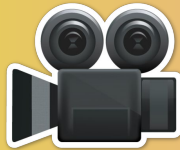


***RECUERDA PONER A GRABAR LA
CLASE***





¿DUDAS DEL ON-BOARDING?

[MIRALO AQUI](#)



Clase 03. DESARROLLO WEB

INCLUYENDO CSS A NUESTRO PROYECTO



OBJETIVOS DE LA CLASE

- Comprender la sintaxis de CSS
- Incluir CSS en nuestro Proyecto
- Conocer el uso de medidas, colores, fuentes y fondos en CSS

GLOSARIO:

Clase 2

Listas: HTML permite agrupar elementos que tienen más significado de forma conjunta. Aunque cada palabra por separado tiene sentido, de forma conjunta constituyen el menú de navegación de la página, por lo que su significado conjunto es mayor que por separado. Esto se denomina listas.

Tablas: son un conjunto de celdas organizadas, dentro del cual es posible alojar distintos contenidos. Sirven para representar información tabulada, en filas y columnas.

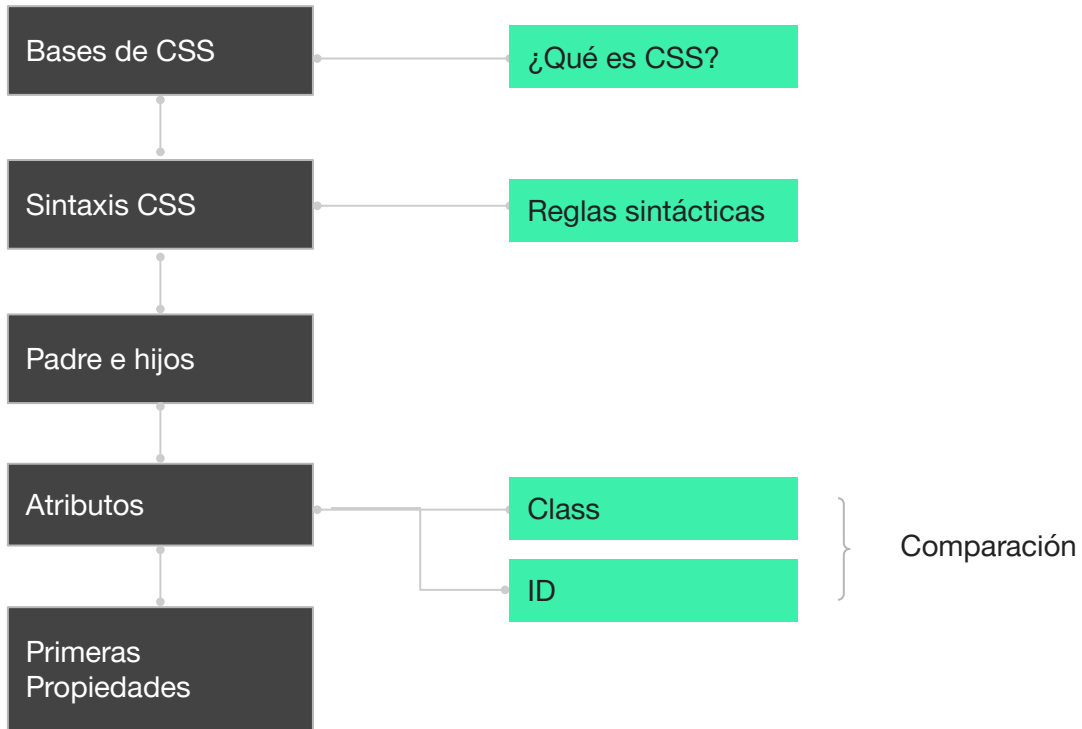
Formularios: son etiquetas donde el usuario ingresará o seleccionará valores, que serán enviados a un archivo encargado de procesar la información.

Enlaces: también conocidos como links o anchors, se utilizan para relacionar partes del mismo documento. Por defecto, se visualizan azules y subrayados.

MAPA DE CONCEPTOS

MAPA DE CONCEPTOS CLASE 3

¡Para
recordar!



CRONOGRAMA DEL CURSO

Clase 3



Incluyendo CSS a nuestro proyecto



PRÁCTICAS DE LO VISTO EN CLASE



ATRIBUTOS

Clase 4



CSS + Box Modeling



PRÁCTICAS DE LO VISTO EN CLASE



ESTILOS



TIPOGRAFÍA



ASIGNANDO ESTILOS

Clase 5



Flexbox



PRÁCTICAS DE LO VISTO EN CLASE



PRIMERA ENTREGA DEL PROYECTO FINAL



GUIÓN DE LA CLASE

Accede al material complementario [aquí](#).



