UNIVERSIDAD MARIANO GALVEZ DE GUALTEMALA

FACULTAD DE INGENIERIA EN SISTEMAS

DESARROLLO WEB

ING. CARMELO ESTUARDO MAYEN MONTERROSO

OCTAVO CICLO

TAREA:

INVESTIGACION Y EJECUCION DE PORTAFOLIO DE TAREAS

NOMBRE:

JOSE DANIEL BRAN BENITO

CARNE:

1790-22-15044

CHIQUIMULILLA, SANTA ROSA 1 DE AGOSTO DEL 2025

LINKS

GITHUB: https://github.com/DanielBrn89/proyecto_portafolio_tareas

PAGINA DE PORTAFOLIO: https://proyecto-portafolio-tareas.vercel.app/

Tipos de elementos <input> en HTML5:

La etiqueta <input> admite una variedad de tipos de campos de entrada, cada uno definido por el atributo “type”. A continuación, se presentan algunos de los tipos más comunes de elementos junto con ejemplos funcionales para cada uno:

1. Texto

El tipo de entrada de texto se utiliza para crear campos de entrada de texto simples.

<input type="text" name="nombre" placeholder="Ingrese su nombre">

Ejemplo de un Input de texto:

2. Correo electrónico

El tipo de entrada de correo electrónico se utiliza para validar que el texto ingresado sea una dirección de correo electrónico válida.

<input type="email" name="correo" placeholder="ejemplo@dominio.com">

Ejemplo de un Input de email:

3. Contraseña

El tipo de entrada de contraseña se utiliza para crear campos de contraseña que ocultan los caracteres escritos.

<input type="password" name="contraseña" placeholder="Ingrese su contraseña">

Ejemplo de un Input de contraseña:

4. Número

El tipo de entrada de número se utiliza para aceptar valores numéricos.

<input type="number" name="cantidad" min="0" max="100" step="1">

5. Fecha

El tipo de entrada de fecha se utiliza para permitir a los usuarios seleccionar fechas en un calendario.

<input type="date" name="fecha">

6. Archivo

El tipo de entrada de archivo permite a los usuarios cargar archivos desde su dispositivo.

<input type="file" name="archivo">

7. Botón

El tipo de entrada de botón se utiliza para crear botones interactivos en un formulario.

<input type="button" value="Enviar">

8. Casilla de verificación

El tipo de entrada de casilla de verificación permite a los usuarios seleccionar una o varias opciones de un conjunto de opciones.

<input type="checkbox" name="opciones" value="opcion1"> Opción 1
<input type="checkbox" name="opciones" value="opcion2"> Opción 2

- [ ] Opción 1
- [ ] Opción 2

9. Botón de radio

El tipo de entrada de botón de radio permite a los usuarios seleccionar una opción de un conjunto de opciones.

<input type="radio" name="opcion" value="opcion1"> Opción 1
<input type="radio" name="opcion" value="opcion2"> Opción 2

Opción 1

Opción 2

10. Área de texto

El elemento <textarea> se utiliza para crear áreas de texto de varias líneas.

<textarea name="mensaje" rows="4" cols="40">Escriba su mensaje aquí</textarea>

11. Reset

El tipo de entrada de reset se utiliza para restablecer los valores de un formulario a sus valores predeterminados.

<input type="reset" value="Restablecer">

12. Teléfono

El tipo de entrada de teléfono se utiliza para permitir a los usuarios ingresar números de teléfono.

<input type="tel" name="telefono" placeholder="Ingrese su número de teléfono">

13. Color

El tipo de entrada de color se utiliza para permitir a los usuarios seleccionar un color.

<input type="color" name="color">

14. Rango

El tipo de entrada de rango se utiliza para permitir a los usuarios seleccionar un valor en un rango determinado.

<input type="range" name="rango" min="0" max="100" step="1">

15. Imagen

El tipo de entrada de imagen se utiliza para mostrar imágenes como botones interactivos.

<input type="image" src="imagen.png" alt="Imagen de ejemplo">

16. Búsqueda

El tipo de entrada de búsqueda se utiliza para permitir a los usuarios buscar información dentro de un campo de búsqueda.

