# INDICE:

Yoselyn y Jakelin

1 INTRODUCCIÓN VENTAJAS DE USAR JAVASCRIPT EN LA ELABORACIÓN DE UNA . PÁGINA WEB

2 PRACTICA 1. AGREGAR TEXTO, AL HACER CLIC A LA IMAGEN

3 PRACTICA 2. CREAR DIVISIONES CON LA ETIQUETA DIV

4 PRÁCTICA 3. COLOCAR UN LOGO EN EL DIV DE ENCABEZADO DE LA PÁGINA

5 PRÁCTICA 4. INSERTAR IMÁGENES EN UN DIV DE MANERA ORDENADA

6 PRÁCTICA 5. CREAR GALERIA DE IMÁGENES

7 PRÁCTICA 6. CREAR UNA TABLA CON HTML Y CSS

8 PRÁCTICA 7. INCRUSTAR VIDEO PARA UNA WEB

9 PRÁCTICA 8. CREAR UN CARRUSEL DE IMÁGENES AUTOMÁTICO

10 PRÁCTICA 9. CREAR UN ENLACE DE UNA ARCHIVO DESCARGA

11 PRÁCTICA 10. CREAR UN EFECTO A UN ICONO DE LA PÁGINA

12 PRÁCTICA 11. PROCEDIMIENTO PARA HOSTING DE LA PÁGINA (GITHUB/NETLIFY)

**1 AGREGAR TEXTO, AL HACER CLIC A LA IMAGEN**

### Paso 1: Estructura HTML básica

En el archivo HTML (index.html), vamos a definir la estructura básica con una imagen y un área para mostrar texto dinámico al hacer clic en la imagen.

html

Copiar código

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Texto al hacer clic en la imagen</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<div class="image-container">

<img src="img/image1.jpg" alt="Imagen 1" id="imagen1">

<div id="texto" class="hidden">Texto dinámico que aparece al hacer clic en la imagen 1.</div>

</div>

<script src="script.js"></script>

</body>

</html>

### Paso 2: Estilizar con CSS

En el archivo CSS (styles.css), vamos a aplicar estilos para ocultar inicialmente el texto y dar formato a la imagen y al texto dinámico.

css

Copiar código

/\* styles.css \*/

body {

font-family: Arial, sans-serif;

background-color: #f0f0f0;

margin: 0;

padding: 20px;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

}

.image-container {

position: relative;

}

img {

max-width: 100%;

height: auto;

cursor: pointer;

}

#texto {

position: absolute;

top: 10px;

left: 10px;

background-color: rgba(255, 255, 255, 0.8);

padding: 10px;

border: 1px solid #ccc;

display: none;

}

.hidden {

display: none;

}

.visible {

display: block;

}

### Paso 3: Agregar interactividad con JavaScript

En el archivo JavaScript (script.js), vamos a añadir funcionalidad para mostrar y ocultar el texto al hacer clic en la imagen.

javascript

Copiar código

// script.js

const imagen1 = document.getElementById('imagen1');

const texto = document.getElementById('texto');

imagen1.addEventListener('click', () => {

texto.classList.toggle('visible'); // Alterna la clase 'visible' para mostrar u ocultar el texto

});

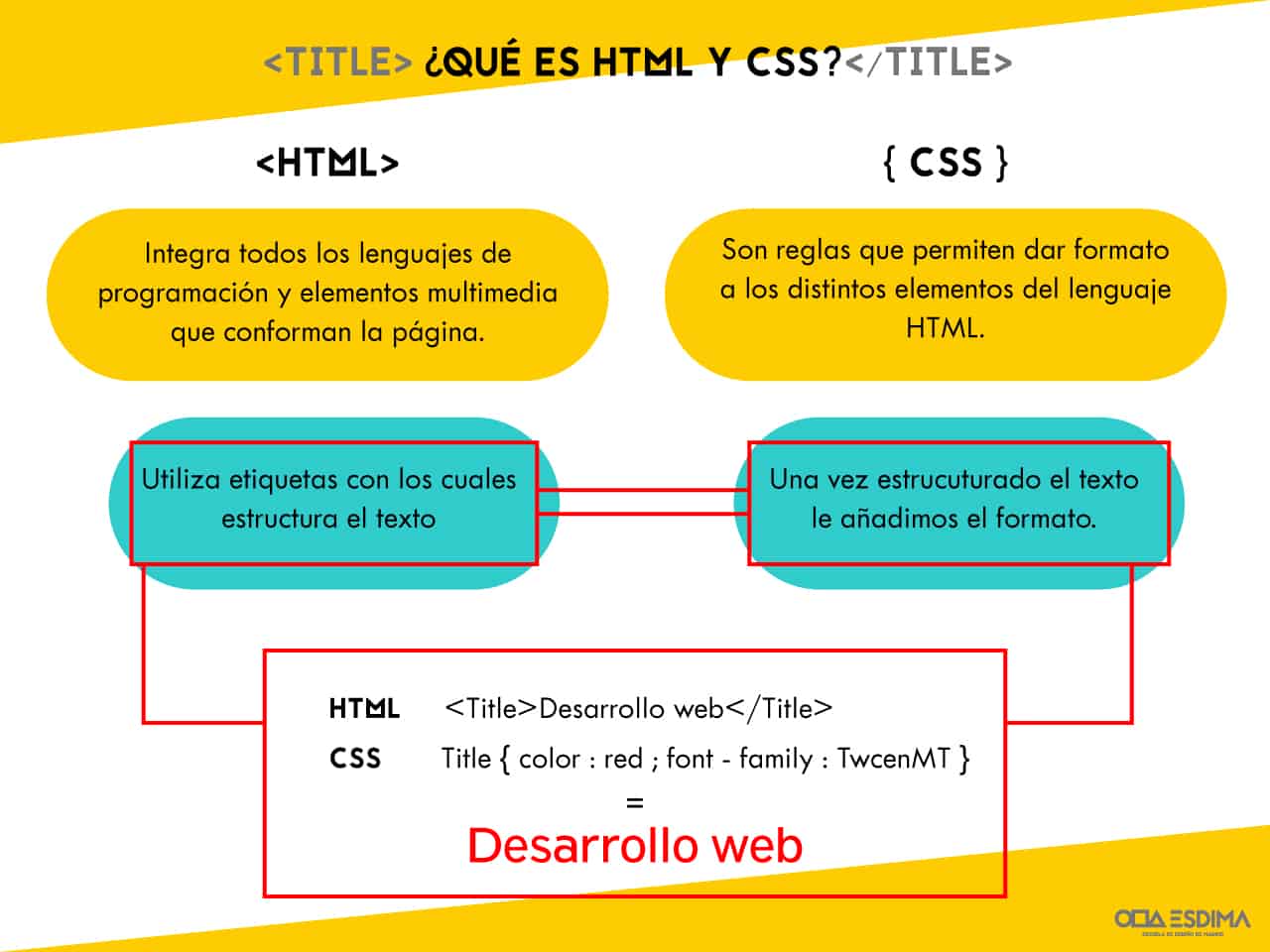
### Explicación del JavaScript:

