



¿DUDAS DEL ON-BOARDING?

[MIRALO AQUI](#)

CODER HOUSE

Clase 02. DESARROLLO WEB

PRIMEROS PASOS CON HTML



OBJETIVOS DE LA CLASE

- Comprender la sintaxis de HTML.
- Conocer las diferentes etiquetas y el uso de cada una.
- Dominar el concepto de web semántica, y las etiquetas HTML5 de estructura.

CODER HOUSE

GLOSARIO:

Clase 1

Sketch: es un dibujo rápido o bosquejo guía, que reproduce de manera muy sencilla un concepto, una idea o generalidad de un proyecto.

Wireframe: es la representación estática, en baja calidad, de un diseño.

Mockup: es la representación estática de un diseño, en calidad media o alta.

Prototipo: es la representación navegable del producto final.

HTML: es un "lenguaje" de marcado de etiquetas, que permite crear documentos para web.

Doctype: cuando escribes tu documento HTML, lo primero que debes hacer es escribir el DOCTYPE, el cual declara el tipo de documento. Es decir, sirve para indicar que tu documento está escrito siguiendo la estructura determinada por un DTD concreto. Un DTD es la definición del tipo de documento.

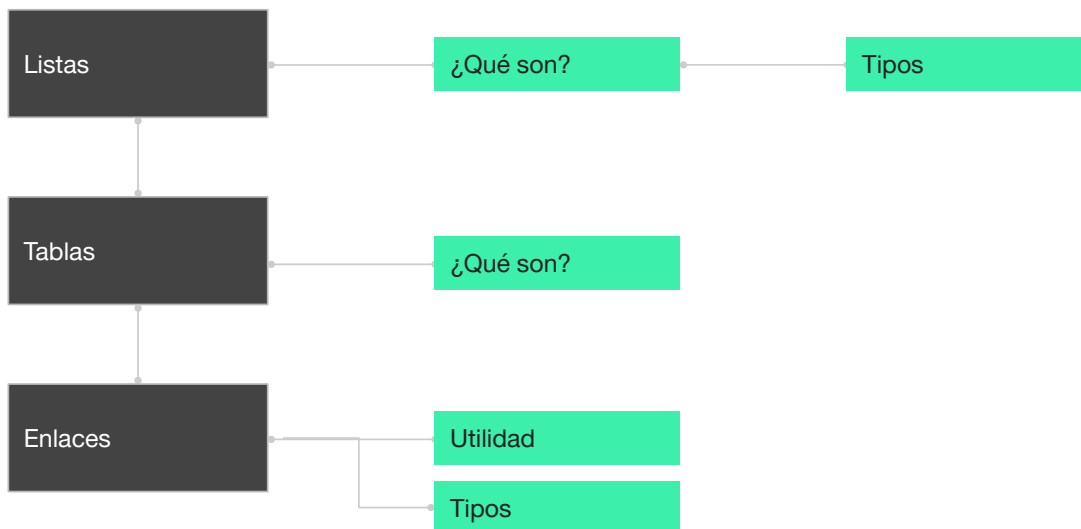
CODER HOUSE

MAPA DE CONCEPTOS

CODER HOUSE

MAPA DE CONCEPTOS CLASE 2

¡Para
recordar!



CODER HOUSE

CRONOGRAMA DEL CURSO

<div>Clase 1</div> <div>Prototipado y conceptos básicos de HTML</div> <div><div>SKETCH</div><div>PRÁCTICA DE LO VISTO EN CLASE</div><div>NUEVO DOCUMENTO</div><div>WIREFRAME Y ESQUELETO HTML</div></div>	<div>Clase 2</div> <div>Primeros pasos con HTML</div> <div><div>LISTAS</div><div>PRÁCTICAS DE LO VISTO EN CLASE</div><div>FORMULARIOS</div><div>ESTRUCTURA HTML DEL PROYECTO</div></div>	<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>
--	---	--

CODER HOUSE



GUIÓN DE LA CLASE

Accede al material complementario [aquí](#).