<input type="search" name="busqueda" placeholder="Buscar...">

17. Time

El tipo de entrada de tiempo se utiliza para permitir a los usuarios seleccionar una hora específica.

<input type="time" name="hora">

La etiqueta <textarea>

Esta etiqueta tiene algunos atributos que podría ser interesante destacar:

Atributo	Descripción
cols	Número de carácteres que caben en horizontal (columnas).
rows	Número de carácteres que caben en vertical (filas).
wrap	Con soft (por defecto) o hard puedes indicar si el texto debe enviarse ajustado o tal cuál se e

Esta etiqueta nos permite añadir gran cantidad de texto, ya que las etiquetas <input type="text"> sólo nos permiten escribir una línea de texto y es muy incómodo si nuestra intención es escribir grandes cantidades de texto:

```
<form method="post" action="/send/">
  <div>
    Nombre: <input name="nickname">
  </div>
  <textarea name="text" cols="32" rows="8">
Este es el valor de texto por defecto</textarea>
</form>
```

Observa que en este caso, la etiqueta <textarea> tiene un valor por defecto, pero no lo hemos indicado en el atributo value como hacíamos en la etiqueta <input>, sino que aquí lo incluímos en el interior de la etiqueta <textarea>, ya que esta etiqueta si tiene etiqueta de cierre.

Ten en cuenta que el aspecto visual y la colocación se puede modificar con CSS. En estos ejemplos estoy intentando mantener el código HTML lo más puro posible para entender como funciona de serie.

Tamaño del ***<textarea>***

Si queremos modificar las dimensiones del elemento <textarea> utilizando única y exclusivamente HTML, podemos utilizar los atributos cols y rows. Estos atributos nos permiten aumentar el tamaño de ancho y alto del área de texto. Los valores numéricos de estos atributos equivalen al tamaño aproximado de un carácter en este campo, de modo que si indicamos cols="20" significa que habrá un espacio de ancho de aproximadamente 20 carácteres.

Veamos un ejemplo básico con una etiqueta HTML <textarea>:

```
<form method="post" action="/send/">
  <div>
    Nombre: <input name="nickname">
  </div>
  <textarea name="text" cols="32" rows="10"></textarea>
</form>
```

Además, recuerda que también puedes cambiar el tamaño de este área de texto utilizando CSS, por ejemplo, con las propiedades width y height. A veces interesa más hacerlo desde CSS, y otras veces desde HTML, incluso podría ser necesario utilizar las dos formas.

Subir a tabs

Área de texto ajustable

Por defecto, si escribes texto en un campo <textarea> hasta el punto que no cabe el texto, aparecerán unas barras de desplazamiento, de modo que el contenido hace scroll hacia dentro.

Sin embargo, podemos utilizar la nueva propiedad CSS field-sizing con el valor content, lo que hará que en lugar de lo anterior, el campo <textarea> crezca y se adapte al contenido:

```
<label><input type="checkbox"> Activar <code>field-sizing: content</code></label>


<form method="post" action="/send/">
  <div>
    Nombre: <input name="nickname">
```

```
  </div>
  <textarea name="text">
Sigue escribiendo texto en este campo, para ver
como se aumenta el área de texto.</textarea>
</form>
```

¿Qué es CSS?

Antes de comenzar, debes tener claro un concepto clave: una página web es realmente un documento de texto. En dicho documento se escribe código HTML, con el que se que crea el contenido de una web. Por otro lado, existe el código CSS, que unido al código HTML permite darle forma, color, posición (y otras características visuales) a un documento web.

En resumen, se trata de un «idioma» o lenguaje, como podría ser el inglés o el alemán, que los navegadores web como Chrome o Firefox conocen y pueden entender. Nuestro objetivo como diseñadores y programadores web es precisamente ese: aprender el idioma e indicarle al navegador lo que debe hacer.

Los navegadores entienden los idiomas HTML y CSS

¿Qué significa CSS?
Las siglas CSS (Cascading Style Sheets) significan «Hojas de estilo en cascada» y parten de un concepto simple pero muy potente: aplicar estilos (colores, formas, márgenes, etc...) a uno o varios documentos (generalmente documentos HTML, páginas webs) de forma automática y masiva.