* **imagen1**: Selecciona la imagen por su ID.
* **Evento click**: Asocia un evento click a la imagen para mostrar u ocultar el texto (#texto).
* **texto.classList.toggle('visible')**: Alterna la clase visible para mostrar u ocultar el texto al hacer clic en la imagen.

### Notas adicionales:

* **Texto dinámico**: Puedes modificar el texto dentro del elemento <div id="texto"> según tus necesidades.
* **Estilos y ajustes**: Personaliza los estilos CSS en styles.css para que se adapten al diseño de tu página web.
* **Imágenes**: Asegúrate de tener una imagen (image1.jpg) en una carpeta img dentro del mismo directorio que tu archivo HTML.

Con estos pasos, has creado una práctica donde al hacer clic en una imagen, se muestra texto dinámico. Esto mejora la experiencia del usuario al proporcionar información adicional o descripciones relacionadas con la imagen mostrada.



**2 CREAR DIVISIONES CON LA ETIQUETA DIV**

### Paso 1: Estructura HTML básica

En el archivo HTML (index.html), vamos a definir varias divisiones utilizando la etiqueta <div> para crear secciones de contenido.

html

Copiar código

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Divisions con la etiqueta &lt;div&gt;</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<div class="container">

<div class="header">

<h1>Encabezado de la página</h1>

</div>

<div class="main-content">

<div class="sidebar">

<h2>Barra lateral</h2>

<ul>

<li>Enlace 1</li>

<li>Enlace 2</li>

<li>Enlace 3</li>

</ul>

</div>

<div class="content">

<h2>Contenido principal</h2>

<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed vitae risus nec nisl laoreet consectetur.</p>

</div>

</div>

<div class="footer">

<p>Pie de página - © 2024 Mi Sitio Web</p>

</div>

</div>

<script src="script.js"></script>

</body>

</html>

### Paso 2: Estilizar con CSS

En el archivo CSS (styles.css), vamos a aplicar estilos para dar formato a las divisiones (<div>) y sus contenidos.

css

Copiar código

/\* styles.css \*/

body {

font-family: Arial, sans-serif;

background-color: #f0f0f0;

margin: 0;

padding: 20px;

}

.container {

max-width: 800px;

margin: 20px auto;

background-color: #fff;

border: 1px solid #ccc;

border-radius: 5px;

padding: 10px;

}

.header {

background-color: #333;

color: #fff;

padding: 10px;

text-align: center;

}

.main-content {

display: flex;

justify-content: space-between;

}

.sidebar {

flex: 1;

padding: 10px;

background-color: #f2f2f2;

}

.content {

flex: 3;

padding: 10px;

}

.footer {

background-color: #333;

color: #fff;

padding: 10px;

text-align: center;

}

### Paso 3: Opcional - Agregar interactividad con JavaScript

En el archivo JavaScript (script.js), podemos agregar interactividad opcional, como cambiar contenido dinámicamente o manejar eventos.

javascript

Copiar código

// script.js

// Aquí puedes agregar tu código JavaScript si deseas añadir interactividad

// por ejemplo, manipulando el contenido de las divisiones.

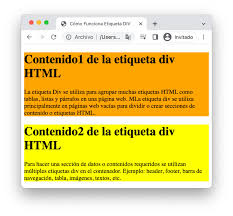
### Explicación del código:

* **HTML**: Utilizamos la etiqueta <div> para crear divisiones (container, header, main-content, sidebar, content, footer) que ayudan a organizar el contenido de la página.
* **CSS**: Los estilos aplicados en styles.css definen la apariencia visual de las divisiones y sus contenidos, utilizando colores de fondo, bordes, márgenes y espaciado para crear una estructura visualmente atractiva y organizada.
* **JavaScript (opcional)**: Si se necesita interactividad adicional, como manipulación dinámica del contenido o manejo de eventos, se puede implementar en script.js.