CODER HOUSE



Momento de exposición

CODER HOUSE

LISTAS

CODER HOUSE

LISTAS

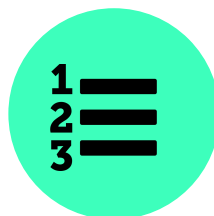
HTML permite agrupar elementos que tienen más significado de forma conjunta. El menú de navegación de un sitio web, por ejemplo, está formado por un grupo de palabras. Aunque cada palabra por separado tiene sentido, de forma conjunta constituyen el menú de navegación de la página, por lo que su **significado conjunto** es mayor que por separado. Esto se denomina **listas**.

CODER HOUSE

TIPOS DE LISTAS



Listas no ordenadas



Listas ordenadas



Listas de definición

Ejemplos: paso a paso de un procedimiento (como una receta de cocina), características de una persona, galería de imágenes, el menú de una página web, entre otros.

CODER HOUSE

¿VIÑETAS O NÚMEROS?

- Las **listas numéricas** establecen un orden en la lectura de sus ítems.
- Las **listas de viñetas** no representan ningún orden o importancia entre sus ítems. Son elementos compuestos.

``: define una lista ordenada de artículos (numéricas).

``: define una lista de artículos sin orden (viñetas).

``: define un artículo de una lista.

CODER HOUSE

EJEMPLO

Ambas listas se deben insertar mediante la etiqueta `` (list-item).

Ejemplo de servicios de una empresa (lista de viñetas/sin orden):

```
<ul>
  <li>Empresa</li>
  <li>Producto</li>
  <li>Servicios</li>
  <li>Contacto</li>
</ul>
```

- Empresa
- Producto
- Servicios
- Contacto

CODER HOUSE

ANIDAR LISTAS

Es probable que te veas en la necesidad de crear una estructura de sublistas como la siguiente:

- Computadoras portátiles:
 - Procesador I4.
 - Procesador I5.
- Computadoras de escritorio:
 - Procesador Pentium.
 - Procesador Celeron.

CODER HOUSE

TABLAS

CODER HOUSE

TABLAS

Son un **conjunto de celdas organizadas**, dentro del cual es posible alojar distintos contenidos. HTML dispone de una gran variedad de etiquetas y atributos para crear tablas.

Sirven para representar información tabulada, en filas y columnas. En HTML4 las tablas se usaban para maquetar. Cuando CSS creció y se hizo más fuerte, nacieron los detractores de las tablas.

CODER HOUSE

ETIQUETAS BÁSICAS PARA TABLAS EN HTML

<th>Fecha</th>					
<tr>					
Ciudad	Fecha	Temperatura	Altitud	Población	Ranking Cena
El Baixo Miño (Galicia)	Agosto 2, 1995	23°	200 m	2.000	5/5
El Maestrazgo de Teruel	Junio 2, 1995	20°	1.300 m	500	3/5
El Priorato Tarragona	Mayo 1, 1995	18°	700m	800	4/5
<td>Mayo 1, 1995</td>					
</table>					

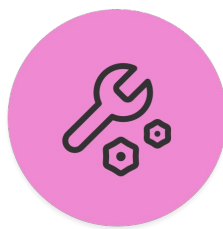
CODER HOUSE

```
<table>
<tr><!-- inicio de fila-->
  <td>Fila 1 - Columna 1</td>
  <td>Fila 1 - Columna 2</td>
  <td>Fila 1 - Columna 3</td>
</tr><!-- cierre de fila -->
<tr><!-- inicio de otra fila-->
  <td>Fila 2 - Columna 1</td>
  <td>Fila 2 - Columna 2</td>
  <td>Fila 2 - Columna 3</td>
</tr><!-- cierre de la segunda fila -->
</table>
```

La etiqueta `<table></table>` acepta 3 atributos de “diseño”:

- **Border:** bordes de la tabla.
- **Cellpadding:** especifica el espacio, en píxeles, entre la pared de la celda y su contenido.
- **Cellspacing:** indica la distancia entre las celdas y el margen exterior de la tabla

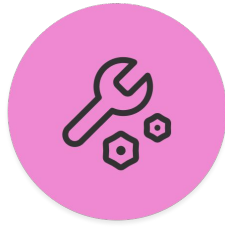
CODER HOUSE



HTML

Crea un archivo HTML.