Se le denomina estilos en cascada porque se lee, procesa y aplica el código desde arriba hacia abajo (siguiendo patrones como herencia o cascada que trataremos más adelante) y en el caso de existir ambigüedad (código que se contradice), se siguen una serie de normas para resolver dicha ambigüedad.

Al empezar, siempre generalizamos mencionando que tenemos varios documentos HTML, pero sólo un documento CSS. En cada documento HTML enlazamos ese único documento CSS, de modo que si hacemos cambios en él, afecta a todos los documentos HTML relacionados. Esto es mucho más práctico que tener el CSS en cada uno de esos documentos, y tener que cambiarlos en cada uno de ellos.

¿Por qué se usa CSS?
La idea de CSS es la de utilizar el concepto de separación de presentación y contenido. Este concepto se basa en que, como programadores, lo ideal es separar claramente el código que escribimos. ¿Por qué? Porque con el tiempo, esto hará que el código sea más fácil de modificar y mantener.

La idea es la siguiente:

Los documentos HTML (contenido) incluirán sólo información y datos, todo lo relativo a la información a transmitir.
Los documentos CSS (presentación) inclurán sólo los aspectos relacionados con el estilo (diseño, colores, formas, etc...).
¿Qué es CSS?

De esta forma, se puede unificar todo lo relativo al diseño, a lo visual en un solo documento CSS, y con ello, varias ventajas:

Si necesitamos hacer modificaciones visuales, lo haremos en un sólo lugar y se aplica a todo el sitio.
Se reduce la duplicación de estilos en diferentes lugares. Es más fácil de organizar y hacer cambios.
La información a transmitir es considerablemente menor (las páginas se descargan más rápido).
Es más fácil crear versiones diferentes para otros dispositivos: tablets, smartphones, etc...

¿Qué son los frameworks CSS?

Los frameworks CSS son bibliotecas preconfiguradas de hojas de estilo CSS, que suelen incluir lenguajes de scripting como JavaScript o Sass para facilitar la interactividad y el mantenimiento. Simplifican el desarrollo de la interfaz de usuario (UI) al proporcionar estilos y componentes prediseñados, e incluyen estilos estandarizados para elementos como botones, tipografía y sistemas de cuadrícula, lo que facilita la creación de diseños adaptables . Como las hojas de estilo ya están configuradas, solo tiene que consultar la documentación y usar las clases, los ID o los componentes correctos para diseñar su sitio web.

Al usar estos frameworks, te ahorras tener que escribir todo tu CSS desde cero. Esto ahorra tiempo y, si se usa correctamente, garantiza la consistencia y la calidad en todo el proyecto. Olvídate de tener que usar "!important" por toda la hoja de estilos solo para centrar el texto.

Uso de media queries

Las ***Media queries*** le permiten aplicar estilos CSS según el tipo general de un dispositivo (como impresión o pantalla) u otras características como la resolución de la pantalla o el ancho del *viewport* del navegador. Las *media queries* se utilizan para lo siguiente:

- Para aplicar estilos condicionalmente utilizando las reglas de arroba CSS @media e @import.
- Para segmentar medios específicos para <style>, <link>, <source> y otros HTML con el atributo media=.
- Para probar y monitorear los estados de los medios usando los métodos Window.matchMedia() y EventTarget.addEventListener().

Nota: Los ejemplos en esta página usan @media de CSS con fines ilustrativos, pero la sintaxis básica sigue siendo la misma para todos los tipos de consultas de medios.

Sintaxis

Una *media query* se compone de un *tipo de medio* opcional y cualquier cantidad de expresiones de *características de medios*, que pueden combinarse opcionalmente de varias

maneras usando *operadores lógicos*. Las consultas de medios no distinguen entre mayúsculas y minúsculas.

