

Curso HTML5, CSS y JavaScript

Introducción a los sitios web

- emilianoagustingallo@gmail.com
- <https://www.linkedin.com/in/emiliano-gallo/>

Una breve introducción

- **¿Qué es programar?** es organizar una secuencia de pasos ordenados a seguir para hacer una determinada cosa
- **En informática:** se refiere a la acción de crear programas o aplicaciones
- **Algoritmo:** es el conjunto de instrucciones organizadas y relacionadas entre sí.
- **Los programas informáticos** suelen seguir algoritmos que permiten llegar a la solución de un problema

Lenguajes más usados en la actualidad



python



¿Que es HTML?

- Lenguaje de Marcas de HyperTexto.
- HTML es el que permite que un navegador sepa lo que está leyendo y cómo debe visualizarse. De esta forma, los navegadores saben donde deben mostrar el texto, donde va una imagen, donde hay un enlace, que color tiene, etc.
- Creado en 1991
- La versión más reciente es HTML5, creada en 2014



Documento HTML simple



A screenshot of a code editor interface. At the top, there is a toolbar with icons for home, file, copy, paste, and run, followed by a green "Run »" button. The main area contains the following HTML code:

```
<!DOCTYPE html>
<html>
<head>
<title>Título</title>
</head>
<body>

<h1>Bienvenidos!</h1>
<p>Al curso de HTML 5, CSS y JavaScript.</p>

</body>
</html>
```

The code includes a title "Título", a heading "Bienvenidos!", and a paragraph "Al curso de HTML 5, CSS y JavaScript.". The code editor has a light gray background and syntax highlighting.

Bienvenidos!

Al curso de HTML 5, CSS y JavaScript.

Estructura de un documento HTML

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

La declaración `<!DOCTYPE html>` define que es un documento HTML

`<html>` es el elemento raíz de una página

`<head>` contiene información adicional acerca del documento

`<title>` especifica el título de un documento. Es el que se muestra en la barra de título del navegador

`<body>` define el cuerpo del documento y es el contenedor de los elementos visibles

Contenido de un documento HTML

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

`<h1>` define un encabezado grande

`<p>` define un párrafo

¿Qué es un elemento HTML?

Es aquel que está definido por una etiqueta de inicio, un contenido y una etiqueta de fin

`<tagname>Aquí va el contenido...</tagname>`

Nota: algunos elementos no tienen contenido ni etiqueta de cierre (ejemplo: `
`). Estos son llamados *elementos vacíos*

¿Qué es un atributo HTML?

- Los atributos proveen información adicional sobre los elementos HTML
- Siempre se declaran en el elemento de apertura
- Usualmente está en par “nombre/valor” como: **nombre=”valor”**

```
<a href="https://www.w3schools.com">Visitá W3Schools</a>
```

Encabezados <h1> <h6>

```
<!DOCTYPE html>
<html>
<body>

<h1>Encabezado 1</h1>
<h2>Encabezado 2</h2>
<h3>Encabezado 3</h3>
<h4>Encabezado 4</h4>
<h5>Encabezado 5</h5>
<h6>Encabezado 6</h6>

</body>
</html>
```

Encabezado 1

Encabezado 2

Encabezado 3

Encabezado 4

Encabezado 5

Encabezado 6

Párrafos <p>

```
<!DOCTYPE html>
<html>
<body>
<p>Esto es un párrafo.</p>
<p>Esto es otro párrafo.</p>
</body>
</html>
```

Esto es un párrafo.

Esto es otro párrafo.

Links <a>

```
<!DOCTYPE html>
<html>
<body>

<h2>Links en HTML</h2>

<a href="https://www.w3schools.com">Esto es un link</a>

</body>
</html>
```

Links en HTML

[Esto es un link](https://www.w3schools.com)

Marcador de texto

```
<!DOCTYPE html>
<html>
<body>

<h1>El elemento span</h1>

<p>Mi mamá tiene ojos <span style="color:blue;font-weight:bold">azules</span> y mi papa tiene ojos <span style="color:darkolivegreen;font-weight:bold">dark green</span>. </p>

</body>
</html>
```

El elemento span

Mi mamá tiene ojos **azules** y mi papa tiene ojos **dark green**.

Imágenes

```
<!DOCTYPE html>
<html>
<body>

<h2>Imágenes en HTML</h2>



</body>
</html>
```

Imágenes en HTML



Imágenes

```
<!DOCTYPE html>
<html>
<body>

<h2>Imágenes en HTML</h2>



</body>
</html>
```

Imágenes en HTML



Estilos en HTML `style=" "`

- El atributo **style** nos permite personalizar el elemento HTML.
- Nos permite modificar el color, el tamaño, la fuente tipográfica, etc.

```
<!DOCTYPE html>
<html>
<body>

<h2>Atributo style</h2>

<p style="color:red;">Esto es un texto rojo.</p>
<p style="color:blue;">Esto es un texto azul.</p>

</body>
</html>
```

Atributo style

Esto es un texto rojo.

Esto es un texto azul.

Elementos Inline y Block

Todos los elementos HTML tiene un valor por defecto para visualizarse dependiendo del tipo de elemento que es

- Un elemento del tipo **block** siempre comienza en una nueva línea y toma todo el ancho disponible
- Un elemento del tipo **inline** no comienza en una nueva línea y solamente toma el ancho que necesita

Elementos Inline

| | | | | | | | | |
|----------|----------|------------|--------|----------|----------|---------|----------|----------|
| <a> | <abbr> | <acronym> | | <bdo> | <big> | | <button> | <cite> |
| <code> | <dfn> | | <i> | | <input> | <kbd> | <label> | <map> |
| <object> | <output> | <q> | <samp> | <script> | <select> | <small> | | |
| <sub> | <sup> | <textarea> | <time> | <tt> | <var> | | | |

```
<!DOCTYPE html>
<html>
<body>

<p>Este es un elemento span <span style="border: 1px solid black">Hola mundo</span>
dentro de un párrafo.</p>

</body>
</html>
```

Este es un elemento span Hola mundo dentro de un párrafo.

Elementos Block

```
<address>      <article>      <aside>      <blockquote>  <canvas>      <dd>        <div>        <dl>        <dt>
<fieldset>     <figcaption>  <figure>      <footer>       <form>       <h1>-<h6>    <header>     <hr>        <li>
<main>         <nav>        <noscript>    <ol>          <p>          <pre>        <section>   <table>      <tfoot>
<ul>           <video>
```

```
<!DOCTYPE html>
<html>
<body>

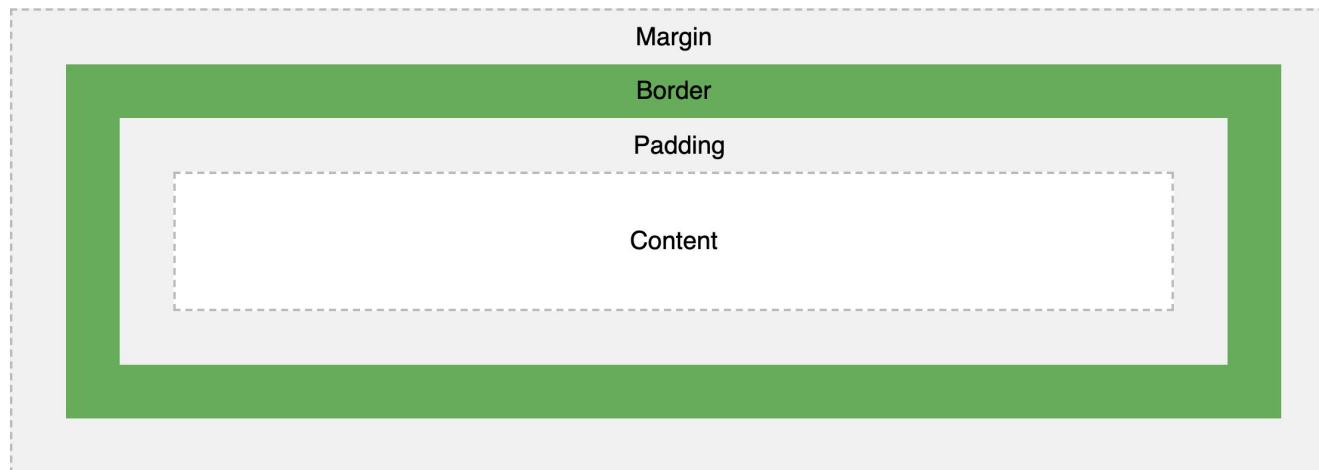
<div style="border: 1px solid black">Hola mundo</div>

</body>
</html>
```

