



# ***¿DUDAS DEL ON-BOARDING?***

[MIRALO AQUI](#)

***CODER HOUSE***

**Clase 04. DESARROLLO WEB**

***CSS***  
***MEDIDAS, COLORES, FUENTES***  
***+ BEM***



## OBJETIVOS DE LA CLASE

- Conocer el uso de medidas, fondos y fuentes en CSS.
- Conocer la metodología BEM
- Aplicar los atributos al Proyecto Final.

**CODER HOUSE**

## GLOSARIO:

### Clase 3

**Etiqueta IFRAME:** es un elemento HTML que permite insertar o incrustar un documento HTML dentro de un documento HTML principal.

**CSS (Cascading Style Sheets):** es un lenguaje web para aplicar formato visual (color, tamaño, separación y ubicación) al HTML. Con él puedes cambiar por completo el aspecto de cualquier etiqueta HTML.

**Padre e hijos:** es un concepto que se aplica cuando tienes una etiqueta “dentro” de otra. Esto te habilita a agregar atributos específicos a “hijos”, sin alterar los del “padre”.

**Class:** generalmente se utiliza para darle estilos a cierta parte del código.

**Atributo ID:** suele usarse para nombrar porciones de código y sectores, como por ejemplo cuando quieres nombrar distintas secciones.

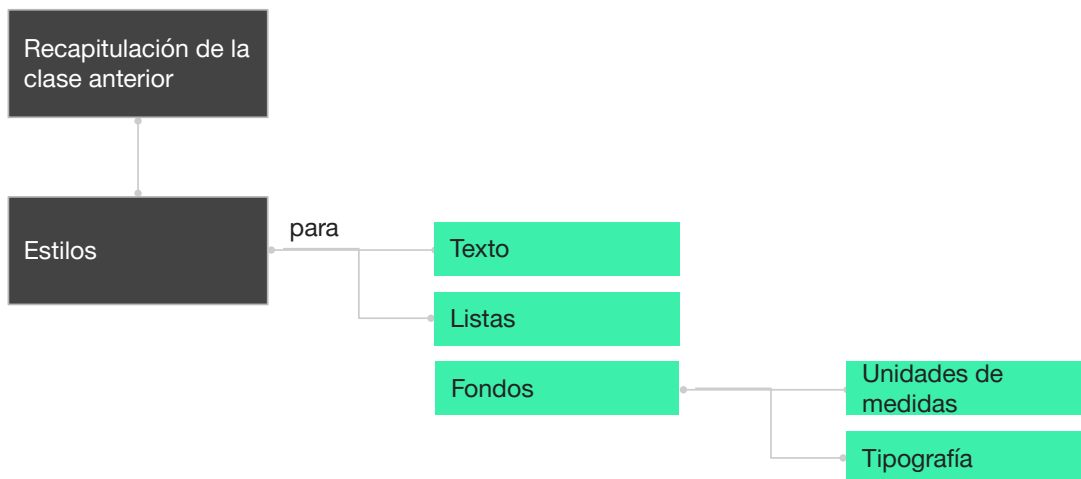
**CODER HOUSE**

# MAPA DE CONCEPTOS

**CODER HOUSE**











## MAPA DE CONCEPTOS CLASE 4

¡Para  
recordar!



**CODER HOUSE**

# ***CRONOGRAMA DEL CURSO***

<div>Clase 3</div> <div>Incluyendo CSS a nuestro proyecto</div> <div><div> PRÁCTICAS DE LO VISTO EN CLASE</div><div> ATRIBUTOS</div><div> AGREGANDO CSS A NUESTRO HTML</div></div>	<div>Clase 4</div> <div>CSS - Medidas, colores, fuentes</div> <div><div> PRÁCTICAS DE LO VISTO EN CLASE</div><div> ESTILOS</div><div> TIPOGRAFÍA</div><div> ASIGNANDO ESTILOS</div><div> PRIMERA ENTREGA DEL PROYECTO FINAL</div></div>	<div>Clase 5</div> <div>Box Modeling</div> <div><div> PRÁCTICAS DE LO VISTO EN CLASE</div><div> BOX MODEL</div></div>
---	--	---

**CODER HOUSE**



## ***GUIÓN DE LA CLASE***

Accede al material complementario [aquí](#).

**CODER HOUSE**



# ***Momento de exposición teórica***

**CODER HOUSE**

## ***RECAPITULACIÓN DE LA CLASE ANTERIOR***

**CODER HOUSE**

# CONCEPTOS PREVIOS

Sabemos que existen tres maneras de aplicar CSS a un documento HTML:

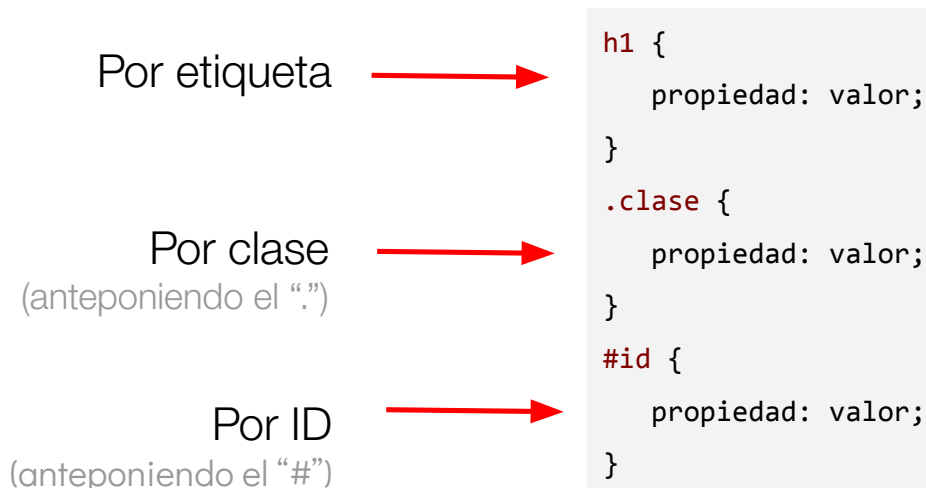
- Hacerlo sobre la etiqueta con el atributo `style=""`
- En el head, insertar la etiqueta `<style>`
- Buscar un archivo externo con un `<link />`

*(Es de las etiquetas que se cierran solas. Requiere `el="stylesheet"` para funcionar. Además un `href=""` con la ruta al archivo.)*

**CODER HOUSE**

## SELECCIÓN DE HTML MEDIANTE CSS

### 3 formas



**CODER HOUSE**

# SELECCIÓN DE HTML MEDIANTE CSS

## Ejemplos



p a

```
<p>  
  Párrafo con <a href="">enlace</a>  
</p>
```



a.foo

```
<a href="" class="foo">Enlace</a>
```



a#foo

```
<a href="" id="foo">Enlace</a>
```



.foo .bar

```
<etiqueta class="foo">  
  <etiqueta class="bar"></etiqueta>  
</etiqueta>
```

**CODER HOUSE**

# SELECCIÓN DE HTML MEDIANTE CSS

## Ejemplos



.foo .foo #foo

```
<etiqueta class="foo">  
  <etiqueta class="foo">  
    <etiqueta id="foo"></etiqueta>  
  </etiqueta>  
</etiqueta>
```



p.foo a.bar

```
<p class="foo">  
  <a href="" class="bar">Enlace</a>  
</p>
```

**CODER HOUSE**

# LOS NOMBRES DE LAS CLASES E IDS

No es posible crear nombres separados por espacios.

La “*joroba de camello*”, permite que se puedan leer de forma más simple palabras compuestas.

`claseDeMaquetacion`

`conBorde`

`productosMasVendidos`



**CODER HOUSE**

**BEM**

**CODER HOUSE**



# BEM

Todos queremos hacer que **nuestro código sea más fácil de leer**. Esto nos ayuda a trabajar más rápidamente y de manera eficiente, y cuando otros trabajen con nosotros podremos mantener claridad y coherencia.

Hoy vamos a cubrir la **metodología BEM**, que nos ayudará a entender estructuras de CSS, y a mejorar las nuestras.

**CODER HOUSE**

# BEM

BEM significa *Modificador de Bloques de Elementos* (*Block Element Modifier*) por sus siglas en inglés. Sugiere una manera estructurada de nombrar tus clases, basada en las propiedades del elemento en cuestión.

BEM tiene como horizonte modularizar lo máximo posible cada uno de los bloques de código dispuesto. Se centra en tres parámetros o variables posibles: bloques (div, section, article, ul, ol, etc.), elementos (a, button, li, span, etc.) y modificadores. Estos últimos se definen de acuerdo a la posterior utilización que haga el desarrollador a cargo.

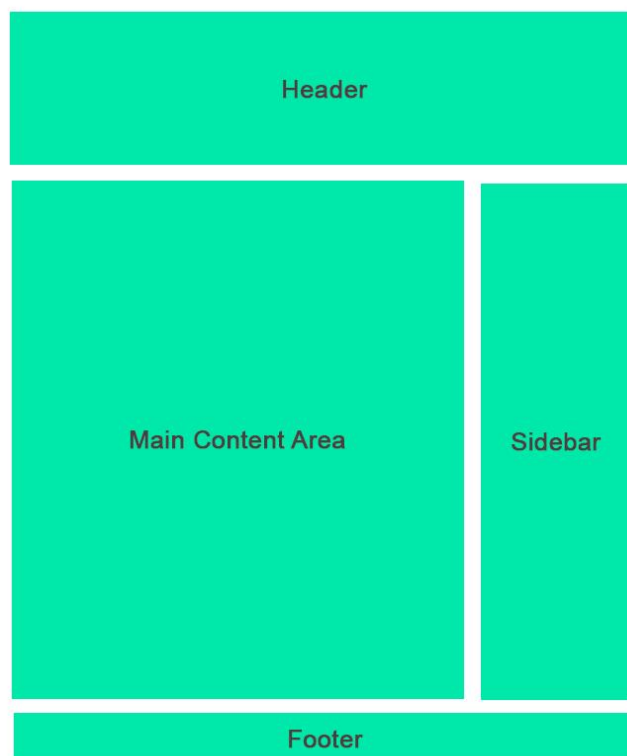
**CODER HOUSE**

# BLOQUE

El *bloque* es un contenedor o contexto donde el elemento se encuentra presente. Piensa como si fueran partes estructurales de código más grandes. Puede que tengas un encabezado, pie de página, una barra lateral y un área de contenido principal; cada uno de estos sería considerado como un bloque. Miremos la imagen a continuación:

**CODER HOUSE**

## BLOQUE



**CODER HOUSE**

# ***BLOQUE***

El bloque de elementos forma la raíz de la clase y siempre irá primero. Solo debes saber que una vez que has definido tu bloque, estarás listo para comenzar a nombrar tus elementos.

```
.block__element {  
  background-color: #FFFFFF;  
}
```

**CODER HOUSE**

# ***BLOQUE***

La doble barra baja te permite visualizar, navegar rápidamente y manipular tu código. Aquí hay algunos ejemplos de cómo funciona la metodología de elementos:

```
.header__logo {}  
.header__tagline {}  
.header__searchbar {}  
.header__navigation {}
```

**CODER HOUSE**

# BLOQUE

## HTML

```
11 <section>
12   <article class="noticia">
13     <!--Bloque contenedor-->
14   </article>
15 </section>
```

## CSS

```
1 .noticia{
2   background: lightgray;
3 }
```

**CODER HOUSE**

El punto es mantener los nombres simples, claros, y precisos.  
No lo pienses demasiado.

Actualizar el nombre de las clases no debería ser un problema cuando encuentras una mejor semántica que funcione (sólo debes tratar de ser consistente, y apegarte a ella).

**CODER HOUSE**

# MODIFICADORES

Cuando nombras una clase, la intención es ayudar a que ese elemento pueda ser repetido, para que no tengas que escribir nuevas clases en otras áreas del sitio si los elementos de estilo son los mismos.

Cuando necesitas modificar el estilo de un elemento específico, puedes usar un modificador. Para lograr esto, añade un doble guión -- luego del elemento (o bloque). Aquí tenemos un corto ejemplo:

```
.block--modifier {}  
.block__element--modifier {}
```

**CODER HOUSE**

# ELEMENTOS

El elemento es una de las piezas que compondrán la estructura de un bloque. El bloque es el todo, y los elementos son las piezas de este bloque.

De acuerdo a la metodología BEM, cada elemento se escribe después del bloque padre, usando dos guiones bajos.

**CODER HOUSE**

# ELEMENTOS

## CSS

```
5  .noticia__titulo{  
6      font-family: 'Times New Roman', serif;  
7  }
```

## HTML

```
11 <section>  
12     <article class="noticia">  
13         <h1 class="noticia__titulo">Título de la noticia</h1>  
14     </article>  
15 </section>
```

**CODER HOUSE**



***¡NUNCA USES SÓLO MODIFICADORES!***

**CODER HOUSE**

Las clases *modificador* siempre deben acompañar a una clase *bloque* o una clase *elemento*, no tiene sentido que aparezcan solas.

## Esto está mal 👎

&lt;button class="button-primary"&gt;&lt;/button&gt;

```
<button class="menu button-primary"></button>
```

**Esto está bien** 👍


&lt;button class="button button-primary"&gt;&lt;/button&gt;

```
<button class="menu button menu button-primary"></button>
```

***CODER HOUSE***

## MODIFICADORES

# CSS

```
9      .noticia--destacada{
10         |      background:  dimgray;
11     }
12     .noticia__titulo--uppercase{
13         |      text-transform: uppercase;
14     }
```

# HTML

```
11     <section>
12         <article class="noticia--destacada">
13             <h1 class="noticia_titulo--uppercase">Título de la noticia</h1>
14         </article>
15     </section>
```

***CODER HOUSE***

# ***BEM***



## ***Ventajas***

- Añade especificidad.
- Aumenta la independencia.
- Mejora la herencia múltiple.
- Permite la reutilización.
- Entrega simplicidad.



## ***Desventajas***

- Las convenciones pueden ser muy largas.
- A algunas personas les puede tomar tiempo aprender la metodología.
- El sistema de organización puede ser difícil de implementar en proyectos pequeños.

***CODER HOUSE***

## ***BEM: ¿PARA QUÉ LO USARÍAS?***

- Para simplificar nuestro CSS y conseguir un estilo consistente, por lo que nuestro código será mucho más legible y fácil de mantener.
- Si estamos usando Bootstrap y queremos modificar ciertas clases.
- Cuando trabajamos en equipo y cada miembro tiene una manera distinta de escribir CSS

***CODER HOUSE***



# ***PROPIEDADES CSS***

**CODER HOUSE**

## **RESETEO CSS**

Los reset CSS contienen en su código fuente definiciones para propiedades problemáticas, que los diseñadores necesitan unificar desde un principio.

Por ejemplo, la mayoría de navegadores establece un margen por defecto entre el contenido de la página web y su propia ventana, cuyo valor varía de un navegador a otro.