BASES DE CSS

PREMISAS

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

Bienvenidos a Diseño Web

Que la fuerza te acompañe!

- Inicio
- Blog
- Contactos

Sobre el profesor

El profesor explica, hace y ustedes practican

Sobre ustedes

Aprenderan a diseñar una página web



BIENVENIDOS A DISEÑO WEB



Que la fuerza te acompañe!

SOBRE EL PROFESOR



El profesor explica, hace y ustedes practican

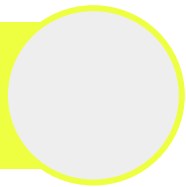
SOBRE USTEDES



Aprenderan a diseñar una página web

SINTAXIS DE CSS

SINTAXIS

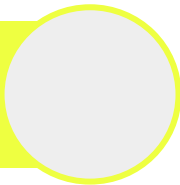


```
selector {  
  propiedad1: valor;  
  propiedad2: valor;  
}
```

Ejemplo

```
h1 {  
  color: red;  
}
```

REGLAS SINTÁCTICAS



- Cada declaración CSS está formada por un juego de pares **propiedad: valor;**
- No se ve afectado por el espacio en blanco. Las propiedades se pueden escribir de corrido o una debajo de la otra.
- Siempre que la propiedad represente un número, el valor debe indicar en qué unidad se expresa.

PADRE E HIJOS

PADRE E HIJOS



Cuando tienes una etiqueta “dentro” de otra, lo que haces es aplicar el concepto de padres e hijos.

En este caso, **section** es padre de **article** y, a su vez, **article** es padre del **h2** y del **p**.

```
<section>
  <article>
    <h2> Título </h2>
    <p> Lorem ipsum dolor sit amet,
consectetur adipiscing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna
aliqua. Ut enim ad minim veniam, quis nostrud
exercitation ullamco laboris nisi ut aliquip ex
ea commodo consequat.
    </p>
  </article>
</section>
```

PADRE E HIJOS



Esto habilita a agregar atributos específicos a “hijos”, sin alterar los del “padre”.
Un padre puede tener muchos hijos, y todos ellos heredan sus características, pudiendo tener también características particulares.

Selector
PADRE

Selector **HIJO**

```
section article {  
  background-color: #cccccc;  
  width: 500px;  
  height: 500px;  
}
```

PADRE E HIJOS



En este caso, se observa la forma correcta de declarar cada estilo. Cuando quieres seleccionar una etiqueta, debes incluir las etiquetas padre/s para que sean más específicas a la hora de aplicar estilos.

```
<section>
  <article>
    <h2> Título </h2>
    <p> Lorem ipsum dolor sit amet,
      consectetur adipiscing elit, sed do
      eiusmod tempor incididunt ut labore et
      dolore magna aliqua. Ut enim ad minim
      veniam, quis nostrud exercitation
      ullamco laboris nisi ut aliquip ex ea
      commodo consequat.
    </p>
  </article>
</section>
```

```
section {
  padding: 50px 30px 20px 60px;
  margin-left: 40px;
}
```

```
section article {
  background-color: #cccccc;
  width: 500px;
  height: 500px;
}
```

```
section article p {
  line-height: 4;
}
```

INSERTAR CSS

INSERTAR CSS EN EL HTML

1

Forma externa

Forma **EXTERNA**: dentro de la etiqueta **<head>**, llamas al archivo CSS que necesites (recuerda el uso de rutas relativas y absolutas).

```
<link rel="stylesheet" href="archivo.css" />
```

INSERTAR CSS EN EL HTML

2

Forma interna

Forma **INTERNA**: es recomendable que esté dentro de la etiqueta **<head>**. Puede estar en **<body>**, pero sería más desprolijo.

```
<style>  
    /* comentario de CSS, dentro de esta etiqueta, va el  
    codigo CSS, */  
</style>
```

INSERTAR CSS EN EL HTML

3

Forma interna

Otra forma **INTERNA**, muy poco recomendable, consiste en usar para “parches” específicos, o pruebas. Se hace difícil mantenerlo.

```
<h1>Un encabezado sin formato</h1>  
<h2 style="CODIGO CSS">H2 con formato CSS</h2>  
  
<p>Párrafo sin formatear</p>  
<p style="CODIGO CSS">Párrafo formateado</p>  
<p>Otro párrafo sin formatear</p>
```

3

CLASS

CLASS

👉 Generalmente se utiliza para darle estilos a cierta parte del código. Por ejemplo, si quieres que una imagen tenga bordes, y que además sean redondeados.