Hola mundo

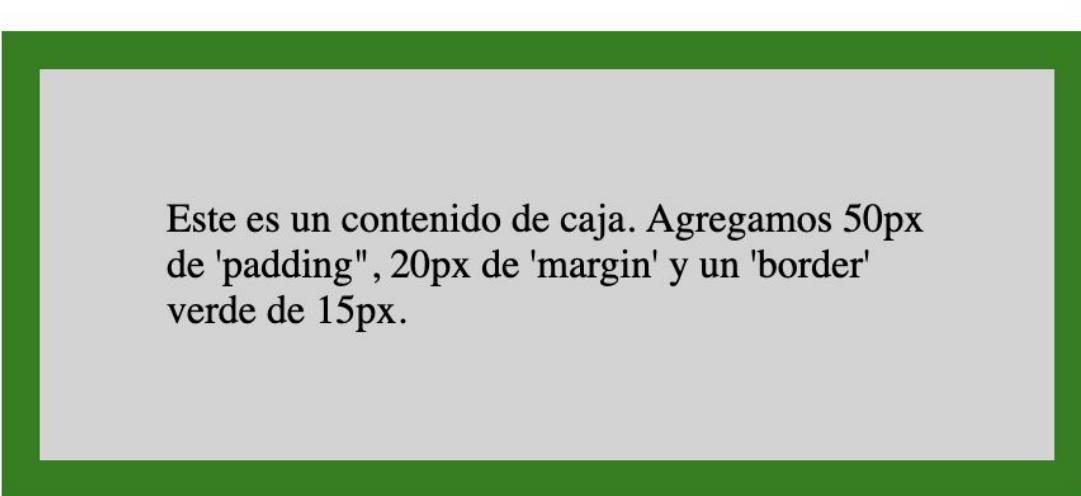
CSS Box model

Todos los elementos HTML son considerados como “cajas”. Este término es usado cuando diseñamos y disponemos elementos dentro del documento



Ejemplo de box model

```
<div style="background-color: lightgrey; width: 300px; border: 15px solid green; padding: 50px; margin: 20px;">  
    Este es un contenido de caja. Agregamos 50px de 'padding', 20px de 'margin' y un 'border' verde de 15px.  
</div>
```



Este es un contenido de caja. Agregamos 50px de 'padding', 20px de 'margin' y un 'border' verde de 15px.

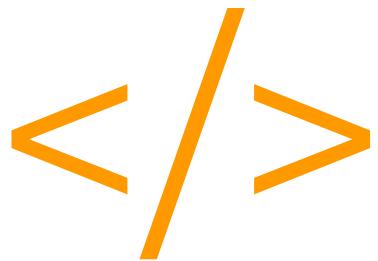
Recomendaciones

- Usar palabras en minúsculas para los elementos HTML
- Usar comillas para los valores de los atributos

```
<a href="https://www.w3schools.com/html/">Visit our HTML tutorial</a>
```

```
<a href=https://www.w3schools.com/html/>Visit our HTML tutorial</a>
```

Armemos una página sencilla!!



Desarrollar un sitio que luzca similar al de la imagen



A GLOBAL-LOCAL, VIRTUAL-PHYSICAL HAPPENING TO
MAKE BUSINESS MORE BEAUTIFUL

There's no going back to "normal." We need a reset, not just a recovery. Now is the time to dream bigger and build better.

[BECOME PART OF THE GREAT WAVE](#)

Experience a new type
of gathering

The Great Wave takes place online and offline, on screen and in thousands of offices, work spaces, households, public places—and immature. Join via virtual stream from anywhere or at one of the local hubs in more than 25 cities around the world.

Festuring talks, performances, experiments, deep dives, masterclasses, field trips, mystery meet-ups, silent dinners, masquerade balls, and much more, all converging in the most beautiful Monday meeting ever. More details coming soon.

HTML5

- Es la más reciente versión de HTML
- Incorpora una serie de elementos útiles para las web actuales
- Incorpora etiquetas que dan soporte a nuevas funcionalidades: es decir, aquellas que nos sirven para extender el HTML. Por ej: vídeo, sonido, lienzos donde diseñar dibujos
- Incorpora etiquetas que componen la web semántica: algunas etiquetas que realmente no proponen nuevas funcionalidades, sino que sirven para componer sitios indicando qué es cada bloque de código.

Funcionalidades multimedia

- Audio: inserta un audio dentro de una página `<audio></audio>`
- Video: inserta un video dentro de una página `<video></video>`
- Embed: inserta contenido externo dentro de una página (ej: plugins) `<embed></embed>`
- Track: especifica varias pistas de sonido o video `<track></track>`
- Source: especifica varias fuentes para un mismo audio o video `<source></source>`

Funcionalidades de formulario

- Meter: permite trabajar con medidas y escalas <`meter`></`meter`>
- Progress: implementa barra de progreso <`progress`></`progress`>
- Output: realiza y muestra cálculos matemáticos <`output`></`output`>
- Datalist: crea campos con funcionalidad de autocompletar <`datalist`></`datalist`>

Funcionalidades de dibujo

- Canvas: genera un lienzo en la página donde realiza cualquier tipo de diseño, soporta dibujo de todo tipo de formas, degradados, imágenes <canvas></canvas>

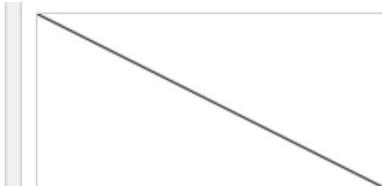
```
<!DOCTYPE html>
<html>
<body>

<canvas id="myCanvas" width="200" height="100" style="border:1px solid #d3d3d3;">
Tu navegador no soporta la etiqueta HTML Canvas</canvas>


```

```
<script>
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.moveTo(0,0);
ctx.lineTo(200,100);
ctx.stroke();
</script>

</body>
</html>
```



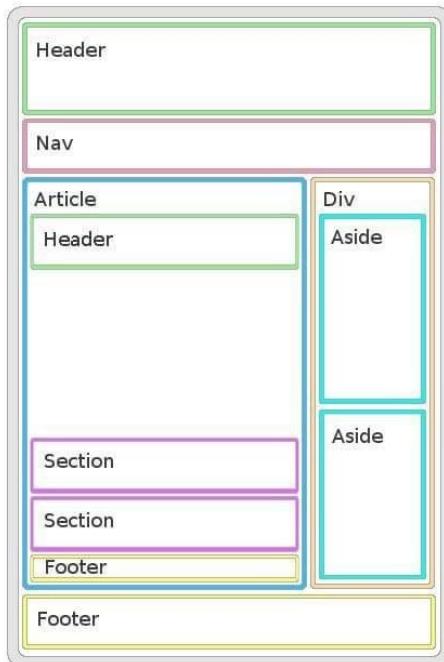
Web semántica

Describe su significado tanto para el navegador como para el desarrollador

- Artículo: especifica un artículo, es decir, una unidad de contenido `<article></article>`
- Section: es una sección dentro de un documento `<section></section>`
- Header: la cabecera de una página `<header></header>`
- Footer: el pie de página o informaciones que formen el pie de una sección `<footer></footer>`
- Aside: es una parte de la web que muestra contenido accesorio, generalmente colocado en un panel lateral `<aside></aside>`
- Nav: navegador principal de una página web `<nav></nav>`

Elementos

HTML tiene varios elementos semánticos que nos ayudan a definir las partes de una web



- `<header>` - Define la cabecera del documento
- `<nav>` - Define los links de navegación
- `<section>` - Define la sección del documento
- `<article>` - Define un contenido independiente
- `<aside>` - Define un contenido aparte como una barra lateral
- `<footer>` - Define el pie de un documento
- `<details>` - Define información adicional que el usuario puede mostrar u ocultar a demanda
- `<summary>` - Define un encabezado para `<details>`

Formularios HTML

Se utiliza un formulario HTML para recopilar información ingresada por un usuario. La entrada del usuario se puede enviar a un servidor para su procesamiento

```
<!DOCTYPE html>
<html>
<body>

<h2>Formulario HTML</h2>

<form action="/action_page.php">
  <label for="fname">Nombre:</label><br>
  <input type="text" id="fname" name="fname" value="John"><br>
  <label for="lname">Apellido:</label><br>
  <input type="text" id="lname" name="lname" value="Doe"><br><br>
  <input type="submit" value="Enviar">
</form>

</body>
</html>
```

Formulario HTML

Nombre:

John

Apellido:

Doe

Enviar

Diseños layouts

Los sitios webs usualmente muestran contenido en columnas

Cities

[London](#)
[Paris](#)
[Tokyo](#)

London

London is the capital city of England. It is the most populous city in the United Kingdom, with a metropolitan area of over 13 million inhabitants.