Para subsanar esa diferencia, los diseñadores y las diseñadoras de sitios webs suelen declarar la siguiente línea al comienzo de sus hojas de estilo:

```
* {  
  Margin:0;  
  padding:0;  
}
```

# RESETEO CSS

Esa única línea indica, con el selector universal de CSS representado por un asterisco, que todos los elementos contenidos en el HTML a los que se aplique, carecerán de márgenes. De esa manera, el diseñador o la diseñadora se verán obligados a declarar luego los márgenes necesarios en el diseño de su página web, en cada uno de los lugares donde se requiera, sin tener que dejar ese aspecto a decisión de ningún navegador, y minimizando las diferencias visuales entre los mismos.

**Atención:** los reset CSS pueden contener esa y otras muchas líneas de código que, en su conjunto, servirán al diseñador/a web para unificar su visualización entre navegadores.

***ESTILO PARA LISTA***

# LIST-STYLE-TYPE

## CSS

```
ol {  
  list-style-type: none;  
}  
ul {  
  list-style-type: none;  
}
```

Aplicando esta propiedad y este valor, vamos a poder eliminar las bullets y los números.

Valores posibles: ([ver aqui](#))

**CODER HOUSE**

# ESTILOS PARA TEXTO

**CODER HOUSE**

# FONT-STYLE

## CSS

```
.normal {  
  font-style: normal;  
}  
  
.italica {  
  font-style: italic;  
}
```

## Se ve así

Texto normal

*Texto en italica*

Valores comunes: normal | italic

**CODER HOUSE**

# FONT-WEIGHT

## CSS

```
.negrita {  
  font-weight: bold;  
}  
  
.normal {  
  font-weight: normal;  
}
```

## Se ve así

**Texto en negrita**

Texto normal

Valores comunes: normal | bold (luego verán, que puede tener otros valores, en números)

**CODER HOUSE**

# FONT-SIZE

## CSS

```
.textoGrande {  
  font-size: 20px;  
}
```

```
.textoRelativo {  
  font-size: 200%;  
}
```

## Se ve así

Texto en 20 px

Texto en 200%

Valores posibles: <medida de longitud> | <porcentaje>

**CODER HOUSE**

# FONT-FAMILY

## CSS

```
.impact {  
  font-family: Impact, sans-serif;  
}  
.comicSans {  
  font-family: "Comic Sans MS",  
  sans-serif;  
}
```

## Se ve así

**Familia Impact**

Familia Comic

Valores posibles: <familia o nombre genérico>

**CODER HOUSE**

# FONT-FAMILY

Cada sistema operativo y navegador interpretan de distinta forma las fuentes predeterminadas.

- **Serif:** «Times New Roman» en Windows, y «Times» en Macintosh (diferente a la de Windows).
- **Sans serif:** «Arial» en Windows, y «Helvetica» en Macintosh.
- **Monospace:** «Courier New» en Windows, «Courier» en Macintosh, y por lo general «VeraSans» o «DejaVuSans» en Linux.

**Nota:** te recomendamos visitar el sitio <https://www.cssfontstack.com/>, para conocer más acerca de cómo funciona cada fuente, en los distintos sistemas operativos.

# FONT-FAMILY



Las **tipografías Serif** son aquellas que llevan remates, es decir, detalles adicionales en los bordes de las letras. El ejemplo por excelencia de Serif es la **Times New Roman**. Son muy usadas en los periódicos impresos, puesto que los detalles de las letras ayudan a seguir la lectura. Úsalas si quieres transmitir clasicismo, formalidad, precisión, tradición, delicadeza y/o refinamiento.

Por el contrario, las **tipografías Sans Serif**, como su nombre indica –sans es sin en francés–, carecen de estos detalles y también son denominadas de palo seco. Algunas de las más conocidas son la **Arial** o la **Calibri**. Se utilizan mucho en entornos digitales, puesto que los detalles son difíciles de plasmar en píxeles. Transmiten fuerza, modernidad, vanguardia, elegancia y actualidad, a los diseños y textos en los que se incluyen.

**Nota:** este texto está escrito en Arial que es una tipografía Sans Serif.

# TEXT-ALIGN

## CSS

```
.centrar {  
  text-align: center;  
}  
  
.aLaDerecha {  
  text-align: right;  
}
```

## Se ve así

texto

texto

Valores posibles: left | right | center | justify

**CODER HOUSE**

# LINE-HEIGHT

## CSS

```
.interlineado {  
  line-height: 1.6;  
}
```

## Se ve así

texto

ejemplo

Valores posibles: none | <número> | <longitud> | <porcentaje>

**CODER HOUSE**

# TEXT-DECORATION

## CSS

```
.subrayado {  
  text-decoration: none;  
}  
  
.tachado {  
  text-decoration: line-through;  
}
```

Se ve así

Enlace

Parrafo

Valores posibles: none | underline | overline | line-through

**CODER HOUSE**

## ***ESTILOS PARA FONDOS***

**CODER HOUSE**



# BACKGROUND-COLOR

## CSS

```
.fondoFuerte {  
  background-color: yellow;  
}
```

## Se ve así

Parrafo fondo amarillo

Valores posibles: [color]

**CODER HOUSE**

# BACKGROUND-IMAGE

## CSS

```
.catsandstars {  
  background-image:  
url("https://mdn.mozillademos.org/  
files/11991/startransparent.gif"),  
url("https://mdn.mozillademos.org/  
files/7693/catfront.png");  
}
```

## Se ve así

Parrafos con gatos  
y estrellas.