- Los tipos de medios definen la amplia categoría de dispositivos para los que se aplica la consulta de medios: all, print, screen.
  El tipo es opcional (se asume que es all) excepto cuando se usan los operadores lógicos not o only.
- Las características multimedia describen una característica específica del user agent, dispositivo de salida o entorno:
  - any-hover
  - any-pointer
  - aspect-ratio
  - color
  - color-gamut
  - color-index
  - device-aspect-ratio Obsoleto
  - device-height Obsoleto
  - device-width Obsoleto
  - display-mode
  - dynamic-range
  - forced-colors
  - grid
  - height
  - hover
  - inverted-colors
  - monochrome
  - orientation
  - overflow-block
  - overflow-inline
  - pointer
  - prefers-color-scheme
  - prefers-contrast
  - prefers-reduced-motion
  - resolution
  - scripting
  - update
  - video-dynamic-range
  - width.

  Por ejemplo, la característica hover permite que una consulta pruebe si el dispositivo admite el desplazamiento sobre los elementos. Las expresiones de características de medios comprueban su presencia o valor y son completamente opcionales. Cada expresión de característica de medios debe estar entre paréntesis.
- Se pueden utilizar operadores lógicos para componer una *media query* compleja: not, and y only. También puede combinar múltiples *media queries* en una sola regla separándolas con comas.

Una *media query* se calcula como true cuando el tipo de medio (si se especifica) coincide con el dispositivo en el que se muestra un documento y todas las expresiones de características de medios se computan como verdaderas. Las consultas que involucran tipos de medios desconocidos siempre son falsas.

Nota: Una hoja de estilo con una *media query* adjunta a su etiqueta <link> se descargará incluso si la consulta devuelve false, la descarga se realizará pero la prioridad de la descarga será mucho menor. No obstante, su contenido no se aplicará a menos que y hasta que el resultado de la consulta cambie a true. Puede leer por qué sucede esto en el blog de Tomayac Por qué el navegador descarga hojas de estilo con consultas de medios que no coinciden.

Destinos de tipos de medios

Los tipos de medios describen la categoría general de un dispositivo determinado. Aunque los sitios web suelen diseñarse teniendo en cuenta las pantallas, es posible que desee crear estilos destinados a dispositivos especiales, como impresoras o lectores de pantalla basados en audio. Por ejemplo, este CSS es para las impresoras:

CSSCopy to Clipboard

```
@media print {
  /* … */
}
```

También puedes considerar múltiples dispositivos. Por ejemplo, esta regla @media usa dos consultas de medios tanto para dispositivos de pantalla como de impresión:

CSSCopy to Clipboard

```
@media screen, print {
  /* … */
}
```

Consulte tipos de medios para obtener una lista de todos los tipos de medios. Debido a que describen dispositivos solo en términos muy amplios, solo algunos están disponibles; para atributos más específicos, use *características de medios* en su lugar.

Destinos de características de los medios

Las características multimedia describen las características específicas de un user agent, dispositivo de salida o entorno determinado. Por ejemplo, puede aplicar estilos específicos a monitores de pantalla ancha, computadoras que usan ratón o dispositivos que se usan en condiciones de poca luz. Este ejemplo aplica estilos cuando el mecanismo de entrada *principal* del usuario (como un ratón) puede pasar sobre los elementos:

CSSCopy to Clipboard

```
@media (hover: hover) {
  /* … */
}
```

Muchas características de medios son *características de rango*, lo que significa que pueden tener el prefijo "min-" o "max-" para expresar restricciones de "condición mínima" o "condición máxima". Por ejemplo, este CSS aplicará estilos solo si el ancho del viewport de su navegador es igual o menor que 1250px:

CSSCopy to Clipboard

```
@media (max-width: 1250px) {
  /* … */
}
```

Si crea una consulta de características multimedia sin especificar un valor, los estilos anidados se utilizarán siempre que el valor de la función no sea cero (o none, en Nivel 4). Por ejemplo, este CSS se aplicará a cualquier dispositivo con una pantalla a color:

CSSCopy to Clipboard

```
@media (color) {
  /* … */
}
```

Si una característica no se aplica al dispositivo en el que se ejecuta el navegador, las expresiones relacionadas con esa característica multimedia siempre son falsas.

Para obtener más ejemplos de Características multimedia, consulte la página de referencia de cada característica específica.