Standing on the River Thames, London has been a major settlement for two millennia, its history going back to its founding by the Romans, who named it Londinium.

Footer

Técnicas de diseños

Hay 4 diferentes técnicas para crear diseños multicolumna. Cada una de ella tiene pros y contras

- Flotante `float`
- Flexible `flexbox`
- Grilla `grid`
- Framework (ej. Bootstrap)

Diseño flotante float

La propiedad **float** es usada para posicionar y formatear contenido. Por ej: para dejar que una imagen flote a la izquierda de un texto. Los elementos flotantes están vinculados al flujo de documentos, lo que puede dañar la flexibilidad

- La propiedad **float** especifica como el elemento debería float
 - Left
 - Right
 - None
 - inherits
- La propiedad **clear** especifica qué elementos pueden flotar junto al elemento y de qué lado

Diseño flexible flex

El uso de **flexbox** asegura que los elementos se comporten de manera predecible cuando el diseño de la página debe acomodarse a diferentes tamaños de pantalla y diferentes dispositivos de visualización.

Facilita el diseño de una estructura de diseño flexible y responsive sin utilizar **float** o **position**

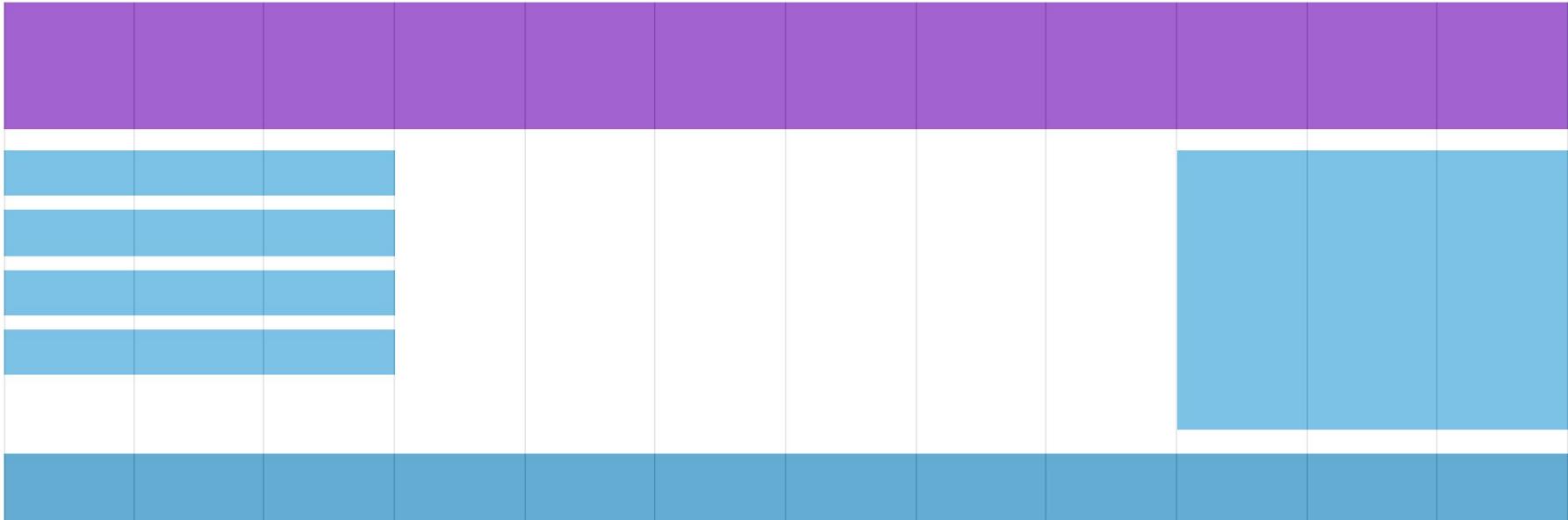
- flex-direction
- flex-wrap
- flex-flow
- justify-content
- align-items
- align-content

https://www.w3schools.com/css/css3_flexbox.asp

<https://flexboxfroggy.com/#es>

Diseño de grilla grid

Ofrece un sistema de diseño basado en cuadrícula, con filas y columnas, lo que facilita el diseño de páginas web sin tener que usar **float** y **position**.

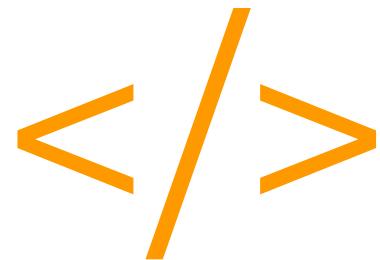


Tipos de rutas

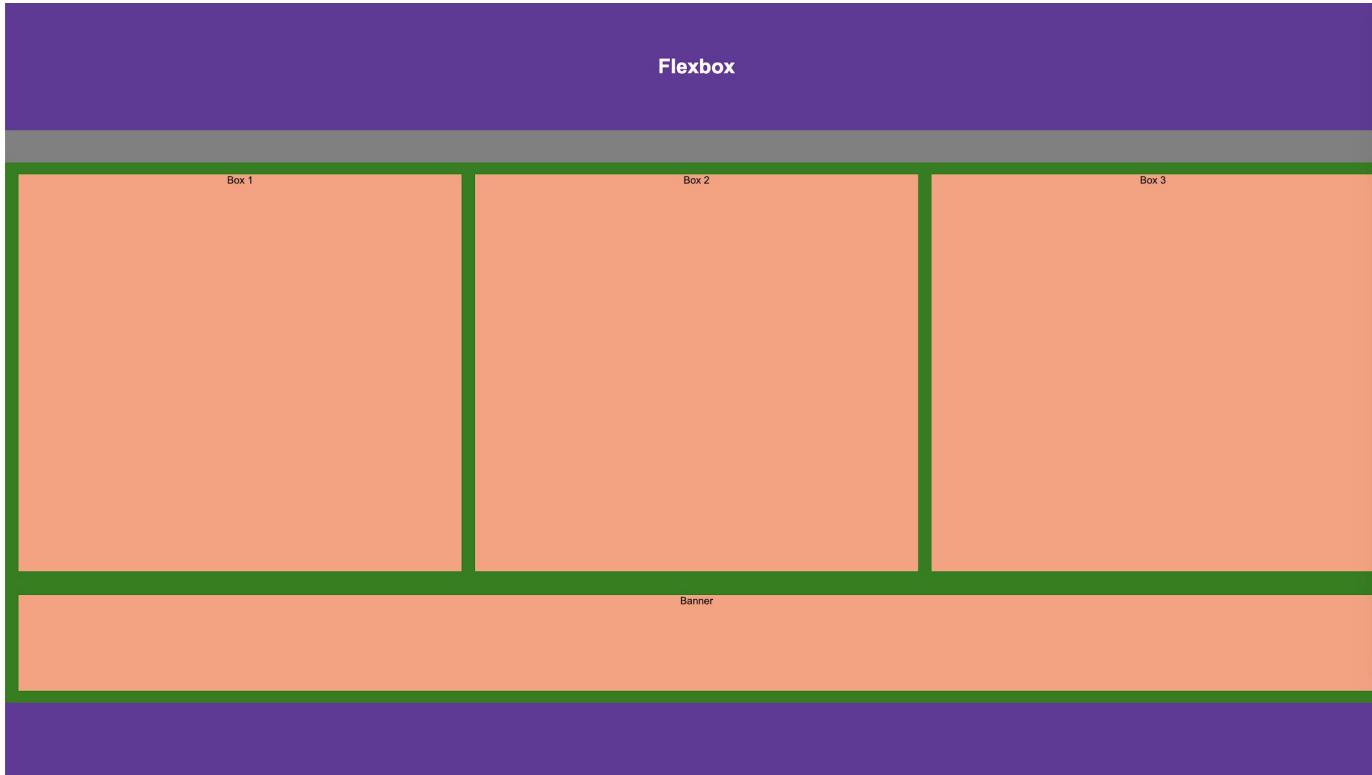
Contamos con 2 formas de acceder a los archivos dentro de un archivo HTML

- Ruta relativa: siempre se parte desde donde se encuentra el archivo HTML
 - En la misma ubicación: “NombreArchivo.extensión”
 - Una carpeta mas adentro: “NombreCarpeta/NombreArchivo.extensión”
 - Una carpeta mas arriba: “../NombreArchivo.extensión”
- Ruta absoluta: siempre se parte desde el raiz del disco rígido: “C: NombreUsuario/Mis documentos/NombreArchivo.extensión”

A trabajar!!