Párrafo sin fondo.

Valores posibles: url | none

**CODER HOUSE**

# BACKGROUND-REPEAT

## CSS

```
.ejemplo {  
  background-image:  
url(https://mdn.mozillademos.org/files/12  
005/starsolid.gif),  
url(https://developer.cdn.mozilla.net/med  
ia/redesign/img/favicon32.png);  
  background-repeat: repeat-x,  
                    repeat-y;  
}
```

## Se ve así

Lorem ipsum dolor sit amet,  
consectetur adipiscing elit.  
Aenean id feugiat est.  
Vestibulum eget imperdiet  
dolor. Interdum et malesuada  
fames ac ante ipsum primis in  
faucibus.

Valores posibles: repeat | repeat-x | repeat-y | no-repeat | space | round ([ver ejemplos](#))

**CODER HOUSE**

# BACKGROUND-POSITION

## CSS

```
.ejemplo {  
  background-image:  
url("https://mdn.mozillademos.org/fil  
es/12005/starsolid.gif");  
  background-repeat: no-repeat;  
  background-position: right center;  
}
```

## Se ve así

Lorem ipsum dolor sit amet,  
consectetur adipiscing elit.  
Aenean id feugiat est.  
Vestibulum eget imperdiet  
dolor. Interdum et malesuada  
fames ac ante ipsum primis in  
faucibus.

Valores posibles: posicionX posicionY ([ver ejemplos](#))

**CODER HOUSE**

# BACKGROUND-SIZE

## CSS

```
.ejemplo {  
  background-image:  
url("https://mdn.mozillademos.org/files/12005/starsolid.gif");  
  background-repeat: no-repeat;  
  background-size: cover;  
}
```

## Se ve así

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean id feugiat est. Vestibulum eget imperdiet dolor. Interdum et malesuada fames ac ante ipsum primis in faucibus.

Valores posibles: [ancho] | [alto] | cover | contain ([ver ejemplos](#))

**CODER HOUSE**

# UNIDADES DE MEDIDAS

**CODER HOUSE**

# UNIDADES DE MEDIDAS

Hay una amplia variedad de absolutas y relativas, pero nos centraremos en:

## Absolutas

- **Px (pixels):** es la unidad que usan las pantallas.

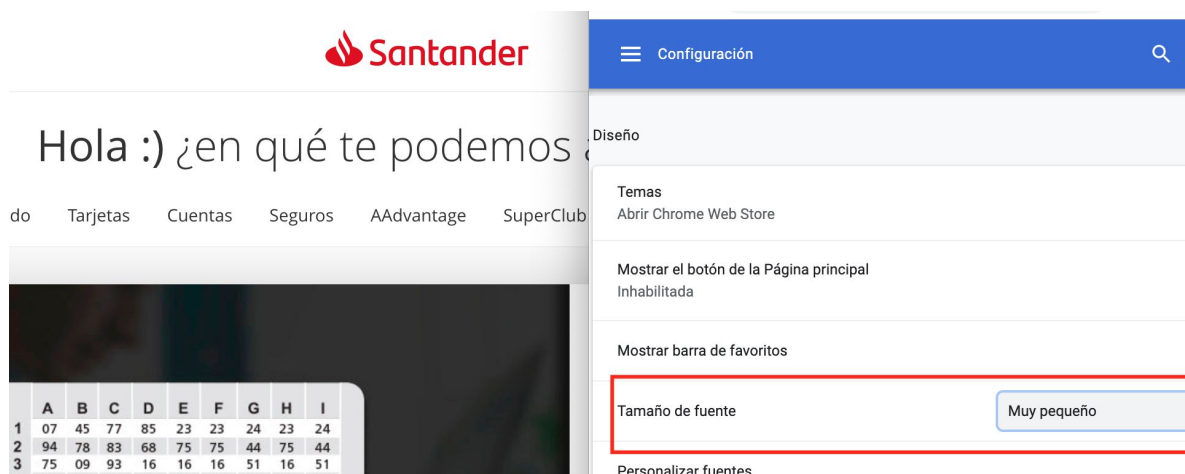
## Relativas

- **Rem:** relativa a la configuración de tamaño de la raíz (etiqueta html).
- **Porcentaje:** tomando en cuenta que 16px es 100%.
- **Viewport:** se utilizan para layouts responsivos (más adelante).

