinamismo a las páginas web.

- **Cliente y servidor:** Funciona tanto en el navegador (frontend) como en el servidor (Node.js).
- **Interpretado:** No requiere compilación, se ejecuta directamente en el navegador.
- **Multiparadigma:** Soporta programación orientada a objetos, funcional e imperativa.
- **Tipado dinámico:** No es necesario declarar el tipo de dato de las variables.

# ENTORNO DE TRABAJO

## a) Instalación de Visual Studio Code

- 1.Descarga **VS Code**: <https://code.visualstudio.com/>.
- 2.Instálalo y abre el programa.
- 3.Crea una **carpeta de proyecto** (por ejemplo: proyectoJS).
- 4.Crea un archivo **index.html** y un archivo **app.js**.



## b) Extensiones recomendadas

- Live Server**: Permite ejecutar el proyecto en tiempo real en el navegador.
- ESLint**: Ayuda a encontrar errores de sintaxis

html

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Mi Primer Proyecto JS</title>
</head>
<body>
  <h1>Hola JavaScript</h1>
  <script src="app.js"></script>
</body>
</html>
```

```
var elem = document.getElementById('animate')
var pos = 0;
var id = setInterval(frame, 10);
function frame() {
  if (pos == 350) {
    clearInterval(id);
  } else {
    pos++;
    elem.style.top = pos + "px";
    elem.style.left = pos + "px";
  }
}
```

## ¿Qué es una variable?

Es un contenedor para almacenar información que puede cambiar durante la ejecución del programa.

Palabra clave	Características
var	Antiguo. Tiene alcance global o de función. Evitar en proyectos modernos.
let	Recomendado. Tiene alcance de bloque ( <code>{ }</code> ). Permite reasignar valores.
const	Recomendado para constantes. No permite reasignación.

# Declarar variables

Desde ES6 (versión moderna de JavaScript)

Tenemos 3 formas principales de declarar variables:

javascript

```
var nombre = "Juan"; // Primera declaración  
// Redeclarando la variable "nombre"  
var nombre = "Pedro";  
console.log(nombre); // Imprimirá: Pedro
```

`var nombre = "Ana";` // Evitar

`let edad = 25;` // Se puede cambiar

`const país = "Perú";` // No se puede cambiar

`edad = 26;` // 

`país = "Chile";` //  Error

## var

```
var nombre = "Ana";
```

- Antiguo (no recomendado)
- Tiene **scope global o de función**
- Puede redeclararse y reasignarse

```
let nombre = "Carlos";  
const nacimiento = 1990;
```

```
nombre = "Juan"; //  se puede cambiar
```

```
// nacimiento = 2000;  Error: no se puede reasignar una constante
```

```
console.log(nombre);      // Juan  
console.log(nacimiento);  // 1990
```



# let

```
let edad = 25;
```

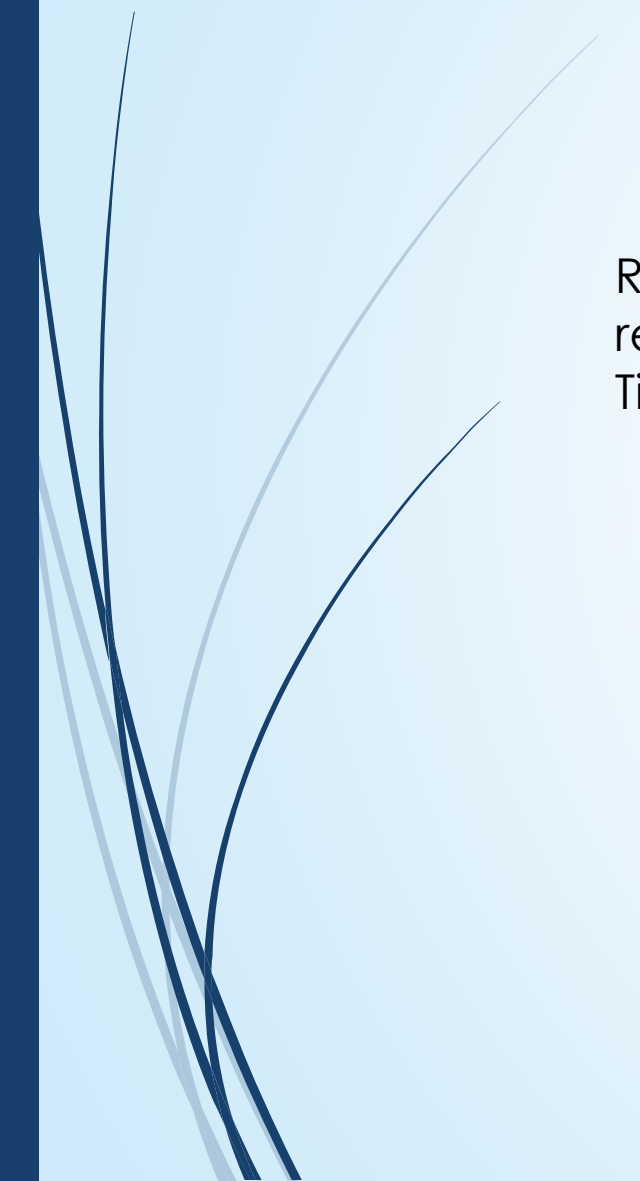
- Recomendado para variables que pueden cambiar
- Tiene scope de bloque (más seguro)
- Puede reasignarse, pero no redeclararse en el mismo scope



## const

```
const PI = 3.1416;
```

Recomendado para valores que no cambian. No puede ser reasignado  
Tiene scope de bloque



# TIPOS DE DATOS PRIMITIVOS

Tipo	Ejemplo	Descripción
String	<code>"Hola"</code> , <code>'Mundo'</code>	Cadenas de texto
Number	<code>42</code> , <code>3.14</code>	Números enteros o decimales
Boolean	<code>true</code> , <code>false</code>	Valores lógicos
Undefined	<code>let x;</code>	Variable declarada sin valor
Null	<code>let x = null;</code>	Valor intencionalmente vacío
Symbol	<code>Symbol("id")</code>	Valor único e inmutable
BigInt	<code>1234567890123456789n</code>	Números muy grandes