Desarrollar un sitio utilizando **flex** que luzca similar a la de la imagen



¿Qué es CSS?

- Es el lenguaje que da estilo a un documento HTML. Describe cómo los elementos HTML deberían ser mostrados
- Cascading **S**tyle **S**heets (hojas de estilo en cascada)
- Nos ahorra mucho trabajo. Podemos controlar la visualización de múltiples páginas web todas juntas
- Ejemplo: https://www.w3schools.com/css/css_intro.asp



Formas de implementar CSS

- **En línea:** es usado para aplicar un único estilo a un solo elemento. No se puede reutilizar
- **Interno:** puede ser usado si una sola página HTML tiene un único estilo. Se puede reutilizar dentro del mismo HTML
- **Externo:** puede ser usado en todos los documentos HTML de todo el sitio. Se puede cambiar el estilo completo de una página con solo cambiar un archivo

En linea

```
<!DOCTYPE html>
<html>
<body>

<h1 style="color:blue;text-align:center;">This is a heading</h1>
<p style="color:red;">This is a paragraph.</p>

</body>
</html>
```

Interno

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
    background-color: linen;
}

h1 {
    color: maroon;
    margin-left: 40px;
}
</style>
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

Externo

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

Selectores CSS

Los selectores son usados para “encontrar” (o seleccionar) el elemento HTML al cual le queremos aplicar el estilo.

Podemos dividirlos en 5 categorías:

- Selector simple (basado en el nombre, id o clase)
- Selector combinador (basado en la relación específica entre ellos)
- Selector pseudo-clase (basado en un estado específico)
- Selector pseudo-elemento (selecciona y da diseño a una parte específica de un elemento)
- Selectores de atributo (seleccione elementos basados en un atributo o valor de atributo)

Por el nombre del elemento

Todos los elementos `<p>` se van a mostrar alineados centralmente y con texto de color rojo

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
  text-align: center;
  color: red;
}
</style>
</head>
<body>

<p>Los elementos de párrafo van a ser afectados por el estilo.</p>
<p id="paral">Yo también!</p>
<p>Y yo!</p>
<h1>Y no!</h1>

</body>
</html>
```

Y no!

Los elementos de párrafo van a ser afectados por el estilo.

Yo también!

Y yo!

Por el id del elemento

La siguiente regla aplica para todos los elementos con `id="para1"`

```
<!DOCTYPE html>
<html>
<head>
<style>
#para1 {
    text-align: center;
    color: red;
}
</style>
</head>
<body>

<p id="para1">Hola!</p>
<p>Este párrafo no está afectado por el estilo.</p>

</body>
</html>
```

Este párrafo no está afectado por el estilo.

Hola!

Por la clase del elemento

El selector de clase selecciona elementos HTML con el atributo **class** específico

```
<!DOCTYPE html>
<html>
<head>
<style>
.center {
  text-align: center;
  color: red;
}
.large {
  font-size: 300%;
  color: black;
}
</style>
</head>
<body>

<p class="center">Este párrafo es rojo y está centrado.</p>
<p class="center large">Este párrafo es negro, está centrado y es de tamaño mas grande.
</p>

</body>
</html>
```

Este párrafo es rojo y está centrado.

Este párrafo es negro, está centrado y es de tamaño mas grande.

Por el nombre y la clase del elemento

El selector de clase selecciona elementos HTML con el atributo **class** específico

```
<!DOCTYPE html>
<html>
<head>
<style>
p.center {
  text-align: center;
  color: red;
}
</style>
</head>
<body>

<h1 class="center">Este encabezado no está afectado.</h1>
<p class="center">Este encabezado es rojo y está centrado.</p>

</body>
</html>
```

Este encabezado no está afectado.

Este encabezado es rojo y está centrado.

Selector universal

El selector universal aplica a todos los elementos del documento HTML

```
<!DOCTYPE html>
<html>
<head>
<style>
* {
  text-align: center;
  color: blue;
}
</style>
</head>
<body>

<h1>Hola!</h1>

<p>Todos los elementos de la página son afectados por el estilo.</p>
<p id="paral">Yo también!</p>
<p>Y yo!</p>

</body>
</html>
```

Hola!

Todos los elementos de la página son afectados por el estilo.

Yo también!

Y yo!

Selector de agrupación

Selecciona todos los elementos HTML con la misma definición de estilo

```
h1 {  
    text-align: center;  
    color: red;  
}  
  
h2 {  
    text-align: center;  
    color: red;  
}  
  
p {  
    text-align: center;  
    color: red;  
}
```

h1, h2, p {
 text-align: center;
 color: red;
}

Selector descendiente

Coincide con todos los elementos que son descendientes de un elemento especificado

```
<!DOCTYPE html>
<html>
<head>
<style>
div p {
    background-color: yellow;
}
</style>
</head>
<body>

<div>
    <p>Párrafo 1 dentro del div.</p>
    <p>Párrafo 2 dentro del div.</p>
    <section><p>Párrafo 3 dentro del div.</p></section>
<!-- not Child but Descendant -->
    <p>Párrafo 4 dentro del div.</p>
</div>

<p>Párrafo 5 fuera del div.</p>
<p>Párrafo 6 fuera del div.</p>

</body>
</html>
```

Párrafo 1 dentro del div.

Párrafo 2 dentro del div.

Párrafo 3 dentro del div.

Párrafo 4 dentro del div.

Párrafo 5 fuera del div.

Párrafo 6 fuera del div.

Selector hijo

El selector hijo selecciona todos los elementos que son hijos de un elemento especificado

```
<!DOCTYPE html>
<html>
<head>
<style>
div > p {
    background-color: yellow;
}
</style>
</head>
<body>

<div>
    <p>Párrafo 1 dentro del div.</p>
    <p>Párrafo 2 dentro del div.</p>
    <section><p>Párrafo 3 dentro del div.</p></section>
    <!-- not Child but Descendant -->
    <p>Párrafo 4 dentro del div.</p>
</div>

<p>Párrafo 5 fuera del div.</p>
<p>Párrafo 6 fuera del div.</p>

</body>
</html>
```

Párrafo 1 dentro del div.

Párrafo 2 dentro del div.

Párrafo 3 dentro del div.

Párrafo 4 dentro del div.

Párrafo 5 fuera del div.

Párrafo 6 fuera del div.

Selector de hermano adyacente

Selecciona el elemento adyacentes de un elemento especificado. El elemento hermano debe tener el mismo elemento padre, y "adyacente" significa "inmediatamente siguiente".

```
<!DOCTYPE html>
<html>
<head>
<style>
div + p {
    background-color: yellow;
}
</style>
</head>
<body>

<div>
    <p>Párrafo 1 dentro del div.</p>
    <p>Párrafo 2 dentro del div.</p>
    <section><p>Párrafo 3 dentro del div.</p>
</section> <!-- not Child but Descendant -->
    <p>Párrafo 4 dentro del div.</p>
</div>

<p>Párrafo 5 dentro del div.</p>
<p>Párrafo 6 dentro del div.</p>

</body>
</html>
```

Párrafo 1 dentro del div.

Párrafo 2 dentro del div.

Párrafo 3 dentro del div.

Párrafo 4 dentro del div.

Párrafo 5 dentro del div.

Párrafo 6 dentro del div.

Selector general de hermano

El selector general de hermanos selecciona todos los elementos que son hermanos de un elemento especificado

```
<!DOCTYPE html>
<html>
<head>
<style>
div ~ p {
    background-color: yellow;
}
</style>
</head>
<body>

<div>
    <p>Párrafo 1 dentro del div.</p>
    <p>Párrafo 2 dentro del div.</p>
    <section><p>Párrafo 3 dentro del div.</p>
</section> <!-- not Child but Descendant -->
    <p>Párrafo 4 dentro del div.</p>
</div>