CODER HOUSE



LISTAS

1. Crea dos listas usando las etiquetas de HTML: una con viñetas que contenga cinco nombres, y otra ordenada con 5 pasos para preparar un mate. Cuentas con 15 minutos para completar la actividad.

CODER HOUSE




BREAK

¡5/10 MINUTOS Y VOLVEMOS!

FORMULARIOS

CODER HOUSE

FORMULARIOS



The image shows a contact form titled "Contact Us" centered on a yellow background. The form is a white rectangle with a thin border. It contains three input fields: "Name" (a single-line text box), "Email" (a single-line text box), and "Message" (a multi-line text area with a vertical scrollbar). Below these fields is an orange "Submit" button.

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

CODER HOUSE

ETIQUETA **<FORM>**

Para insertar un formulario se usa la etiqueta **<form>**, que dentro lleva todos los controles que vayan al mismo destino. Un formulario requiere 3 atributos para funcionar:

- **Action:** documento que se encarga de recibir los datos y procesarlos.
- **Method:** la forma en que será enviada la información. Existen dos métodos de envío, que son GET y POST.
- **Enctype:** cómo se codificarán los contenidos.

CODER HOUSE

ACTION

En este atributo se indicará **cuál es el archivo que recibe y procesa los datos**. Debe ser de un lenguaje de los llamados “del lado del servidor” (PHP / ASP / JSP). Si no se indica un valor, el **Action** será por defecto el mismo archivo donde está el formulario.

CODER HOUSE



¡HTML NO ES UN LENGUAJE DE PROGRAMACIÓN!

CODER HOUSE

METHOD

Forma en la que se recopilan y envían los datos. Existen dos métodos comunes en el HTML:

- **GET:** la información viajará por la barra de direcciones a continuación del nombre del archivo.
- **POST:** la información viajará junto a los encabezados del HTML (será “invisible”).

Si el method no se indica, por defecto será GET.

CODER HOUSE

ENCTYPE

Cuando el valor del atributo **method** es **post**, el mismo es el [tipo MIME](#) del contenido, que es usado para enviar el formulario al servidor. Los posibles valores son:

- **application/x-www-form-urlencoded:** será el **valor por defecto** si un atributo no está especificado.
- **multipart/form-data:** usar este valor si se está usando el elemento **input** con el atributo **type** ajustado a **"file"**.
- **text/plain** (HTML5)

Normalmente se utiliza para permitir el envío de archivos a través de un formulario.

CODER HOUSE

INGRESO DE TEXTO

Existen tres controles generales para el ingreso de texto:

- Cajas de texto de una sola línea (no acepta el uso de la tecla Enter).
- Cajas para el ingreso de contraseñas (el contenido no será visible).
- Cajas para contenido multilínea. Puede ser una o muchas líneas de texto.

Atributo **"name"**

Control de formulario: `<input>`: Text, Email, Password.

Control de formulario: `<textarea></textarea>`

CODER HOUSE

BOTONES

Los botones disparan las acciones del formulario. Hay 3 tipos:

- El que envía los datos al archivo indicado como **Action**.
- El que vacía todo lo ingresado y resetea los campos.
- El que “no hace nada”, pensado para usarse con Javascript.

Todos los botones son etiquetas **<input>**, con distintos tipos de “Type”. El botón debe de estar dentro del **<form>** que afectará.

CODER HOUSE

ATRIBUTO “VALUE”

Representa la etiqueta del botón, la cual es normalmente mostrada por los navegadores dentro de éste.