Con estos pasos, has aprendido a crear divisiones con la etiqueta <div> en HTML para estructurar el contenido de tu página web, aplicar estilos con CSS para mejorar su apariencia y, opcionalmente, utilizar JavaScript para agregar funcionalidades interactivas según tus necesidades.

Principio del formulario

Final del formulario



**3 COLOCAR UN LOGO EN EL DIV DE ENCABEZADO DE LA PÁGINA**

### 1. Creación del HTML

Primero, vamos a configurar la estructura básica del HTML, que incluirá un div de encabezado (header) donde colocaremos el logo.

html

Copiar código

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Colocar Logo en el Encabezado</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<header class="encabezado">

<div class="logo">

<img src="logo.png" alt="Logo de la empresa">

</div>

<nav>

<ul>

<li><a href="#">Inicio</a></li>

<li><a href="#">Productos</a></li>

<li><a href="#">Servicios</a></li>

<li><a href="#">Contacto</a></li>

</ul>

</nav>

</header>

<section class="contenido">

<!-- Aquí va el contenido principal de la página -->

</section>

<footer>

<!-- Aquí va el pie de página -->

</footer>

<script src="script.js"></script>

</body>

</html>

* **Explicación del HTML:**
  + Se utiliza un <header> con la clase encabezado para definir la sección de encabezado de la página.
  + Dentro del <header>, se agrega un <div> con la clase logo donde se incluye la etiqueta <img> para mostrar el logo de la empresa. Asegúrate de ajustar el atributo src con la ruta correcta a tu imagen de logo (logo.png en este ejemplo).
  + También se incluye un menú de navegación <nav> con una lista desordenada <ul> y elementos de lista <li> que contienen enlaces <a> a diferentes secciones de la página.

### 2. Estilizado con CSS

A continuación, utiliza CSS para estilizar el encabezado y el logo:

css

Copiar código

/\* styles.css \*/

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

}

.encabezado {

background-color: #333;

color: #fff;

padding: 10px 20px;

display: flex;

align-items: center;

justify-content: space-between;

}

.logo img {

max-width: 150px;

height: auto;

}

nav ul {

list-style-type: none;

margin: 0;

padding: 0;

}

nav ul li {

display: inline-block;

margin-left: 20px;

}

nav ul li a {

color: #fff;

text-decoration: none;

font-weight: bold;

}

.contenido {

padding: 20px;

}

footer {

background-color: #333;

color: #fff;

text-align: center;

padding: 10px 0;

}

* **Explicación del CSS:**
  + Se establecen estilos básicos para el cuerpo (body) de la página para evitar márgenes predeterminados y usar una fuente específica.
  + .encabezado define el estilo del encabezado, incluyendo el color de fondo, color del texto, relleno y utiliza flexbox para alinear elementos y distribuir el espacio.
  + .logo img establece un ancho máximo para el logo y asegura que mantenga su relación de aspecto.
  + nav ul y sus descendientes estilizan el menú de navegación y los enlaces.

### 3. Funcionamiento y Ajustes Adicionales

No es necesario utilizar JavaScript para este ejemplo específico, ya que el comportamiento estático del logo en el encabezado no requiere interactividad. Sin embargo, si deseas añadir interactividad o algún efecto específico, puedes extender la funcionalidad con JavaScript según sea necesario.

### Conclusiones

Este tutorial te proporciona una guía paso a paso para colocar un logo en el div de encabezado de una página web utilizando HTML para la estructura, CSS para el diseño y opcionalmente JavaScript para la interactividad. Asegúrate de personalizar los nombres de clases, rutas de imágenes y estilos según las necesidades y el diseño de tu sitio



**4 INSERTAR IMÁGENES EN UN DIV DE MANERA ORDENADA**

### Paso 1: Estructura HTML

html

Copiar código

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Insertar Imágenes en un Div</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<div id="contenedor-imagenes">

<!-- Aquí se insertarán las imágenes de manera dinámica -->

</div>

<script src="scripts.js"></script>

</body>

</html>

### Paso 2: Estilo CSS (styles.css)

css

Copiar código

#contenedor-imagenes {

display: flex;

flex-wrap: wrap;

gap: 10px; /\* Espacio entre imágenes \*/

}

#contenedor-imagenes img {

width: 200px; /\* Tamaño de las imágenes \*/

height: auto; /\* Altura automática para mantener la proporción \*/

object-fit: cover; /\* Ajuste de la imagen dentro del contenedor \*/

}

### Paso 3: Manipulación con JavaScript (scripts.js)

En este ejemplo, vamos a insertar imágenes de forma estática en el HTML utilizando JavaScript para agregarlas dinámicamente al <div> con id contenedor-imagenes.

javascript

Copiar código

document.addEventListener("DOMContentLoaded", function() {

const contenedorImagenes = document.getElementById("contenedor-imagenes");

// Array de URLs de imágenes (puedes cambiar las URLs por las de tus propias imágenes)

const imagenes = [

"imagen1.jpg",

"imagen2.jpg",

"imagen3.jpg",

"imagen4.jpg"

];

// Iterar sobre el array de imágenes y crear elementos <img> para cada una

imagenes.forEach(url => {

const imagen = document.createElement("img");

imagen.src = url;

contenedorImagenes.appendChild(imagen); // Agregar la imagen al contenedor

});

});

### Explicación:

1. **HTML (index.html)**: Se define un contenedor (<div id="contenedor-imagenes">) donde se insertarán las imágenes dinámicamente mediante JavaScript.
2. **CSS (styles.css)**: Se utiliza Flexbox para asegurar que las imágenes se coloquen de manera ordenada y responsiva dentro del contenedor. gap establece el espacio entre las imágenes y se ajusta el tamaño de las imágenes según sea necesario.
3. **JavaScript (scripts.js)**: Se utiliza document.addEventListener("DOMContentLoaded", function() {...}) para asegurarse de que el JavaScript se ejecute después de que el DOM esté completamente cargado. Dentro de esta función:
   * Se obtiene el contenedor de imágenes (contenedor-imagenes) por su ID.
   * Se define un array imagenes que contiene las URLs de las imágenes que deseas mostrar.
   * Se utiliza un bucle forEach para recorrer el array de URLs.
   * Para cada URL, se crea un elemento <img>, se establece su atributo src con la URL correspondiente y se agrega al contenedor contenedor-imagenes.