    <p>Párrafo 5 dentro del div.</p>
    <p>Párrafo 6 dentro del div.</p>

</body>
</html>
```

Párrafo 1 dentro del div.

Párrafo 2 dentro del div.

Párrafo 3 dentro del div.

Párrafo 4 dentro del div.

Párrafo 5 dentro del div.

Párrafo 6 dentro del div.

Selector de pseudo clase

Se usa para definir un estado especial de un elemento.

Por ejemplo, se puede usar para:

- Diseñar un elemento cuando un usuario pase el mouse sobre él
- Estilo de enlaces visitados y no visitados de manera diferente
- Dale estilo a un elemento cuando se enfoca

Ejemplo: https://www.w3schools.com/css/css_pseudo_classes.asp

Selector de pseudo elemento

Un pseudo elemento se usa para diseñar partes específicas de un elemento.

Por ejemplo, se puede usar para:

- Aplicar estilo a la primera letra o línea de un elemento
- Insertar contenido antes o después del contenido de un elemento

Ejemplo: https://www.w3schools.com/css/css_pseudo_elements.asp

Selector de atributo

Es usado para seleccionar elementos con un atributo específico y/o con un valor específico de atributo

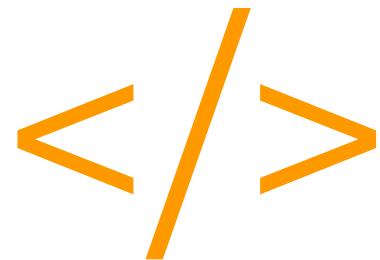
```
<!DOCTYPE html>
<html>
<head>
<style>
a[target] {
    background-color: yellow;
}
a[target=_top] {
    background-color: red;
}
</style>
</head>
<body>

<a href="https://www.w3schools.com">w3schools.com</a>
<br>
<a href="http://www.disney.com" target="_blank">disney.com</a>
<br>
<a href="http://www.wikipedia.org" target="_top">wikipedia.org</a>

</body>
</html>
```

w3schools.com
disney.com
wikipedia.org

A trabajar!!



Realizar un sitio de 4 secciones

- Menu superior con 4 botones, cada uno dirige a la sección correspondiente
- Cada sección debe contener una imagen, título, subtítulo y texto
- Aplicar la modalidad **CSS externo**
- Utilizar la mayor cantidad de selectores posibles



What do
we believe?

At Milk Moon, we believe that the nervous system and adrenals are a crucial, mostly overlooked component of postpartum health. In the early months (and years) of parenthood we are in a nearly constant state of high alert. The work of caring for a tiny human keeps our bodies in a low level fight-or-flight mode. Combined with a lack of high quality restorative sleep, this leads to burnout for so many moms. We believe that supporting and nourishing these systems enables women to show up to motherhood with a sense of calm, clarity and resilience.

¿Qué es el diseño web responsive?

- El diseño web adaptable o adaptativo (en inglés, **Responsive Web Design**) es una técnica de diseño y desarrollo web que mediante el uso de **estructuras e imágenes fluidas**, así como de **media-queries**, consigue adaptar el sitio web al entorno del usuario.
- El diseño web responsive utiliza solo **HTML** y **CSS**.
- El diseño web responsive no es un programa o código **JavaScript**

Diseñamos para todos los tamaños

Las páginas web se pueden ver usando muchos dispositivos diferentes

- Computadoras de escritorio
- Tabletas
- Teléfonos

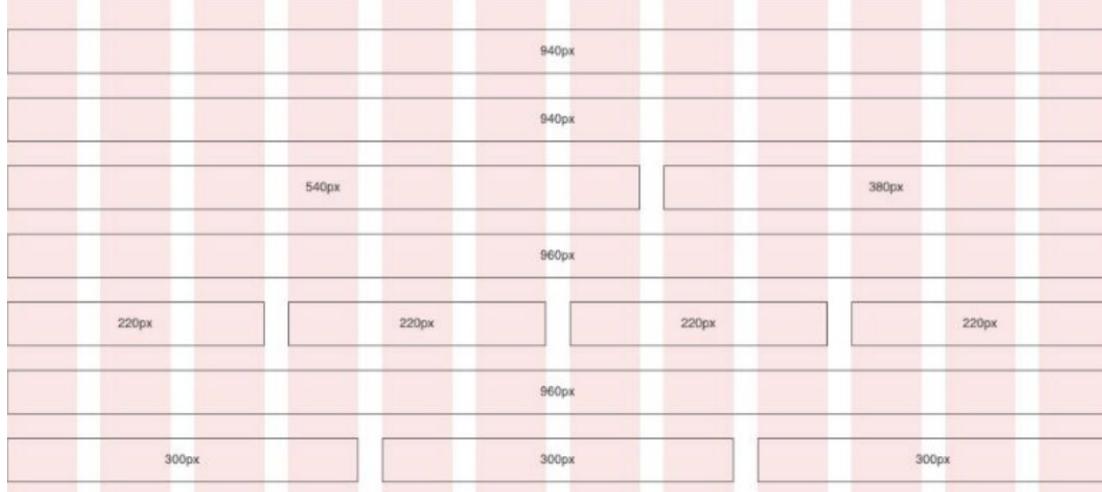
Y deben verse bien y ser fáciles de usar, independientemente del dispositivo.

Las páginas web no deben omitir información para adaptarse a dispositivos más pequeños, sino adaptar su contenido a cualquier dispositivo

Conceptos para empezar

Para poder empezar necesitamos entender ciertos conceptos, que constan en principio o en forma general de los tres siguientes

- Grillas
- Componentes flexibles
- Media Queries



Ventajas del diseño responsive

Las ventajas de trabajar con diseños responsive, en sí es que cualquier usuario puede acceder y entender el contenido, lo cual genera mayor conversión ya sea de servicios, o de productos

- Mejora experiencia del usuario (usabilidad y conversión)
- Mejora mantenimiento
- Evita contenido duplicado
- Mejora el posicionamiento Web

Diseño fluido

En este tipo de diseño se usan porcentajes en lugar de pixeles, para que los elementos se adapten según el ancho de la pantalla. Si bien el resultado puede parecer atractivo en pantallas medianas, como computadores de escritorio y tablets, se producen muchos problemas en las pantallas grandes y pequeñas.

Por ejemplo, en televisores las imágenes se estiran mucho y en teléfonos los textos son difíciles de leer.

Diseño adaptable

Usa plantillas estáticas basadas en puntos de quiebre. Cuando la pantalla alcanza cierto límite de tamaño, se cambia a otro diseño.

Por ejemplo, puedes diseñar una página en tres dimensiones diferentes. Cuando un usuario visite tu sitio, los archivos adaptativos detectarán el aparato que está usando y mostrarán la plantilla adecuada.

Sin embargo, ofrece un bajo nivel de usabilidad y muestra una versión reducida de los contenidos en móviles, un medio que crece con fuerza y gana cada vez más relevancia.

Diseño responsive

En el Responsive Web Design el diseño y el contenido se adaptan a cada pantalla, entregando una experiencia de usuario muy similar en resoluciones bajas, altas o en formatos de distintas pulgadas.

Las características a tener en cuenta son

- Ancho y alto de la ventana del navegador
- Orientación del dispositivo
- Proporción entre el alto y ancho de la pantalla
- Resolución del dispositivo, es decir, la precisión del detalle en las imágenes
- Imágenes de mapa de bits.

Siempre mobile first

Significa diseñar para dispositivos móviles antes de diseñar para computadoras de escritorio o cualquier otro dispositivo (esto hará que la página se muestre más rápido en dispositivos más pequeños)

Esto significa que debemos hacer algunos cambios en nuestro CSS

En lugar de cambiar los estilos cuando el ancho se hace más pequeño que 768px, deberíamos cambiar el diseño cuando el ancho se hace más grande que 768px.

Esto hará que nuestro diseño sea móvil primero