- Input de tipo “*submit*”: envía el formulario.
- Input de tipo “*reset*”: resetea el formulario.
- Input de tipo “*button*”: no tiene acciones por defecto.

```
<form>  
  <input type="submit" value="Enviar formulario"/>  
  <input type="reset" value="Limpiar formulario"/>  
  <input type="button" value="Sin acciones"/>  
</form>
```

CODER HOUSE

CONTROLES DE SELECCIÓN

En estos casos, el usuario no puede ingresar libremente un texto, sino que el programador le da una lista predefinida. El dato que llega al elegir una opción se define desde el atributo “value”. Existen 3 grupos de controles de selección:

- **Botones de radio:** sólo se puede elegir una opción.
- **Casillas de chequeo:** de toda la lista de opciones, el usuario puede optar por una, todas o ninguna opción.
- **Menú desplegable:** sólo es posible seleccionar una opción.

CODER HOUSE

ATRIBUTO “VALUE”

En este caso es el valor que se enviará al enviarse el formulario.

Botones de radio:

```
<form>
  <div>hombre</div>
  <input type="radio" name="sexo" value="hombre" />
  <div>mujer</div>
  <input type="radio" name="sexo" value="mujer" />
</form>
```

Casillas de chequeo:

```
<form>
  <div>Acepta términos y condiciones</div>
  <input type="checkbox" name="acepta" value="1" />
</form>
```

CODER HOUSE

ETIQUETA <LABEL>

Define formalmente a cada elemento de un formulario. Esta etiqueta es de mucha ayuda para generar un formulario accesible.

Su principal atributo es “for”, que va a referenciar a “label” con su elemento del formulario. El valor del atributo “for” debe ser igual al valor del atributo “id” o “name” del elemento.

```
<form>
  <label for="nombre_apellido">Nombre:</label>
  <input type="text" name="nombre_apellido" />
</form>
```

CODER HOUSE

MENÚ DESPLEGABLE

Es el llamado **combo-box, selector o menú**. De toda la lista, se puede elegir una opción (aunque tiene un atributo que permite cambiarlo). Lo ideal es que sean al menos dos elementos distintos para observar el select:

```
<form>
  <select name="talles">
    <option value="L">Large</option>
    <option value="M">Medium</option>
    <option value="S">Small</option>
  </select>
</form>
```

CODER HOUSE

CONJUNTO DE CAMPO

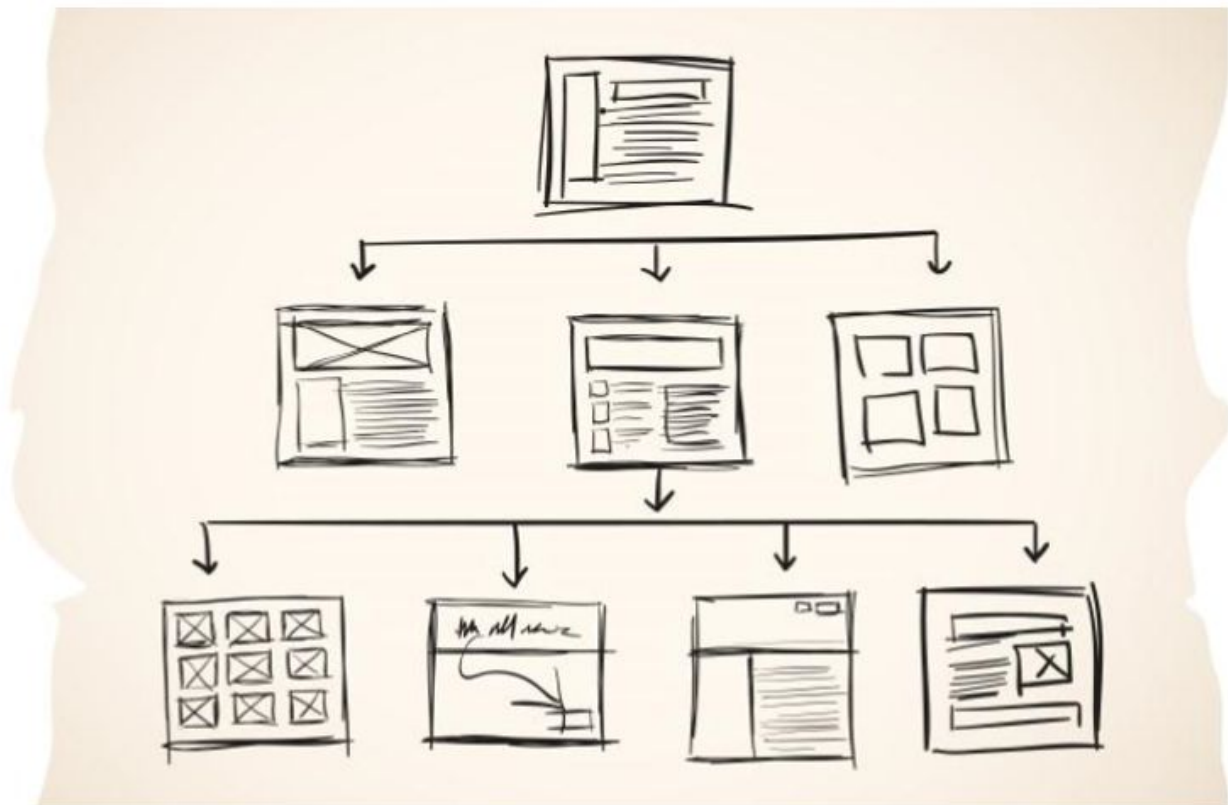
Las etiquetas `<fieldset>` y `<legend>` se utilizan en conjunto. La primera, tiene como objetivo crear grupos de elementos del formulario que posean un mismo propósito; mientras que la segunda, define formalmente el propósito del elemento fieldset. Se estructuran de la siguiente manera:

```
<form>
  <fieldset>
    <legend>Talle de remeras</legend>
    <!-- Aquí irán los elementos de formulario -->
  </fieldset>
</form>
```