Este ejemplo muestra cómo insertar imágenes en un <div> de manera ordenada utilizando HTML, CSS y JavaScript de manera conjunta para una experiencia dinámica y responsive.

Principio del formulario

Final del formulario



**5 CREAR GALERIA DE IMÁGENES**

### Paso 1: Estructura HTML básica

En el archivo HTML (index.html), vamos a definir la estructura básica de la galería de imágenes.

html

Copiar código

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Galería de Imágenes</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<div class="gallery">

<div class="thumbnails">

<img src="img/image1-thumbnail.jpg" alt="Imagen 1">

<img src="img/image2-thumbnail.jpg" alt="Imagen 2">

<img src="img/image3-thumbnail.jpg" alt="Imagen 3">

<!-- Agrega más miniaturas según sea necesario -->

</div>

<div class="preview">

<img src="img/image1.jpg" alt="Imagen 1">

</div>

</div>

<script src="script.js"></script>

</body>

</html>

### Paso 2: Estilizar con CSS

En el archivo CSS (styles.css), vamos a aplicar estilos para dar formato a las miniaturas y a la imagen de vista previa.

css

Copiar código

/\* styles.css \*/

body {

font-family: Arial, sans-serif;

background-color: #f0f0f0;

margin: 0;

padding: 20px;

display: flex;

justify-content: center;

}

.gallery {

max-width: 800px;

margin-top: 20px;

}

.thumbnails {

display: flex;

justify-content: space-between;

margin-bottom: 10px;

}

.thumbnails img {

width: 100px;

height: auto;

cursor: pointer;

transition: transform 0.2s ease;

}

.thumbnails img:hover {

transform: scale(1.1);

}

.preview img {

width: 100%;

height: auto;

border: 2px solid #333;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

### Paso 3: Agregar interactividad con JavaScript

En el archivo JavaScript (script.js), vamos a añadir funcionalidad para cambiar la imagen principal al hacer clic en las miniaturas.

javascript

Copiar código

// script.js

const thumbnails = document.querySelectorAll('.thumbnails img');

const preview = document.querySelector('.preview img');

thumbnails.forEach(thumbnail => {

thumbnail.addEventListener('click', () => {

preview.src = thumbnail.src.replace('-thumbnail', ''); // Cambia la imagen principal al hacer clic en miniatura

thumbnails.forEach(t => t.classList.remove('active')); // Quita la clase 'active' de todas las miniaturas

thumbnail.classList.add('active'); // Agrega la clase 'active' a la miniatura clicada

});

});

### Explicación del JavaScript:

* **thumbnails**: Selecciona todas las miniaturas de imágenes.
* **Evento click**: Asocia un evento click a cada miniatura para cambiar la imagen de vista previa (preview) al hacer clic.
* **preview.src**: Actualiza el atributo src de la imagen de vista previa con el de la miniatura clicada, eliminando el sufijo -thumbnail para cargar la imagen de tamaño completo.
* **Clase active**: Añade y remueve la clase active para resaltar visualmente la miniatura seleccionada.

### Notas adicionales:

* **Ruta de Imágenes**: Asegúrate de tener las imágenes (image1-thumbnail.jpg, image1.jpg, etc.) en una carpeta img dentro del mismo directorio que tu archivo HTML.
* **Estilos y Ajustes**: Personaliza los estilos CSS según tus preferencias para la disposición y apariencia de la galería.
* **Responsive**: Si deseas que la galería sea responsive (se adapte a diferentes tamaños de pantalla), considera utilizar media queries en tu archivo CSS.

Con estos pasos, has creado una galería de imágenes básica utilizando HTML para la estructura, CSS para los estilos visuales y JavaScript para la interactividad al hacer clic en las miniaturas.



Principio del formulario

Final del formulario

**6 CREAR UNA TABLA CON HTML Y CSS**

### Paso 1: Crear la estructura HTML de la tabla

En el archivo HTML (index.html), vamos a definir la estructura básica de la tabla.

html

Copiar código

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Tabla con HTML y CSS</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<div class="tabla-container">

<table>

<thead>

<tr>

<th>Nombre</th>

<th>Edad</th>

<th>Ciudad</th>

</tr>

</thead>

<tbody>

<tr>

<td>Juan</td>

<td>30</td>

<td>Madrid</td>

</tr>

<tr>

<td>María</td>

<td>25</td>

<td>Barcelona</td>

</tr>

<tr>

<td>Pablo</td>

<td>28</td>

<td>Valencia</td>

</tr>

</tbody>

</table>

</div>

</body>

</html>

### Paso 2: Estilizar la tabla con CSS

En el archivo CSS (styles.css), vamos a aplicar estilos para mejorar el aspecto visual de la tabla.

css

Copiar código

/\* styles.css \*/

body {

font-family: Arial, sans-serif;

background-color: #f0f0f0;

margin: 0;

padding: 20px;