```
/* For mobile phones: */
[class*="col-"] {
    width: 100%;
}

@media only screen and (min-width: 768px) {
    /* For desktop: */
    .col-1 {width: 8.33%;}
    .col-2 {width: 16.66%;}
    .col-3 {width: 25%;}
    .col-4 {width: 33.33%;}
    .col-5 {width: 41.66%;}
    .col-6 {width: 50%;}
    .col-7 {width: 58.33%;}
    .col-8 {width: 66.66%;}
    .col-9 {width: 75%;}
    .col-10 {width: 83.33%;}
    .col-11 {width: 91.66%;}
    .col-12 {width: 100%;}
}
```

Porque mobile first

El 85% de la población utiliza móviles, este dato no es menor. De esta manera, en realidad , más que **Mobile First** nos centramos en **Content First**, es decir en el contenido y lo que queremos comunicar más que en la cuestión más superficial y exterior de la página

- Se simplifica el diseño
- Se pone el contenido en primer lugar y se trata de priorizar lo esencial
- No hay información de más sin tanto sentido para comprender el sitio

Ventana gráfica **viewport**

- El **viewport** es el área visible del usuario de una página web
- Varía con el dispositivo y será más pequeña en un teléfono móvil que en la pantalla de una computadora
- Antes de las tabletas y los teléfonos móviles, las páginas web estaban diseñadas solo para pantallas de computadora, y era común que las páginas web tuvieran un diseño estático y un tamaño fijo.
- Fuente <meta> tag: https://www.w3schools.com/tags/tag_meta.asp

Configurando el `viewport`

HTML5 introdujo un método para permitir que los diseñadores web tomen el control de la ventana gráfica, a través de la etiqueta `<meta>`

Debe incluir el siguiente elemento de ventana gráfica `<meta>` en todas sus páginas web:

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

Esto le da al navegador instrucciones sobre cómo controlar las dimensiones y la escala de la página.

La parte `width = device-width` establece el ancho de la página para seguir el ancho de la pantalla del dispositivo (que variará según el dispositivo).

La parte `initial-scale = 1.0` establece el nivel de zoom inicial cuando el navegador carga por primera vez la página.

Importancia de viewport



Without the viewport meta tag



With the viewport meta tag

Ajustar el contenido al **viewport**

Los usuarios están acostumbrados a desplazarse verticalmente por sitios web, pero no horizontalmente!

Por lo tanto, si el usuario se ve obligado a desplazarse horizontalmente, o alejarse, para ver toda la página web, resulta en una mala experiencia del usuario.

- NO usar elementos grandes de ancho fijo: por ejemplo, si una imagen se muestra con un ancho más ancho que la vista, puede hacer que la vista se desplace horizontalmente
- NO permitir que el contenido se base en un viewport de ancho fijo para renderizar bien: dado que las dimensiones de la pantalla y el ancho en píxeles CSS varían ampliamente entre dispositivos
- Usar **media query** para aplicar diferentes estilos para pantallas pequeñas y grandes

Media queries

Para entender el concepto de Media Query, iremos a la definición oficial del W3C que dice: **Una media query consiste en un tipo de dispositivo (media) y cero o más expresiones que comprueban el estado de las características particulares de ese dispositivo.**

Mediante el uso de media queries, la presentación del contenido puede adaptarse a un rango específico de dispositivos de salida sin necesidad de modificar el propio contenido. Una Media Query es una serie de condiciones que pueden aplicarse a nuestras hojas de estilo CSS para que, según el tamaño y resolución del dispositivo, adopte una u otra forma

Importancia de las media queries

- Nos da la funcionalidad de **diseño responsive** en nuestro sitio.
- Permite que la URL de nuestra web sea la misma, mostrando un diseño que se adapta a cualquier dispositivo.
- Evita duplicar contenido y código.
- Es un estándar en los navegadores.
- Ahorramos tiempo al pensar en un solo diseño que se adapta a cualquier tamaño de pantalla.

Media queries

La consulta de medios es una técnica CSS introducida en CSS3.

Utiliza la regla `@media` para incluir un bloque de propiedades CSS solo si cierta condición es verdadera.

En el siguiente ejemplo, si la ventana del navegador es 600px o menor, el color de fondo será azul claro

```
@media only screen and (max-width: 600px) {  
    body {  
        background-color: lightblue;  
    }  
}
```

Media queries al importar CSS

También se pueden importar diferentes archivos de CSS dependiente la condición media query

```
<link rel="stylesheet" media="screen and (min-width: 900px)" href="widescreen.css">
<link rel="stylesheet" media="screen and (max-width: 600px)" href="smallscreen.css">
```

Uso de **landscape** y **portrait**

Describe la proporción de aspecto del dispositivo de salida. Este valor consiste de dos enteros positivos separados por una barra ("/").

Este representa la proporción de aspecto de los píxeles horizontales (primer término) a los píxeles verticales (segundo término).

```
@media screen and (device-aspect-ratio: 16/9){ ... }  
@media screen and (device-aspect-ratio: 16/10) { ... }
```

También se puede utilizar el **min-aspect-radio** donde por ejemplo la proporción debe ser al menos la misma

Puntos de quiebre breakpoints

Los **breakpoints** son los puntos de rotura o interrupción donde el diseño sufrirá un cambio drástico.

Según algunos autores, se deben cumplir 2 reglas:

- El contenido es el que marca el breakpoint y no el dispositivo
- El breakpoint no debe ser en pixeles, ha de ser en “em”, para darle el control al dispositivo

Se pueden crear tantos breakpoints como nos resulte necesario, es habitual crear entre 3 y 4 breakpoints, a los cuales se los llama comúnmente como small, medium, large y extra large

Puntos de quiebre breakpoints

Nos puede pasar que al crear una página web con filas y columnas, estas no se ajusten correctamente en una pantalla pequeña.

Las **media queries** pueden ayudarnos con eso. Podemos agregar un punto de interrupción donde ciertas partes del diseño se comporten de manera diferente en cada lado del punto de interrupción.

Cuando la ventana se vuelve más pequeña que 768 px, cada columna debe tener un ancho del 100%:

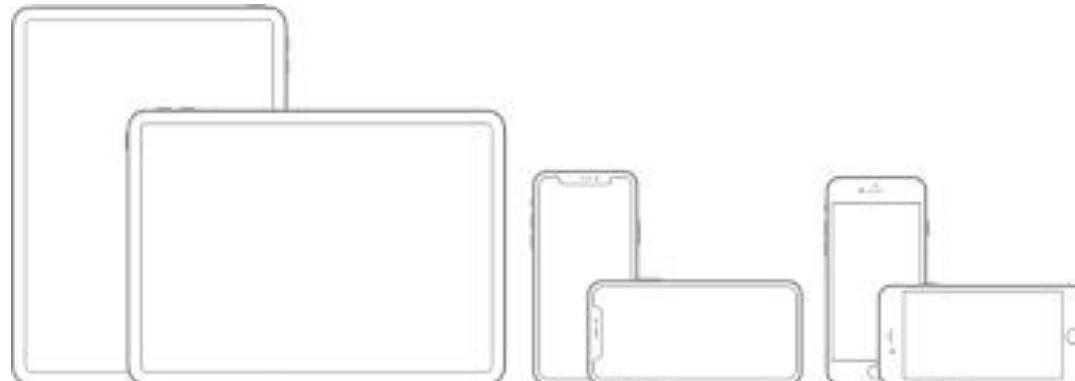
```
/* For desktop: */
.col-1 {width: 8.33%;}
.col-2 {width: 16.66%;}
.col-3 {width: 25%;}
.col-4 {width: 33.33%;}
.col-5 {width: 41.66%;}
.col-6 {width: 50%;}
.col-7 {width: 58.33%;}
.col-8 {width: 66.66%;}
.col-9 {width: 75%;}
.col-10 {width: 83.33%;}
.col-11 {width: 91.66%;}
.col-12 {width: 100%;}

@media only screen and (max-width: 768px) {
    /* For mobile phones: */
    [class*="col-"] {
        width: 100%;
    }
}
```

Uso de device-aspect-ratio

El valor de orientation es portrait (retrato) cuando el valor de la altura del medio de salida es mayor o igual al valor de la anchura de ese medio. De lo contrario el valor es landscape (paisaje).

```
@media screen and (orientation: landscape) {}  
@media screen and (orientation: portrait) {}
```



Propiedad `display`

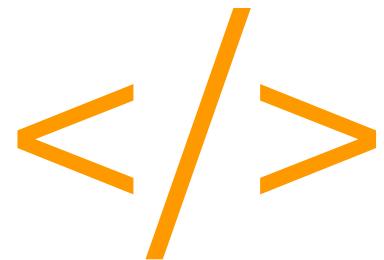
Especifica el comportamiento de visualización (el tipo de cuadro de representación) de un elemento.

En HTML, el valor de propiedad de visualización predeterminado se toma de las especificaciones HTML o de la hoja de estilo.

Referencia: https://www.w3schools.com/cssref/pr_class_display.asp

Ejemplo: https://www.w3schools.com/cssref/playit.asp?filename=playcss_display&preval=table

A trabajar!!



Diseñar un sitio responsive

- Diseñar un sistema de grillas de 4 columnas para dispositivos desktop y de 1 columna para dispositivos celulares y tablets
- Ubicar distintos elementos en las columnas (imágenes, textos, párrafos, etc)
- Al pasar de la vista ‘desktop’ a la ‘mobile’ los componentes tienen que acomodarse en una sola columna



Unidades de medida

CSS tiene varias unidades diferentes para expresar una longitud.

Muchas propiedades CSS toman valores de "longitud", como **width**, **margin**, **padding**, **font-size**, etc.

La longitud es un número seguido de una unidad de longitud, como 10px, 2em, etc.

Hay dos tipos de unidades de longitud: **absoluta** y **relativa**.

Medidas absolutas

Las unidades de longitud absolutas son fijas y una longitud expresada en cualquiera de ellas aparecerá exactamente de ese tamaño.

No se recomienda el uso de unidades de longitud absoluta en la pantalla, porque los tamaños de pantalla varían mucho.

Sin embargo, se pueden usar si se conoce el medio de salida, por ej: para el diseño de una impresión.

Diferentes medidas absolutas

- Centímetros: **cm**
- Milímetros: **mm**
- Pulgadas ($1\text{in} = 96\text{px} = 2.54\text{cm}$): **in**
- Píxeles ($1\text{px} = 1/96\text{th of 1in}$): **px**
- Puntos ($1\text{pt} = 1/72 \text{ of 1in}$): **pt**
- Picas ($1\text{pc} = 12 \text{ pt}$): **pc**

px son relativos al dispositivo. Para dispositivos de baja resolución, 1 px es un píxel (punto) de dispositivo de la pantalla. Para impresoras y pantallas de alta resolución, 1px implica múltiples píxeles del dispositivo.

Diferentes medidas absolutas