**CODER HOUSE**

# UNIDADES DE MEDIDAS

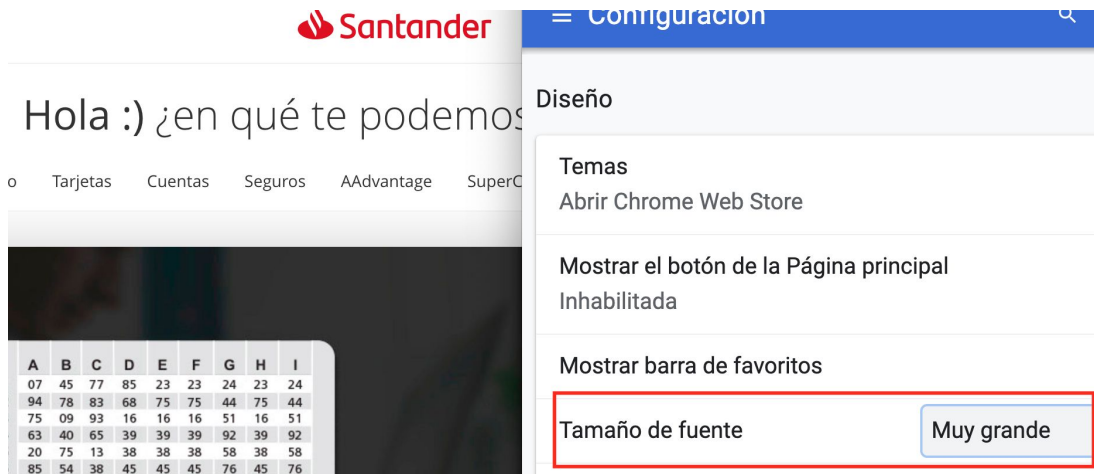
El problema con los píxeles.



**CODER HOUSE**

# UNIDADES DE MEDIDAS

¿Encuentras las diferencias? Exacto, no cambió nada



Santander

Hola :) ¿en qué te podemos

o Tarjetas Cuentas Seguros AAdvantage SuperC

Configuración

Diseño

Temas

Abrir Chrome Web Store

Mostrar el botón de la Página principal

Inhabilitada

Mostrar barra de favoritos

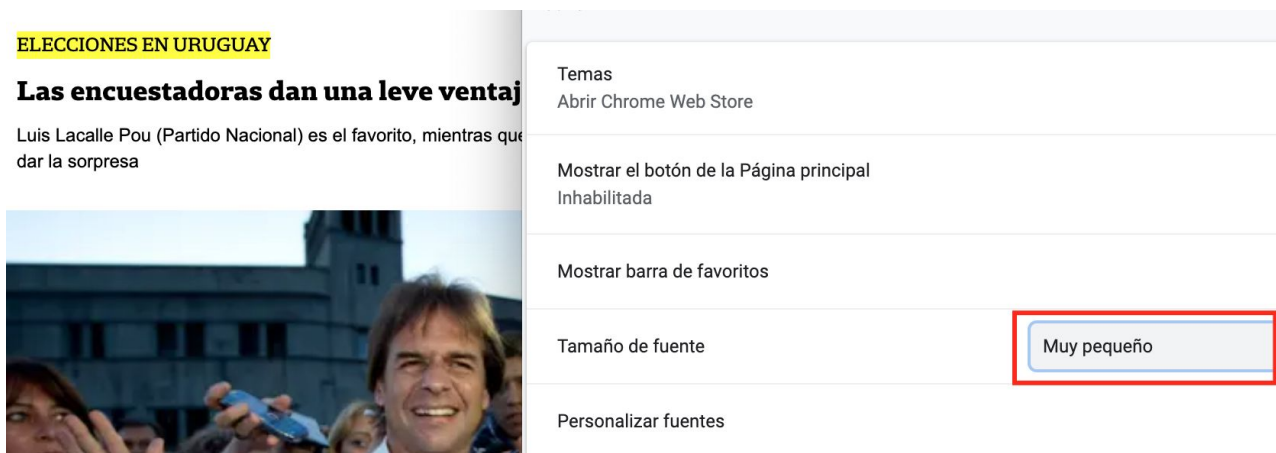
Tamaño de fuente

Muy grande

**CODER HOUSE**

# UNIDADES DE MEDIDAS

Ejemplo que tiene la solución:



ELECCIONES EN URUGUAY

**Las encuestadoras dan una leve ventaja**

Luis Lacalle Pou (Partido Nacional) es el favorito, mientras que...