}

.tabla-container {

max-width: 600px;

margin: 20px auto;

background-color: #fff;

border: 1px solid #ccc;

border-radius: 5px;

overflow: hidden;

}

table {

width: 100%;

border-collapse: collapse;

}

th, td {

border: 1px solid #ccc;

padding: 8px;

text-align: left;

}

thead {

background-color: #f2f2f2;

}

th {

background-color: #ddd;

font-weight: bold;

}

tbody tr:nth-child(even) {

background-color: #f2f2f2;

}

### Explicación de los estilos CSS:

* **.tabla-container**: Define un contenedor para la tabla con márgenes y bordes.
* **table**: Establece el ancho completo y elimina los espacios entre las celdas.
* **th, td**: Define el borde y el relleno para las celdas y encabezados.
* **thead**: Aplica un color de fondo a la fila de encabezados.
* **th**: Establece un fondo diferente y negrita para los encabezados.
* **tbody tr:nth-child(even)**: Alterna el color de fondo para las filas pares de datos.

### Opcional: Añadir interactividad con JavaScript

Si deseas agregar interactividad, como ordenar o filtrar la tabla dinámicamente, puedes usar JavaScript. Aquí hay un ejemplo básico utilizando JavaScript para ordenar la tabla por la columna haciendo clic en el encabezado:

javascript

Copiar código

// script.js (opcional)

const thList = document.querySelectorAll('th');

thList.forEach((th, index) => {

th.addEventListener('click', () => {

sortTable(index);

});

});

function sortTable(colIndex) {

const table = document.querySelector('table');

const tbody = table.querySelector('tbody');

const rows = Array.from(tbody.rows);

rows.sort((rowA, rowB) => {

const cellA = rowA.cells[colIndex].textContent.trim();

const cellB = rowB.cells[colIndex].textContent.trim();

return cellA.localeCompare(cellB, 'es', { numeric: true });

});

while (tbody.firstChild) {

tbody.removeChild(tbody.firstChild);

}

rows.forEach(row => {

tbody.appendChild(row);

});

}

### Explicación del JavaScript:

* **thList**: Selecciona todos los encabezados de columna (th).
* **Evento click**: Asocia un evento click a cada encabezado para llamar a la función sortTable con el índice de columna como parámetro.
* **sortTable**: Ordena las filas de la tabla según el contenido de la columna seleccionada usando localeCompare para ordenar correctamente los números y texto en español.

Estos pasos te permitirán crear una tabla básica con HTML y CSS, y opcionalmente añadir funcionalidades interactivas con JavaScript para mejorar la experiencia del usuario.



**7 INCRUSTAR VIDEO PARA UNA WEB**

### Paso 1: Estructura HTML

html

Copiar código

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Video en una página web</title>

<style>

/\* Estilos CSS opcionales \*/

body {

font-family: Arial, sans-serif;

margin: 20px;

}

.video-container {

max-width: 800px;

margin: 0 auto;

}

video {

width: 100%;

display: block;

}

</style>

</head>

<body>

<div class="video-container">

<video controls>

<source src="video.mp4" type="video/mp4">

<!-- Agrega más fuentes según los formatos de video que desees soportar -->

Tu navegador no soporta el elemento de video.

</video>

</div>

</body>

</html>

### Explicación:

1. **HTML (index.html)**:
   * Se utiliza el elemento <video> de HTML5 para incrustar el video.
   * El atributo controls agrega controles de reproducción estándar (como reproducción, pausa, volumen) al reproductor de video.
   * <source> se usa para especificar los diferentes formatos de video (src="video.mp4"). Puedes añadir más <source> dentro del <video> para proporcionar diferentes formatos de video (por ejemplo, WebM, Ogg) para asegurar la compatibilidad con diferentes navegadores.
2. **CSS** (incluido en el <head> del HTML):
   * Se agregan estilos opcionales para centrar el contenedor del video (video-container) y ajustar el tamaño del video dentro del contenedor (video).

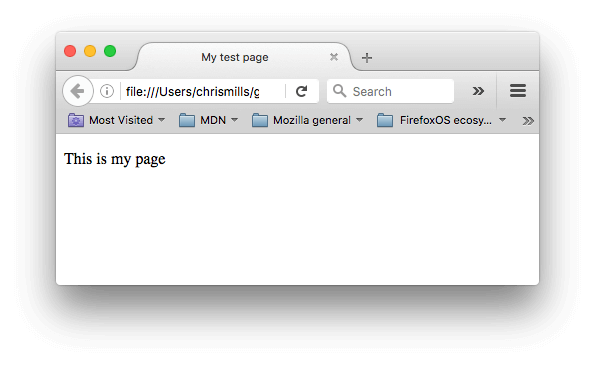
### Notas adicionales:

* Asegúrate de tener diferentes formatos de video (mp4, webm, ogg) para garantizar la compatibilidad con varios navegadores. Esto se logra añadiendo múltiples elementos <source> dentro del elemento <video>.
* Puedes ajustar los estilos CSS según las necesidades de diseño específicas de tu página web.
* JavaScript no es necesario para la funcionalidad básica de incrustar y reproducir videos, ya que HTML5 proporciona soporte nativo para esto.

Este ejemplo proporciona una manera simple de incrustar un video en una página web utilizando HTML5. Puedes expandirlo agregando más funcionalidades como reproducción automática, controles personalizados o eventos de JavaScript para manejar la reproducción del video de forma programática si es necesario.