```
<!DOCTYPE html>
<html>
<head>
<style>

.cm {
    font-size: 0.5cm;
    line-height: 1cm;
}
.pc {
    font-size: 3.5pc;
    line-height: 3pc;
}
.pt {
    font-size: 40pt;
    line-height: 25pt;
}
.px {
    font-size: 15px;
    line-height: 20px;
}
.in {
    font-size: 0.2in;
    line-height: 0.5in;
}
.mm {
    font-size: 11mm;
    line-height: 10mm;
}
</style>
</head>
<body>

<p class="cm">Tamaño de texto en centímetros.</p>
<p class="pc">Tamaño de texto en picas.</p>
<p class="pt">Tamaño de texto en puntos.</p>
<p class="px">Tamaño de texto en píxeles.</p>
<p class="in">Tamaño de texto en pulgadas.</p>
<p class="mm">Tamaño de texto en milímetros.</p>

</body>
</html>
```

Tamaño de texto en centímetros.

Tamaño de texto en picas.

Tamaño de texto en puntos.

Tamaño de texto en píxeles.

Tamaño de texto en pulgadas.

Tamaño de texto en milímetros.

Medidas relativas

Las unidades de longitud relativa especifican una longitud relativa a otra propiedad de longitud.

Escalan mejor entre diferentes medios de representación.

Son muy usadas para los diseños responsive.

Diferentes medidas relativas

- Relativo al font-size del elemento (2em significa 2 el tamaño de la fuente actual): **em**
- Relativo al x-height de la fuente actual: **ex**
- Relativo al ancho del número "0": **ch**
- Relativo al font-size del elemento raiz: **rem**
- Relativo al 1% del ancho del viewport: **vw**
- Relativo al 1% del alto del viewport: **vh**
- Relativo al elemento padre: **%**

em y **rem** son muy útiles para crear diseños escalables

Viewport = al tamaño de la ventana del navegador. Si el viewport es 50 cm de ancho, **1vw** = 0.5cm.

Diferentes medidas relativas

```
<!DOCTYPE html>
<html>
<head>
<style>
.parent {
  font-size: 2%;
}
.vmax {
  font-size: 2vmax;
}
.vmin {
  font-size: 2vmin;
}
.vh {
  font-size: 2vh;
}
.vw {
  font-size: 2vw;
}
.rem {
  font-size: 2rem;
}
.ch {
  font-size: 2rem;
}
.ex {
  font-size: 2ex;
}
.em {
  font-size: 2em;
}
</style>
</head>
<body>

<p class="parent">Tamaño de texto en %.</p>
<p class="vmax">Tamaño de texto en vmax.</p>
<p class="vmin">Tamaño de texto en vmin.</p>
<p class="vh">Tamaño de texto en vh.</p>
<p class="vw">Tamaño de texto en vw.</p>
<p class="rem">Tamaño de texto en rem.</p>
<p class="ch">Tamaño de texto en ch.</p>
<p class="ex">Tamaño de texto en ex.</p>
<p class="em">Tamaño de texto en em.</p>
```

Tamaño de texto en %.

Tamaño de texto en vmax.

Tamaño de texto en vmin.

Tamaño de texto en vh.

Tamaño de texto en vw.

Tamaño de texto en rem.

Tamaño de texto en ch.

Tamaño de texto en ex.

Tamaño de texto en em.

Trabajando con em y rem

La unidad **em** hace referencia al tamaño en puntos de la letra que se está utilizando. Si se utiliza una tipografía de 12 puntos, 1em equivale a 12 puntos.

Como se trata de una unidad de medida relativa, es necesario realizar un cálculo matemático para determinar la anchura real de ese margen. La unidad de medida **em** siempre hace referencia al tamaño de letra del elemento.

Por otra parte, todos los navegadores muestran por defecto el texto de los párrafos con un tamaño de letra de 16 píxel. Por tanto, en este caso el margen de **1em** equivale a un margen de anchura **16px**.

Trabajando con em y rem

Si el font-size es **16px** y establecemos **0.75em** en un elemento, este será de **12px**



Trabajando con em y rem

La gran ventaja de las unidades relativas es que siempre mantienen las proporciones del diseño de la página.

Establecer el margen de un elemento con el valor **1em** equivale a indicar que "el margen del elemento debe ser del mismo tamaño que su letra y debe cambiar proporcionalmente". En efecto, si el tamaño de letra de un elemento aumenta hasta un valor enorme, su margen de **1em** también será enorme. Si su tamaño de letra se reduce hasta un valor diminuto, el margen de **1em** también será diminuto.

El uso de unidades relativas permite mantener las proporciones del diseño cuando se modifica el tamaño de letra de la página.

Trabajando con em y rem

Las unidades de medida se pueden mezclar en los diferentes elementos de una misma página, como en el siguiente ejemplo

```
body { font-size: 10px; }
```

```
h1 { font-size: 2.5em; }
```

En primer lugar, se establece un tamaño de letra base de **10 px** para toda la página. Luego, se asigna un tamaño de **2.5em** al elemento, por lo que su tamaño de letra real será de **$2.5 \times 10\text{px} = 25\text{px}$** .

Trabajando con em y rem

Como el valor de la mayoría de propiedades CSS se **hereda de padres a hijos**. Así por ejemplo, si se establece el tamaño de letra al elemento , todos los elementos de la página tendrán el mismo tamaño de letra, salvo que indique otro valor.

Sin embargo, el valor de las medidas relativas no se hereda directamente, sino que se hereda su valor real una vez calculado.

El siguiente ejemplo muestra este comportamiento:

```
body { font-size: 12px; text-indent: 3em; }
```

```
h1 { font-size: 15px }
```

Trabajando con em y rem

La propiedad **text-indent**, se utiliza para tabular la primera línea de un texto. El elemento define un valor para esta propiedad, pero el elemento no lo hace, por lo que heredará el valor de su elemento padre. Sin embargo, el valor heredado no es **3em**, sino **36px**. Si se heredará el valor **3em**, al multiplicarlo por el valor de font-size del elemento (que vale 15px) el resultado sería **3em x 15px = 45px**.

Los valores que se heredan no son los relativos, sino los valores ya calculados. Por lo tanto, en primer lugar se calcula el valor real de **3em** para el elemento : **3em x 12px = 36px**. Una vez calculado el valor real, este es el valor que se hereda para el resto de elementos.

```
body { font-size: 12px; text-indent: 3em; }
```

```
h1 { font-size: 15px }
```

Diferencias entre **em** y **rem**

rem es una unidad de medida, basada en **em**, posee una diferencia muy importante, no hereda desde su elemento **padre**, sino desde el elemento **raíz** del documento, es decir desde la etiqueta **<body>**, de ahí viene su nombre **root em**.

Es una de las unidades de medida preferidas para la creación de proyectos responsivos, dado que permite trabajar de forma más cómoda y eficiente, evitando la creación de reglas de estilo redundantes como suele pasar cuando trabajamos con la unidad de medida **em**.

Porcentajes %

El porcentaje también es una unidad de medida relativa, aunque por su importancia CSS la trata de forma separada a **em**, y **px**.

Un porcentaje está formado por un valor numérico seguido del símbolo **%** y siempre está referenciado a otra medida.

Cada una de las propiedades de CSS que permiten indicar como valor un porcentaje, define el valor al que hace referencia ese porcentaje. Los porcentajes se pueden utilizar por ejemplo para establecer el valor del tamaño de letra de los elementos

```
body { font-size: 1em; }  
h1 { font-size: 200%; }  
h2 { font-size: 150%; }
```

Porcentajes %

Los porcentajes también se utilizan para establecer la anchura de los elementos.

En el siguiente ejemplo, la referencia del valor **80%** es la anchura de su elemento padre. Por tanto, el elemento cuyo atributo class es **principal** tiene una anchura de **80% x 600px = 480px**.