dar la sorpresa

Temas

Abrir Chrome Web Store

Mostrar el botón de la Página principal

Inhabilitada

Mostrar barra de favoritos

Tamaño de fuente

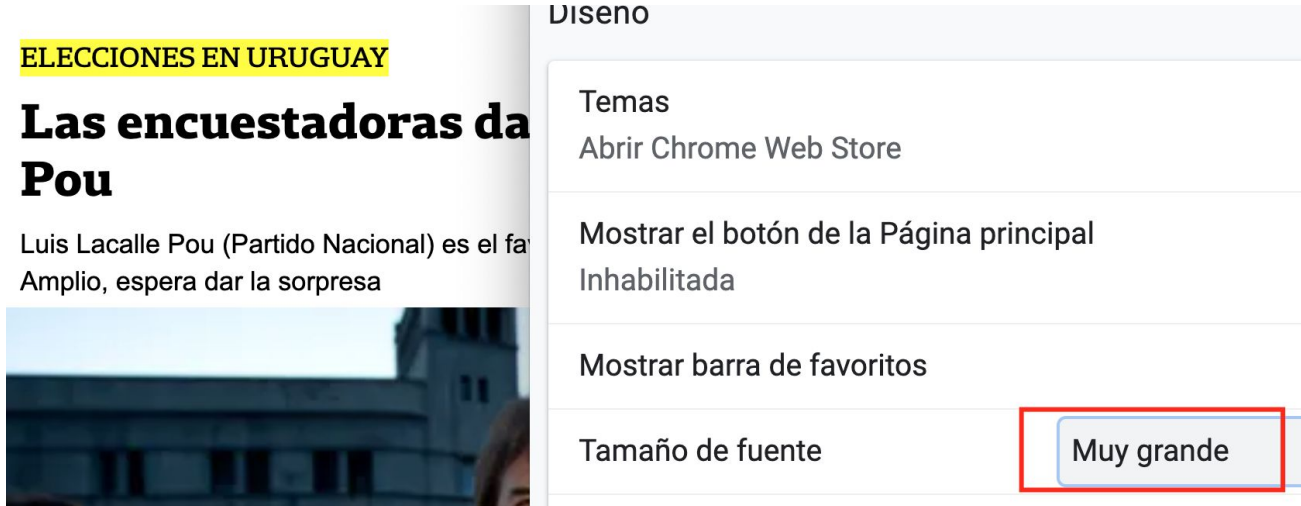
Muy pequeño

Personalizar fuentes

**CODER HOUSE**

# UNIDADES DE MEDIDAS

¿Ahora sí está diferente, no?



**CODER HOUSE**

# UNIDADES DE MEDIDAS

Ahora veamos qué medida es más conveniente para los textos.

```
html { /* etiqueta raíz */
  font-size: 62.5%;
}
p {
  font-size: 2rem; /* 20px */
}
```

Texto simulando  
20px

62.5%, hace que en vez de que 16px sea el valor a tomar en cuenta para calcular las unidades relativas, se use 10px.

**CODER HOUSE**

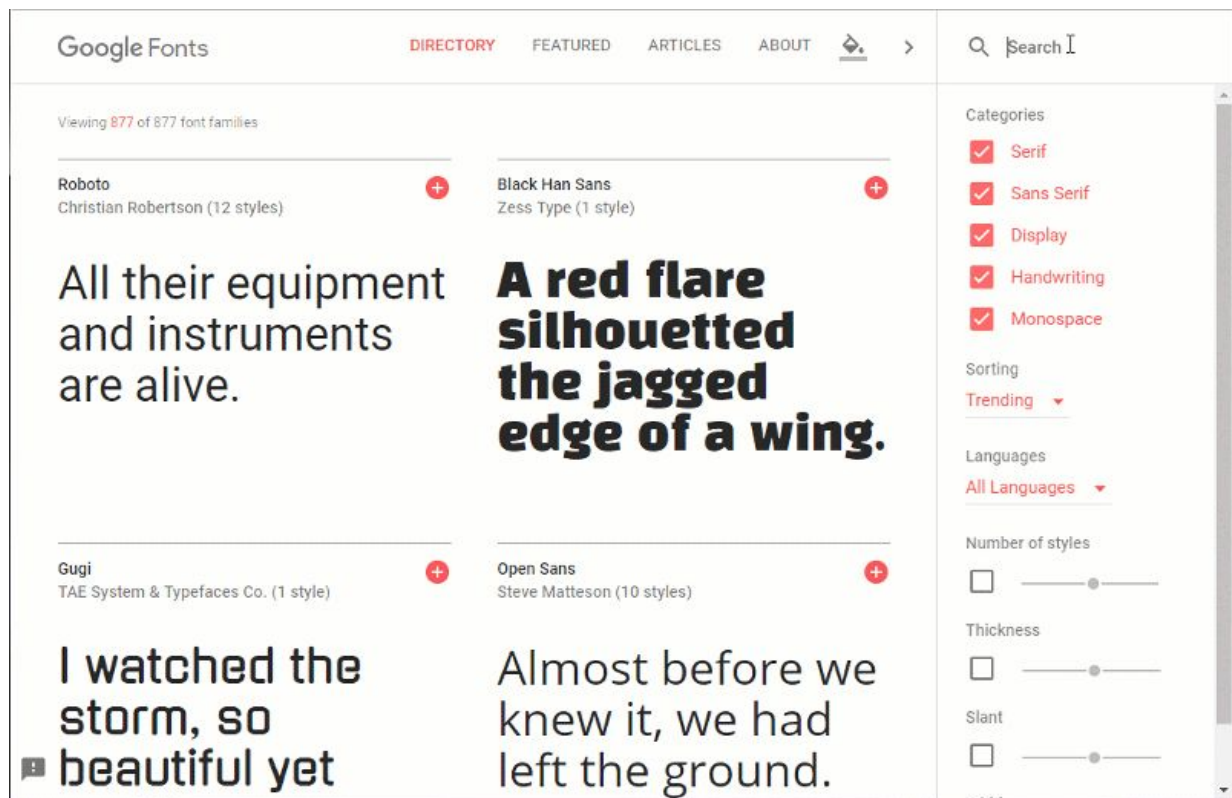
# ***TIPOGRAFÍA***

***CODER HOUSE***

## ***TIPOGRAFÍA WEB***

Habíamos visto que usando “*font-family*”, es posible agregar algunas limitadas fuentes, pero... podemos usar muchísimas opciones de fuentes con “**Google Fonts**”.

