

# UNIVERSIDAD DEL VALLE



## PROYECTO FINAL PROGRAMACIÓN WEB I: MANUAL TÉCNICO

**ESTUDIANTE : IVO ISMAEL AVENDAÑO IBAÑEZ**

**CARRERA : ING. EN SISTEMAS INFORMATICOS**

**DOCENTE : ING. HENRY MIRANDA ORDOÑEZ**

**MATERIA : PROGRAMACIÓN WEB I**

**2025**

## HTML

### 1. span

Es un contenedor en línea (inline) usado para encerrar palabras o partes específicas de un texto con el fin de aplicar estilos sin romper el flujo del contenido.

No genera saltos de línea.

(línea 42 en HTML)

## CSS

### 2. z-index

Propiedad que define qué elemento está por encima de otro en el eje “vertical” imaginario de la pantalla.

Solo funciona en elementos que tengan un position distinto de static (relative, absolute, fixed, sticky).

Ejemplo:

Un z-index: 999; aparece sobre otro con z-index: 1;.

(línea 17 en CSS)

### 3. backdrop-filter

Aplica efectos gráficos al fondo que está detrás de un elemento, como desenfoque o brillo.

Ejemplo típico:

backdrop-filter: blur(10px); → Produce un vidrio esmerilado.

(línea 20 en CSS)

### 4. display

Define cómo se comporta un elemento en el flujo del documento.

Los valores más comunes:

block: ocupa toda la línea.

inline: no rompe la línea.

inline-block: inline con capacidad de tamaño.

flex: convierte al elemento en contenedor flexible.

grid: contenedor en cuadrícula.

none: oculta el elemento eliminándolo del flujo.

(línea 53 en CSS)

## **5. rem**

Unidad relativa basada en el tamaño de fuente del elemento raíz (html).

Si `html { font-size: 16px }`, entonces:

1rem = 16px

2rem = 32px

## **6. overflow**

Define cómo se comporta el contenido que se sale del tamaño del elemento.

Valores comunes:

visible: deja que se desborde.

hidden: corta lo que sobresale.

scroll: agrega scroll siempre.

auto: agrega scroll solo cuando hace falta.

(línea 103 en CSS)

## **7. pointer-events**

Controla si un elemento puede recibir clics o interacciones del mouse.

none: no recibe clicks ni hover.

auto: comportamiento normal.

all: cualquier click.

(línea 120 en CSS)

## **8. gap**

Propiedad que define el espacio entre los elementos internos cuando usas `display: flex` o

`display: grid`.

Reemplaza la necesidad de usar márgenes entre items.

Ejemplo:

gap: 20px;

(línea 137 en CSS)

## **9. text-decoration**

Controla decoraciones aplicadas al texto:

underline

line-through

overline

none

(línea 160 en CSS)

## **10. space-between**

Valor de justify-content. Distribuye los elementos de forma que queden separados al máximo, dejando el espacio sobrante repartido entre ellos.

Primer elemento pegado a la izquierda, último a la derecha.

(línea 173 en CSS)

## **11. flex-direction**

Define la dirección principal en la que se colocan los elementos dentro de un contenedor flex.

Valores:

row: horizontal (por defecto).

column: vertical.

row-reverse

column-reverse

(línea 203 en CSS)

## JavaScript

### 12. **window.scrollY**

Propiedad que indica cuántos píxeles ha sido desplazada la ventana hacia abajo.

Devuelve un número entero.

Ejemplo:

```
if (window.scrollY > 100) { }
```

(línea 3 en JavaScript)

### 13. **element.classList.add('clase')**

Añade una clase CSS a un elemento.

No elimina otras clases existentes, solo suma.

(línea 6 en JavaScript)

### 14. **element.classList.remove('clase')**

Elimina una clase del elemento si la tiene.

(línea 8 en JavaScript)

### 15. **element.classList.toggle('clase')**

Si el elemento tiene la clase, la quita.

Si no la tiene, la agrega.

(línea 15 en JavaScript)

### 16. **function(e)**

Forma abreviada de declarar una función que recibe un parámetro llamado e usado para eventos.

e suele representar el objeto del evento, que contiene información como la tecla presionada o el elemento clickeado.

(línea 23 en JavaScript)

### 17. **e.stopPropagation()**

Evita que el evento siga propagándose hacia elementos padres cuando un elemento debe responder al click sin activar el click del contenedor.

(línea 24 en JavaScript)

## **18. document.createElement(' ')**

Crea un nuevo elemento HTML desde JavaScript.

El elemento existe en memoria y debe insertarse en el DOM manualmente.

(línea 60 en JavaScript)

## **19. document.getElementsByTagName(' ')**

Devuelve una colección (HTMLCollection) de todas las etiquetas con el nombre especificado.

Ejemplo:

```
var s = d.getElementsByTagName(t)[0];
```

(línea 61 en JavaScript)

## **20. onload**

Evento que se ejecuta cuando un recurso termina de cargar:

La página entera

Una imagen

Un iframe

Un script

Ejemplo:

```
window.onload = function() { ... }
```

(línea 66 en JavaScript)

## **21. load**

Se usa como evento:

```
element.addEventListener('load', function(){...})
```

Generalmente aplicado a imágenes o iframes.

(línea 67 en JavaScript)

## **22. setTimeout(función, tiempo)**

Ejecuta un bloque de código una sola vez después de un tiempo determinado en milisegundos.

Ejemplo:

```
setTimeout(() => console.log('Hola'), 2000);
```

(línea 76 en JavaScript)

## **23. parentNode.insertBefore(nuevo, referencia)**

Inserta un nodo antes de otro nodo dentro del mismo padre.

Ejemplo:

```
padre.insertBefore(nuevoElemento, elementoExistente);
```

Si se usa referencia = null, lo agrega al final.

(línea 83 en JavaScript)

## **24. scroll**

Se dispara cada vez que el usuario desplaza la página o un contenedor.

Ejemplo:

```
window.addEventListener('scroll', () => {  
    console.log("scrolleando");  
});
```

(línea 1 en JavaScript)