```
div#contenido { width: 600px; }
```

```
div.principal { width: 80%; }
```

```
<div id="contenido">
```

```
    <div class="principal"> </div>
```

```
</div>
```

Algunos ejemplos



Maquetar un sitio responsive

- Si el ancho de la pantalla es igual o mayor a 768px. Se debe visualizar como la imagen de la izquierda
- Si es menor a 768px. como la imagen de la derecha

Nuestro título

Opción 1

Opción 2

Opción 3

Opción 4

Sub título

Lorem ipsum dolor sit amet consectetur adipisicing elit. Voluptas nostrum sed nobis, molestias labore quos corrupti voluptatum voluptatibus consequatur laboriosam animi quidem tenetur nisi, pariatur facilis molestiae vero aperiam dicta!

Qué?

Lorem ipsum dolor sit amet consectetur adipisicing elit.

Dónde?

Lorem ipsum dolor sit amet consectetur adipisicing elit.

Cómo?

Lorem ipsum dolor sit amet consectetur adipisicing elit.

Modifica el tamaño del navegador para ver como responde el tamaño del contenido.

Nuestro título

Opción 1

Opción 2

Opción 3

Opción 4

Sub título

Lorem ipsum dolor sit amet consectetur adipisicing elit. Voluptas nostrum sed nobis, molestias labore quos corrupti voluptatum voluptatibus consequatur laboriosam animi quidem tenetur nisi, pariatur facilis molestiae vero aperiam dicta!

Qué?

Lorem ipsum dolor sit amet consectetur adipisicing elit.

Dónde?

Lorem ipsum dolor sit amet consectetur adipisicing elit.

Cómo?

Lorem ipsum dolor sit amet consectetur adipisicing elit.

Curso HTML5, CSS y JavaScript

Introducción a Bootstrap

- emilianoagustingallo@gmail.com
- <https://www.linkedin.com/in/emiliano-gallo/>

Frameworks y conceptos

Un Framework nos permite hacer nuestro **CSS** mucho más sencillo. De hecho, estos sistemas nos permiten trabajar con grillas, componentes pre-armados hasta elementos que incluyen de manera sencilla **JavaScript**.

Eso no significa que no tenemos que trabajar en absoluto, sino por el contrario , luego la idea es adaptar este framework a nuestras necesidad, muchas veces sobreescribiendo lo necesario para adaptarlo a nuestro **Look and Feel**.

Primer paso con Bootstrap

Para empezar a trabajar lo primero que debemos hacer es ir a la página oficial:

<http://getbootstrap.com/>

Luego en principio debemos vincular nuestro CSS. Bootstrap también cuenta con otros elementos como su propio **JS**, y **themes** para poder trabajar.

Inicialmente sólo necesitaremos vincular el **css**. Podemos hacerlo de dos formas, una es bajarnos el archivo de la página oficial , otra forma es trabajar con **CDN**

CSS

Copy-paste the stylesheet <link> into your <head> before all other stylesheets to load our CSS.

```
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sh" data-bbox="945 875 977 895" style="float: right;">Copy
```

Containers en Bootstrap

Los contenedores se utilizan para llenar el contenido dentro de ellos, y hay dos clases de contenedores disponibles:

- La clase **.container** proporciona un contenedor de ancho fijo sensible
- La clase **.container-fluid** proporciona un contenedor de ancho completo, que abarca todo el ancho de la ventana gráfica (**viewport**)

Contenedor .container

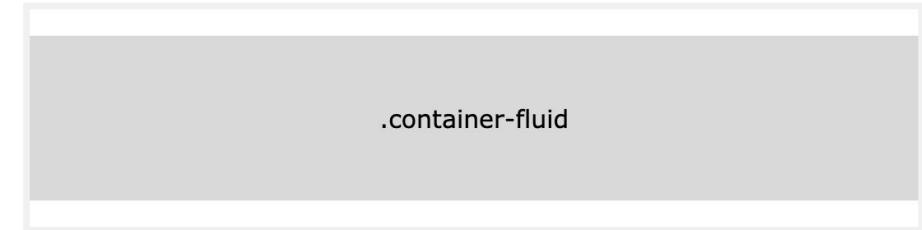
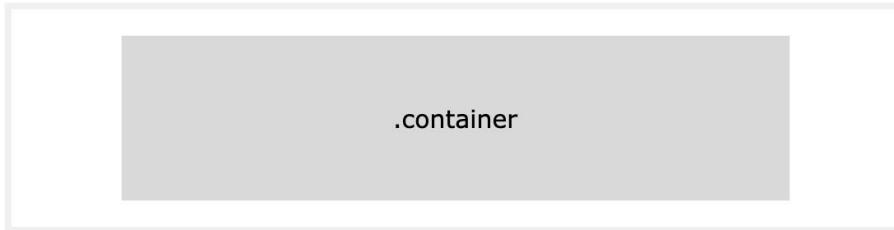
Utilice la clase **.container** para crear un contenedor responsive de ancho fijo.

Su ancho (ancho máximo) cambiará en diferentes tamaños de pantalla:

| | Extra small <576px | Small ≥576px | Medium ≥768px | Large ≥992px | Extra large ≥1200px |
|-----------|----------------------------------|-------------------------|--------------------------|-------------------------|--------------------------------|
| max-width | 100% | 540px | 720px | 960px | 1140px |

Contenedor .container-fluid

Use la clase **.container-fluid** para crear un contenedor de ancho completo, que siempre abarca todo el ancho de la pantalla (el ancho es siempre 100%)



Grillas en Bootstrap

Un detalle interesante y simple de trabajar, es el sistema de grillas que tenemos con bootstrap.

Por ejemplo, podemos incluir, grillas de **12 columnas**, donde indicaremos cuánto ocupará cada una en base al nombre de clase asignado

```
<div class="container">
  <div class="row">
    <div class="col-sm">
      One of three columns
    </div>
    <div class="col-sm">
      One of three columns
    </div>
    <div class="col-sm">
      One of three columns
    </div>
  </div>
</div>
```

Grillas en Bootstrap

El sistema de grilla de Bootstrap está construido con **flexbox** y permite hasta 12 columnas en la página. Es **responsive** y las columnas se reorganizan automáticamente según el tamaño de la pantalla.

Si no se desea utilizar las 12 columnas individualmente, se puede agrupar las columnas para crear columnas más anchas.

| | | | | | | | | | | | | | | | | |
|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--|--|--|--|--|
| span 1 | span 1 | span 1 | span 1 | span 1 | span 1 | span 1 | span 1 | span 1 | span 1 | span 1 | span 1 | | | | | |
| span 4 | | | span 4 | | | | span 4 | | | | | | | | | |
| span 4 | | | | | | | span 8 | | | | | | | | | |
| span 6 | | | | | | span 6 | | | | | | | | | | |
| span 12 | | | | | | | | | | | | | | | | |

Grillas en class

- **col-** dispositivos **extra small**, ancho de pantalla menor a **576px**
- **col-sm-** dispositivos **small**, ancho de pantalla mayor o igual a **576px**
- **col-md-** dispositivos **medium**, ancho de pantalla mayor o igual a **768px**
- **col-lg-** dispositivos **large**, ancho de pantalla mayor o igual a **992px**
- **col-xl-** dispositivos **extra large**, ancho de pantalla mayor o igual a **1200px**

Estas clases se pueden combinar para crear diseños más dinámicos y flexibles.

Cada clase se escala, por lo que si desea establecer los mismos anchos para **sm** y **md**, sólo necesita especificar **sm**.

Reglas para el uso de grillas

- Las filas deben colocarse dentro de un **.container** o **.container-fluid** para una alineación adecuada
- Utilizar filas **(class="row")** para crear grupos horizontales de columnas
- Las columnas se crean especificando el número de 12 columnas disponibles que se desea abarcar. Por ejemplo, **tres columnas iguales** usarían tres **.col-sm-4**
- Los anchos de columna están en porcentaje, por lo que siempre son fluidos y tienen un tamaño relativo a su elemento principal

Algunos ejemplos