Principio del formulario

Final del formulario



8 CREAR UN CARRUSEL DE IMÁGENES AUTOMÁTICO

### Paso 1: Estructura HTML

html

Copiar código

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Carrusel de Imágenes Automático</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<div class="carrusel">

<div class="slide">

<img src="imagen1.jpg" alt="Imagen 1">

</div>

<div class="slide">

<img src="imagen2.jpg" alt="Imagen 2">

</div>

<div class="slide">

<img src="imagen3.jpg" alt="Imagen 3">

</div>

<!-- Agrega más imágenes según sea necesario -->

</div>

<script src="scripts.js"></script>

</body>

</html>

### Paso 2: Estilo CSS (styles.css)

css

Copiar código

.carrusel {

overflow: hidden;

width: 100%;

position: relative;

}

.slide {

display: none;

width: 100%;

}

.slide img {

width: 100%;

height: auto;

}

/\* Estilos opcionales para los controles \*/

.controles {

position: absolute;

top: 50%;

transform: translateY(-50%);

width: 100%;

display: flex;

justify-content: space-between;

z-index: 100;

}

.control-izquierdo, .control-derecho {

cursor: pointer;

background-color: rgba(0, 0, 0, 0.5);

color: #fff;

padding: 10px;

}

.control-izquierdo {

margin-left: 10px;

}

.control-derecho {

margin-right: 10px;

}

### Paso 3: JavaScript (scripts.js)

javascript

Copiar código

document.addEventListener("DOMContentLoaded", function() {

const slides = document.querySelectorAll(".slide");

const tiempoIntervalo = 5000; // Tiempo en milisegundos entre cada cambio de imagen

let indiceSlide = 0;

// Función para mostrar el slide actual

function mostrarSlide() {

slides.forEach(slide => {

slide.style.display = "none";

});

slides[indiceSlide].style.display = "block";

}

// Función para avanzar al siguiente slide

function siguienteSlide() {

indiceSlide++;

if (indiceSlide >= slides.length) {

indiceSlide = 0;

}

mostrarSlide();

}

// Intervalo para cambiar automáticamente de slide

setInterval(siguienteSlide, tiempoIntervalo);

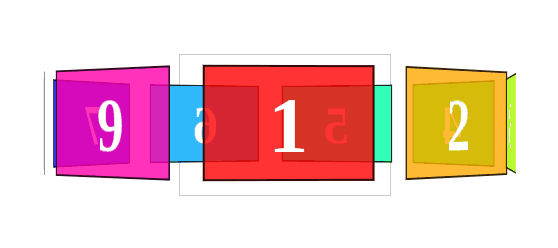
});

### Explicación:

1. **HTML (index.html)**:
   * Se define una estructura básica con un contenedor .carrusel que contiene varios elementos .slide, cada uno con una imagen (<img>).
2. **CSS (styles.css)**:
   * Se oculta el desbordamiento (overflow: hidden) del contenedor .carrusel para asegurar que solo se vea una imagen a la vez.
   * Los .slide se ocultan inicialmente (display: none) excepto el primero que se muestra por defecto (display: block).
   * width: 100% en .slide img asegura que las imágenes se ajusten al ancho del carrusel.
3. **JavaScript (scripts.js)**:
   * Se utiliza document.addEventListener("DOMContentLoaded", function() {...}) para asegurarse de que el JavaScript se ejecute después de que el DOM esté completamente cargado.
   * setInterval se utiliza para cambiar automáticamente al siguiente slide después de un intervalo de tiempo especificado (tiempoIntervalo).
   * mostrarSlide() y siguienteSlide() son funciones que manejan el cambio de slides. mostrarSlide() oculta todos los slides y muestra solo el slide actual (slides[indiceSlide]).
4. **Notas adicionales**:
   * Puedes personalizar el carrusel añadiendo controles de navegación (control-izquierdo y control-derecho en el ejemplo CSS) para permitir a los usuarios avanzar o retroceder manualmente entre las imágenes.
   * Asegúrate de ajustar los nombres de archivo (imagen1.jpg, imagen2.jpg, etc.) y los estilos CSS según las necesidades específicas de diseño de tu página web.

Este ejemplo proporciona una implementación básica de un carrusel de imágenes automático utilizando HTML, CSS y JavaScript. Puedes expandirlo añadiendo más funcionalidades como controles de navegación adicionales, efectos de transición entre slides o ajustes responsive para dispositivos móviles.

Principio del formulario



9 CREAR UN ENLACE DE UNA ARCHIVO DESCARGABLE 9 CREAR UN ENLACE DE UNA ARCHIVO DESCARGABLE 9 CREAR UN ENLACE DE UNA ARCHIVO DESCARGABLE 9 CREAR UN ENLACE DE UNA ARCHIVO DESCARGABLE 9 CREAR UN ENLACE DE UNA ARCHIVO DESCARGABLE 9 CREAR UN ENLACE DE UNA ARCHIVO DESCARGABLE 9 CREAR UN ENLACE DE UNA ARCHIVO DESCARGABLE 9 CREAR UN ENLACE DE UNA ARCHIVO DESCARGABLE 9 CREAR UN ENLACE DE UNA ARCHIVO DESCARGABLE 9 CREAR UN ENLACE DE UNA ARCHIVO DESCARGABLE 9 CREAR UN ENLACE DE UNA ARCHIVO DESCARGABLE Final del formulario

**9 CREAR UN ENLACE DE UNA ARCHIVO DESCARGABLE**

### Paso 1: Estructura HTML

html

Copiar código

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Enlace de Archivo Descargable</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<div class="contenedor">