***CODER HOUSE***



**CODER HOUSE**

# TIPOGRAFÍA WEB

## CSS

```
h1 {  
  font-family: 'Roboto',  
  sans-serif;  
}
```

## HTML

```
<head>  
  <link  
    href="https://fonts.googleapis.com  
/css?family=Roboto&display=swap"  
    rel="stylesheet">  
  <title>Document</title>  
</head>
```

Ver [Google Fonts](#)

**CODER HOUSE**



# iBONUS!

EMBED

CUSTOMIZE

Load Time: Moderate

## Roboto

- ☒ thin 100
- ☐ *thin 100 Italic*
- ☒ light 300
- ☐ *light 300 Italic*
- ☒ regular 400
- ☐ *regular 400 Italic*
- ☒ medium 500
- ☐ *medium 500 Italic*
- ☒ bold 700

**CODER HOUSE**

# iBONUS!

## CSS

```
h1 {  
  font-family: 'Roboto',  
  sans-serif;  
  font-weight: 100;  
}
```

## HTML

```
<link  
  href="https://fonts.googleapis.com  
/css?family=Roboto:100,300,400,500  
,700&display=swap"  
  rel="stylesheet">
```

Ver [Google Fonts](#)

**CODER HOUSE**

# ***¡BONUS!***

Se ve así porque 100 es lo más delgado posible:

## Titulo

***CODER HOUSE***



# ***BREAK***

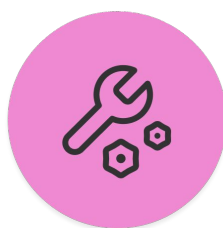
**¡5/10 MINUTOS Y VOLVEMOS!**



# ***¡VAMOS A PRACTICAR LO VISTO!***

Momento de consolidación de aprendizajes

***CODER HOUSE***



## ***ESTILOS***

- Crea un archivo HTML y otro CSS.
- Agrega un estilo de tipografía a un archivo.

***CODER HOUSE***



## ¡A PRACTICAR!

Crea un archivo HTML con un elemento párrafo dentro de él, con texto Lorem Ipsum. Además, crea un archivo CSS y agrégale estilo al párrafo con los siguientes valores:

- Color de texto: #0033ff
- Indexar una google font a elección.
- Espacio entre renglones: 15px
- Tamaño de texto: 12px

Cuentas con **15 minutos** para completar la actividad.

**CODER HOUSE**



## ASIGNANDO ESTILOS

Agrega nuevas propiedades en el CSS de tu Proyecto.

**CODER HOUSE**

## ASIGNANDO ESTILOS

**Formato:** archivo HTML y CSS. Debe tener el nombre "Idea+Apellido".  
**Sugerencia:** carpeta en formato zip o rar con el/los archivos html y css.

Desafío  
Complementario



**>> Consigna:** asigna estilos vistos en esta clase al proyecto final, en función de lo que tenías en mente gracias al boceto.

**>>Aspectos a incluir en el entregable:**

Incluir de acuerdo a tu proyecto algunas de las propiedades vistas:

- Texto: font-style, font-weight, font-size, font-family, text-align, line-height y text-decoration.
- Listas: list-style-type, list-style-position y/o list-style.
- Fondos: background-color, background-image, background-repeat, background-position y/o background-size.
- Tipografía de [Google Fonts](#).

**>>Ejemplo:**

[Carpeta comprimida con los archivos](#)

**CODER HOUSE**

## SUGERENCIA: ¡HAZ ESTE DESAFÍO COMPLEMENTARIO!

Hacer el desafío "**Asignando estilos**" es importante porque podrás desarrollar código. Además, te ayudará a ver cómo traducir todo tu diseño al lenguaje de las computadoras.

De esta manera, aprenderás a poner en práctica propiedades que se usan todo el tiempo en el mundo del trabajo, y a la hora de la pre-entrega y posteriormente la entrega de tu proyecto final, aportarán valor agregado si las puedes manejar de manera correcta.



# PRIMERA ENTREGA DEL PROYECTO FINAL

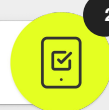
**CODER HOUSE**

## PRIMERA ENTREGA DEL PROYECTO FINAL

**Formato:** aclarar tipo de archivo en que se espera el desafío. Debe tener el nombre "Idea+Apellido".

**Sugerencia:** activar comentarios en el archivo.

Proyecto  
Final



### >>Objetivos Generales:

1. Lograr armar un primer prototipo de la web, su estructura en HTML y estilo inicial.

### >>Objetivos Específicos:

1. **Prototipar la web** para tener una idea clara del resultado al que quiere llegar.
2. **Maquetar la web:** El estudiante deberá utilizar los tags, en especial los semánticos, para desde el código describir la estructura de la web.
3. **Crear un estilo inicial:** El estudiante deberá comenzar a darle estilo básico a su web definiendo colores. El documento debe estar linkeado en las páginas del proyecto.

### >>Se debe entregar:

- Sketch.
- Estructura de la web en HTML: página principal (index.html), cada una de las secciones y enlaces entre secciones.
- Estilo básico: agregar al CSS colores pertinentes al diseño inicial.

**CODER HOUSE**