CLASS DESDE CSS



Desde CSS, **puedes usar los nombres que quieras**, siempre y cuando empiecen con **LETRAS**, y pongas un “.” adelante. Lo recomendable es poner un nombre que haga referencias a los estilos que tendrá. Por ejemplo:

```
.bordesRedondeados {  
  /* codigo CSS */  
}
```

HTML: ATRIBUTO CLASS=""



En el HTML, para aplicar una clase debes usar el atributo **“class”**, y luego colocar en el **valor** el **nombre de la clase** (que has especificado en CSS).

```
<img src="" class="bordesRedondeados" />
```

MÁS DE UNA CLASS



Puedes aplicar **más de una clase** a cada etiqueta separada por un espacio. De esta manera, podrás tener estilos diferenciados para cada clase.

```
<img src="" class="bordesRedondeados imgChica"/>
```

ATRIBUTO ID

ID

👉 Generalmente se usa para nombrar porciones de código y sectores, como por ejemplo cuando quieres nombrar distintas secciones.

👉 Es posible ponerle ID a cualquier elemento HTML para darle un "nombre". Y así como el ID, todos los elementos también aceptan el atributo `class=""`.

- Dicha clase se utiliza cuando quieres aplicar el mismo estilo a más de un elemento, y la búsqueda por etiqueta no sirve para lograrlo.
- 👉 No necesitas escribir varias veces el mismo CSS, ni repetir el ID.



Desde CSS, **puedes usar los nombres que quieras**, siempre y cuando empiecen con **LETRAS**, y pongas un “#” adelante. Lo recomendable es poner un nombre que haga referencias a los estilos que tendrá. Por ejemplo:

```
#productos {  
  /* codigo CSS */  
}
```

HTML: ATRIBUTO ID=""



Para aplicar un ID en el HTML, debes usar el atributo “id”, y luego en el valor el nombre del ID (que has especificado en CSS). Por ejemplo:

```
<section id="productos">  
  
</section>
```


COMPARACIÓN CLASS VS. ID

	¿Se puede reutilizar su nombre en el HTML?	¿Se puede usar varias veces en un atributo en el HTML?	¿Cuándo lo uso?
ID	NO	NO	Nombrar secciones, divisiones de código
CLASS	SI	SI	Especificar diseño aparte del código
Ejemplo ID	id="productos" id="productos2"		<section id="productos">
Ejemplo CLASS	class="bordes" class="bordes"	class="bordes destacado"	<p class="destacado">

EJEMPLO

HTML:

```
<section id= "prod">
  <article class= "rojo">
  </article>
  <article id= "prod">
  </article>
</section>
```

HTML:

```
<section id= "prod">
  <article class= "rojo">
  </article>
  <article class= "rojo">
  </article>
</section>
```

Tanto **ID** como **Class** pueden ser utilizadas dentro del html en diferentes etiquetas. Sin embargo, **los nombres otorgados a las clases se pueden repetir**, mientras que utilizados en **los IDs no**.

HERENCIA Y CASCADA

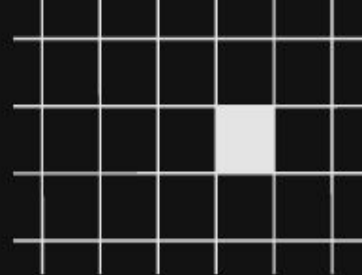
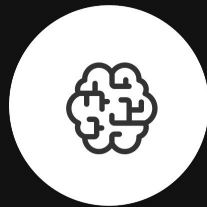
HERENCIA



En general, estas propiedades son intuitivas. Por ejemplo, podrás heredar de un elemento padre el tamaño de letra y color de la misma, *a menos que el elemento hijo tenga otros estilos aplicados*. Puedes ver más al respecto [aquí](#).

```
div {  
  color: red;  
}
```

```
<div>  
  <p>Este párrafo quedará  
  en rojo, por herencia</p>  
</div>
```