<h2>Descargar Archivo</h2>

<p>Haz clic en el siguiente enlace para descargar el archivo:</p>

<a href="ruta/al/archivo.pdf" download="nombre-archivo.pdf" class="boton-descargar">Descargar Archivo PDF</a>

</div>

</body>

</html>

### Paso 2: Estilo CSS (styles.css)

css

Copiar código

.contenedor {

max-width: 600px;

margin: 20px auto;

text-align: center;

font-family: Arial, sans-serif;

}

.boton-descargar {

display: inline-block;

background-color: #007bff;

color: #fff;

text-decoration: none;

padding: 10px 20px;

border-radius: 5px;

margin-top: 10px;

}

.boton-descargar:hover {

background-color: #0056b3;

}

### Explicación:

1. **HTML (index.html)**:
   * Se define un enlace <a> con el atributo href apuntando a la ruta del archivo que se desea descargar (ruta/al/archivo.pdf).
   * El atributo download especifica el nombre del archivo que se descargará cuando se haga clic en el enlace (download="nombre-archivo.pdf"). Puedes cambiar "nombre-archivo.pdf" por el nombre que desees que tenga el archivo descargado.
   * Se añade un texto descriptivo y un contenedor (div.contenedor) para organizar el contenido.
2. **CSS (styles.css)**:
   * Se aplican estilos simples para centrar el contenido (text-align: center), establecer márgenes (margin) y ajustar el tamaño del contenedor (max-width).
   * display: inline-block en .boton-descargar asegura que el enlace se comporte como un bloque en línea con opciones de estilización adicionales.
   * background-color, color, padding, border-radius y otros estilos son aplicados para mejorar la apariencia del botón de descarga.

### Notas adicionales:

* No se necesita JavaScript para la funcionalidad básica de descarga de archivos. El atributo download en el enlace <a> maneja la descarga del archivo automáticamente.
* Asegúrate de proporcionar la ruta correcta y el nombre del archivo en el atributo href y download.
* Puedes personalizar los estilos CSS según las necesidades de diseño específicas de tu página web, como colores, tipografía, márgenes, etc.

Este ejemplo proporciona una manera simple y efectiva de crear un enlace para descargar un archivo en una página web utilizando HTML y CSS. Puedes expandirlo agregando más contenido descriptivo, mejorando los estilos o integrando funcionalidades adicionales si son necesarias para tu proyecto.



**10 CREAR UN EFECTO A UN ICONO DE LA PÁGINA**

### Paso 1: Crear la estructura HTML

Primero, necesitamos crear el marcado HTML básico que incluya el icono sobre el cual queremos aplicar el efecto.

Html

Copiar código

<!DOCTYPE html>

<html ont=”es”>

<head>

<meta charset=”UTF-8”>

<meta name=”viewport” content=”width=device-width, initial-scale=1.0”>

<title>Efecto en Icono</title>

<link rel=”stylesheet” href=”styles.css”>

</head>

<body>

<div class=”icon-container”>

<i id=”icon” class=”fas fa-star”></i>

</div>

<script src=”script.js”></script>

</body>

</html>

En este ejemplo, hemos utilizado un icono de estrella de Font Awesome (fas fa-star). Asegúrate de incluir la referencia al archivo de estilos CSS y al archivo de script JavaScript.

### Paso 2: Estilizar el icono con CSS

Luego, en el archivo styles.css, vamos a aplicar estilos básicos al icono y definir cómo se comportará cuando el usuario interactúe con él.

Css

Copiar código

/\* styles.css \*/

.icon-container {

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

}

#icon {

ont-size: 3rem;

color: #333; /\* Color inicial del icono \*/

transition: color 0.3s ease; /\* Transición suave de color \*/

}

#icon:hover {

color: #ffcc00; /\* Nuevo color al pasar el mouse \*/

cursor: pointer; /\* Cambia el cursor al pasar el mouse \*/

}

En este código CSS, hemos utilizado la pseudo-clase :hover para cambiar el color del icono (#ffcc00) cuando el usuario pasa el mouse sobre él. La propiedad transition asegura que el cambio de color sea suave y gradual.

### Paso 3: Agregar interactividad con JavaScript (opcional)

En este ejemplo, JavaScript no es necesario para el efecto de cambio de color, pero si deseas añadir interactividad adicional, como una animación o efecto diferente al hacer clic en el icono, puedes hacerlo creando un archivo script.js.

javascript

Copiar código

// script.js (opcional)

// Ejemplo de cómo añadir un efecto de clic

const icon = document.getElementById(‘icon’);

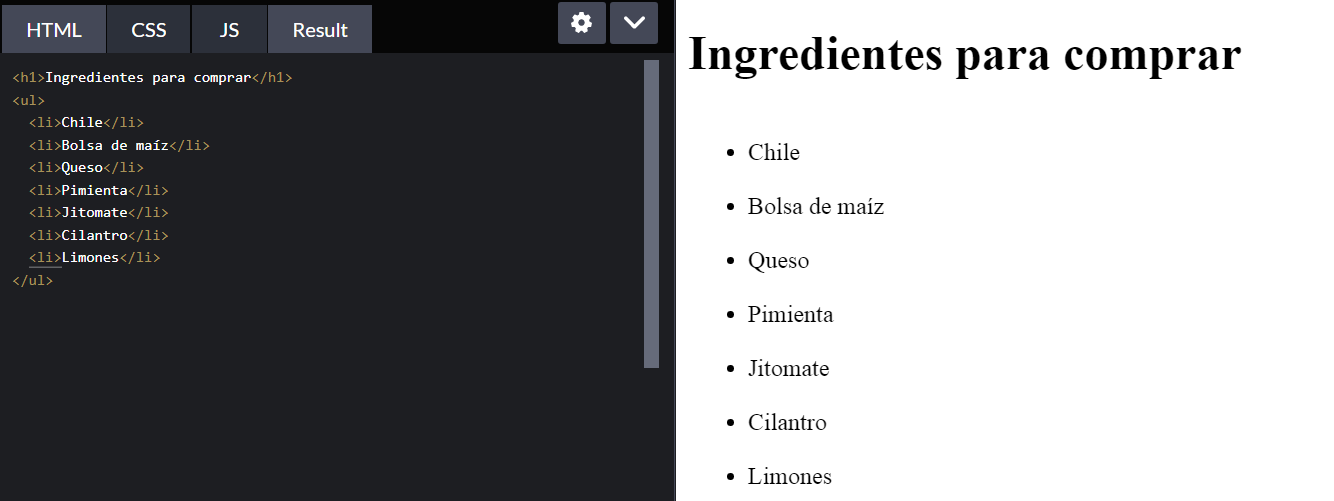
icon.addEventListener(‘click’, function() {

icon.classList.toggle(‘rotate’); // Añade o remueve la clase ‘rotate’

});

### Explicación adicional

* **HTML**: Creamos una estructura simple con un contenedor para el icono.
* **CSS**: Estilizamos el icono y definimos el efecto de cambio de color al pasar el mouse sobre él.
* **JavaScript**: Opcionalmente, se puede agregar interactividad adicional como una animación al hacer clic en el icono.

Este ejemplo muestra cómo puedes empezar a crear efectos simples en los iconos de tu página web utilizando HTML, CSS y opcionalmente JavaScript para interacciones más complejas. 

**11 PROCEDIMIENTO PARA HOSTING DE LA PÁGINA (GITHUB/NETLIFY)**

### Paso 1: Preparación del Proyecto

1. **Estructura de archivos**: Asegúrate de que tu proyecto tenga la siguiente estructura básica de archivos:

plaintext

Copiar código

proyecto/

├── index.html

├── styles.css

├── script.js

└── (otros archivos y carpetas si es necesario)

Donde index.html es tu página principal, styles.css contiene los estilos y script.js contiene el código JavaScript.

### Paso 2: Hosting en GitHub Pages

GitHub Pages permite alojar sitios web estáticos directamente desde repositorios de GitHub.

1. **Crear un repositorio**: Crea un nuevo repositorio en GitHub.
2. **Subir tu proyecto**: Sube los archivos de tu proyecto al repositorio de GitHub.
3. **Configurar GitHub Pages**:
   * Ve a la configuración de tu repositorio en GitHub.
   * Desplázate hacia abajo hasta la sección "GitHub Pages".
   * En la opción "Source" elige main branch (o master branch dependiendo de tu configuración).
   * Guarda la configuración.
4. **Acceder al sitio**: Tu sitio estará disponible en https://tu\_usuario.github.io/tu\_repositorio/. Por ejemplo, si tu nombre de usuario es usuario y tu repositorio se llama mi-proyecto, el sitio estará en https://usuario.github.io/mi-proyecto/.

### Paso 3: Hosting en Netlify

Netlify es otro servicio popular para el alojamiento de sitios estáticos y ofrece integraciones y funciones avanzadas.

1. **Crear una cuenta en Netlify**: Si no tienes una cuenta, regístrate en Netlify.
2. **Subir tu proyecto**:
   * Inicia sesión en Netlify.
   * En el panel principal, haz clic en "New site from Git" (Nuevo sitio desde Git).
   * Conecta tu repositorio de GitHub (o GitLab, Bitbucket, etc.) donde tienes tu proyecto.
   * Configura las opciones de construcción (por lo general, Netlify detectará automáticamente la configuración necesaria para un sitio estático básico).
3. **Desplegar tu sitio**:
   * Netlify comenzará a construir y desplegar tu sitio automáticamente.
   * Una vez completado, recibirás una URL del tipo https://nombre-aleatorio-de-sitio.netlify.app.

### Código Generado en HTML, CSS y JavaScript

A continuación, un ejemplo básico de cómo podrían ser los archivos HTML, CSS y JavaScript en tu proyecto:

#### index.html

html

Copiar código

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Mi Sitio Web</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<header>

<h1>Bienvenido a mi sitio web</h1>

</header>

<main>

<p>Este es un sitio web de ejemplo.</p>

<button onclick="saludar()">Haz clic para saludar</button>

</main>

<script src="script.js"></script>

</body>

</html>

#### styles.css

css

Copiar código

/\* styles.css \*/

body {

font-family: Arial, sans-serif;

background-color: #f0f0f0;

margin: 0;

padding: 0;

}

header {

background-color: #333;

color: #fff;

text-align: center;

padding: 10px 0;

}

main {

max-width: 800px;

margin: 20px auto;

padding: 20px;

background-color: #fff;

border: 1px solid #ccc;

border-radius: 5px;

}

button {

background-color: #007bff;

color: #fff;

border: none;

padding: 10px 20px;

cursor: pointer;

}

button:hover {

background-color: #0056b3;

}

#### script.js

javascript

Copiar código

// script.js

function saludar() {

alert('¡Hola, bienvenido a mi sitio web!');

}

Estos archivos constituyen un ejemplo básico de un sitio web estático que podrías desplegar en GitHub Pages o Netlify. Asegúrate de ajustar el contenido y estilos según tus necesidades antes de desplegarlo en producción.



**Final del formulario