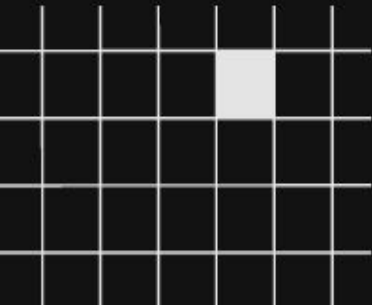


¡PARA PENSAR!- CASCADA

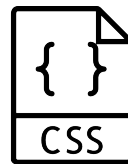
El navegador lee de arriba hacia abajo (forma de cascada)

¿De qué color crees que se aplicará al párrafo (p) al ver el siguiente código?

```
p {  
    color: red;  
}  
  
p {  
    color: green;  
}
```



EJEMPLO



HTML:

```
<ul>
  <li class="rojo">Item
R1</li>
  <li class="rojo">Item
R2</li>
  <li class="rojo">Item
R3</li>
  <li class="rojo">Item
R4</li>
</ul>
```

CSS:

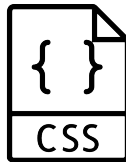
```
.rojo{ color: red; }
.azul{ color: blue; }
```

HTML:

```
<ul>
  <li>Item R1</li>
  <li>Item R2</li>
  <li>Item R3</li>
  <li>Item R4</li>
</ul>
<ol>
  <li>Item A1</li>
  <li>Item A2</li>
</ol>
```

CSS:

```
ul li{ color: red; }
ol li{ color: blue; }
```



PRECEDENCIA DE DECLARACIONES

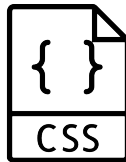
Cuando reglas distintas apuntan al mismo objeto:

- Si son **propiedades distintas**, se suman (se combinan).
- Si tienen **alguna propiedad repetida**, sólo una queda.

Esto es lo que se denomina *precedencia*.

- **ID** pisa cualquier otra regla.
- **Class** sobrescribe las reglas de etiqueta, pero no las de **ID**.
- **Etiquetas** tienen la menor precedencia.

ID > Class > Etiquetas



ESTILOS INLINE

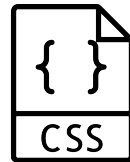
Si utilizas estilos inline, sobrescribirán cualquier estilo de las páginas externas de CSS.

Se podría decir que los estilos inline son los que tienen una mayor especificidad, por lo tanto:

 **no es recomendable utilizarlos en tu página.**

```
<p style= "color: red">Párrafo rojo</p>
```

IMPORTANCIA DE LOS SELECTORES



En este gráfico se resume cuán **importante** es cada selector:



Estilo aplicado
a la **Etiqueta**.



Estilo aplicado
a la **Class**.

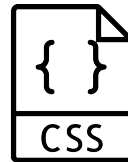


Estilo aplicado
al **ID**.



Estilo aplicado
al **Inline**.

!important;



- Si tienes 3 reglas CSS, es poco probable que “choquen”, pero en un CSS extenso es más común.
- La declaración ***!important;*** corta la precedencia. Se escribe después del valor de la propiedad CSS que se quiere convertir en la más importante. Se utiliza un ***!important;*** por cada valor a pisar.

Si necesitas más de 5 ***!important;*** en todo tu CSS, algo estás haciendo mal.



RESET

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.

RESETEO CSS

- 👉 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;  
}
```

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.

PRIMERAS PROPIEDADES

PROPIEDAD: COLOR

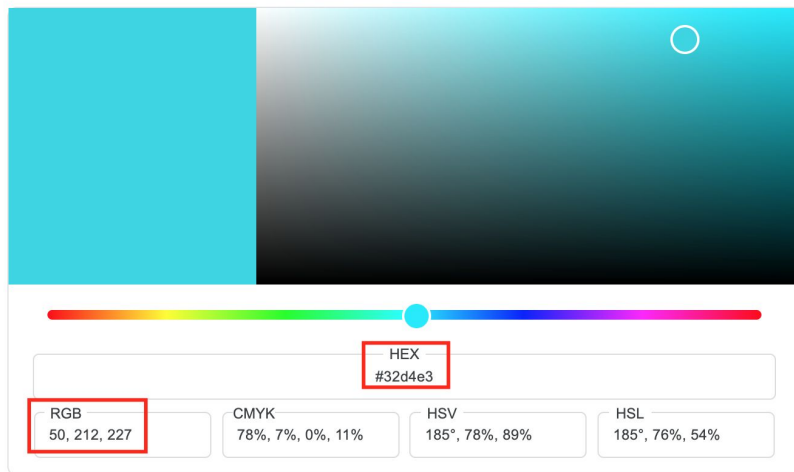
👉 Mediante esta propiedad, podrás agregar color a los textos de tu sitio

pero... **¿cómo se eligen los colores?** 😞

PROPIEDAD: COLOR

1

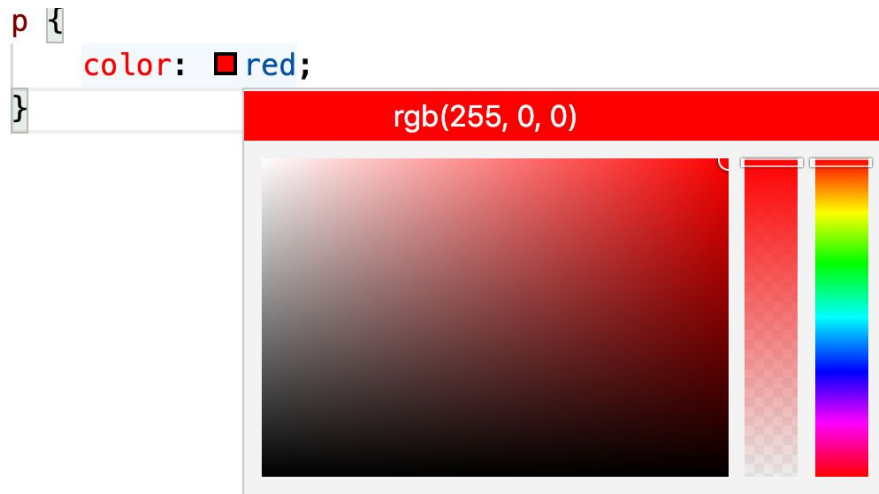
Desde Google, puedes buscar “[color picker](#)”
(alternativa [w3schools](#)).



PROPIEDAD: COLOR

2

Desde Visual Studio Code, simplemente te “paras” sobre el color. Por ejemplo, escribe “red” y haz la prueba:



TIPOS DE VALORES PARA COLOR

Existen distintos valores, pero **nos centraremos en 3:**

- Por nombre del color (ej: red).
- Hexadecimal (ej: #ffffff).
- RGB (por ejemplo: 50, 212, 227). Si agregas un valor más, puedes manejar su opacidad. (red, green, blue) cada color permite hasta 256 valores.

ESTILO 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 aquí](#))

ESTILO TEXTO

FONT-STYLE

CSS

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

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

Se ve así

Texto normal

Texto en italica

Valores comunes: normal | italic

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)

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>

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>

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

LINE-HEIGHT

CSS

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

Se ve así

texto
ejemplo

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

TEXT-DECORATION

CSS

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

Se ve así

Enlace

Parrafo

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



BREAK

¡5/10 MINUTOS Y VOLVEMOS!



PILAR
MUNICIPIO

conectados
con las *oportunidades*

CODERHOUSE

ESTILO BACKGROUND

BACKGROUND-COLOR

CSS

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

Se ve así

Parrafo fondo amarillo

Valores posibles: [color]

BACKGROUND-IMAGE

CSS

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

Valores posibles: url | none

Se ve así



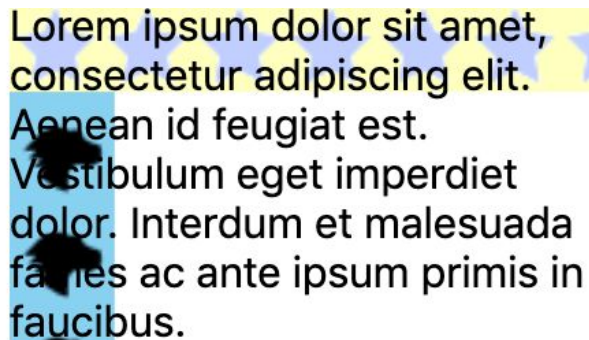
Párrafo sin fondo.

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](#))

BACKGROUND-POSITION

CSS

```
.ejemplo {  
    background-image:  
url("https://mdn.mozillademos.org/files/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](#))

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](#))

UNIDADES DE MEDIDA

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 / em:** 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).

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.

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).

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.

TIPOGRAFÍA

TIPOGRAFÍA LOCAL

Habíamos visto que usando “*font-family*”, es posible agregar algunas limitadas fuentes, pero... podemos usar muchísimas opciones de fuentes si las descargamos y las agregamos a nuestro directorio raíz.

TIPOGRAFÍA LOCAL

CSS

```
@font-face {  
  font-family: "Mystery Quest";  
  src: url("mystery-quest.ttf");  
}  
  
p {  
  font-family: "Mystery Quest", cursive;  
}
```

El valor de la propiedad src debe indicar en qué parte de nuestro directorio raíz guardamos nuestra tipografía post descarga.

TIPOGRAFÍA WEB

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

TIPOGRAFÍA WEB

CSS

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

Ver [Google Fonts](#)

HTML

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

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

iBONUS!

CSS

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

HTML

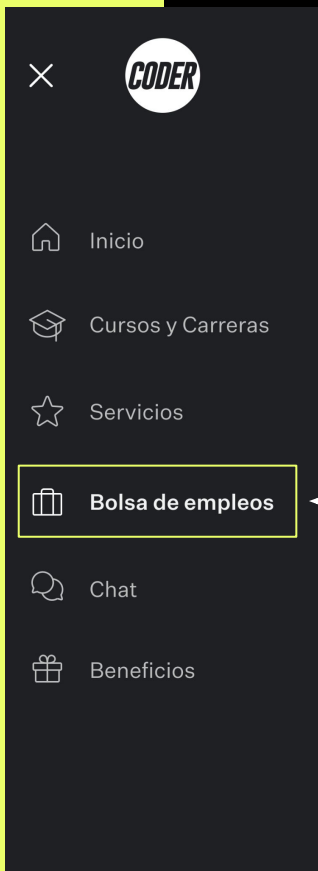
```
<link  
href="https://fonts.googleapis.com  
/css?family=Roboto:100,300,400,500  
,700&display=swap"  
rel="stylesheet">
```

Ver [Google Fonts](#)



¿PREGUNTAS?

#CoderTip: Ingresa al [siguiente link](#) y revisa el material interactivo que preparamos sobre **Preguntas Frecuentes**, estamos seguros de que allí encontrarás algunas respuestas.



Nuevo

¡Lanzamos la Bolsa de Empleos!

Un espacio para seguir **potenciando tu carrera** y que tengas más **oportunidades de inserción laboral**.

Podrás encontrar la **Bolsa de Empleos** en el menú izquierdo de la plataforma.

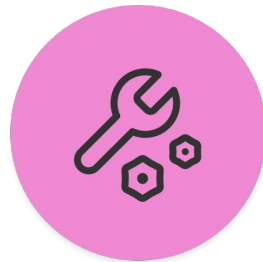
Te invitamos a conocerla y ¡postularte a tu futuro trabajo!

Conócela

Ejemplo
en vivo



¡VAMOS A PRACTICAR LO VISTO!



ATRIBUTOS

Agrega atributos a los archivos.

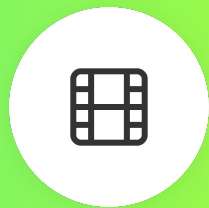


¡A PRACTICAR!

Al archivo HTML creado previamente, agrega un div con un párrafo lleno de texto, y asígnale la clase “desafio1”. Además, crea un archivo CSS con una clase llamada “desafio1”; y asígnale la propiedad color con valor naranja (orange). Tienes 15 minutos para completar la actividad.

¿PREGUNTAS?





***¿QUIERES SABER MÁS? TE DEJAMOS
MATERIAL AMPLIADO DE LA CLASE***

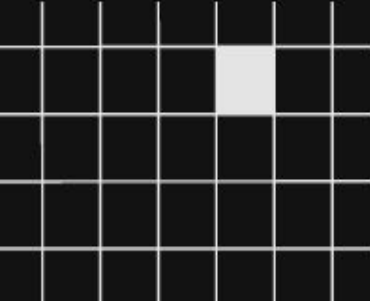


- [Patrones sutiles](#) | **Toptal**
- [Recursos de Dominio Público](#) | **Internet Archive**
- [Jardin Zen CSS](#) | **CSS Zen Garden**



¡MUCHAS GRACIAS!

Resumen de lo visto en clase hoy:

- Multimedia con HTML.
 - CSS: Sintaxis, Reglas, Atributos class & ID.
- 



OPINA Y VALORA ESTA CLASE