CODER HOUSE

ENLACES

CODER HOUSE



ARQUITECTURA DE UN SITIO

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

Para crear uno, es necesario utilizar la etiqueta de ancla **<a>** con el atributo “*href*”, que establecerá el destino al que apunta. Por ejemplo:

```
<a href="productos.html">Productos</a>
```

ENLACES RELATIVOS, ABSOLUTOS E INTERNOS

CODER HOUSE

ENLACES RELATIVOS

Los **enlaces relativos** son aquellos que apuntan a páginas ubicadas dentro del mismo proyecto. Si la página referenciada se encuentra en el mismo directorio, alcanza con mencionar el nombre de la misma para generar el enlace.

```
<a href="contacto.html">Contacto</a>
```

En caso de que el archivo se encuentre en un directorio específico, el mismo deberá ser mencionado.

```
<a href="imagenes/mapa.jpg">ver mapa</a>
```

CODER HOUSE

ENLACES ABSOLUTOS

Los **enlaces absolutos** son aquellos cuyo destino apunta a un documento que está fuera del sitio, y debe ser especificado utilizando la URL completa:

```
<a href="http://www.coderhouse.com/frontend">Curso de Frontend</a>
```

CODER HOUSE

ENLACES INTERNOS

Los **enlaces internos** permiten referenciar secciones de tu página, para lo cual se utiliza el **id**:

```
<a href="#pie">Ir al pie de página</a>  
...  
<footer id="pie"></footer>
```

CODER HOUSE

También puedes usar como destino una sección específica una página distinta:

```
<a href="contacto.html#formulario">Formulario de contacto</a>
```

En el ejemplo anterior, el enlace apunta a la sección que tiene el id formulario dentro de la página “contacto.html”. No sólo es posible agregar enlaces a texto, también puedes hacerlo con otros elementos. Por lo general, se usan textos o imágenes. Veamos un ejemplo de enlaces

con una imagen:

```
<a href="http://www.coderhouse.com/cursos.html#frontend">  
    
</a>
```

CODER HOUSE

Desafío
generico



¡A PRACTICAR!

Crea un formulario de contacto como indica la imagen a continuación, usando los códigos vistos anteriormente. Tienes 15 minutos para realizar la actividad.

Información Personal:

Sexo:
☐ hombre ☐ mujer

Nombre:

Apellido:

Talle:

CODER HOUSE



ESTRUCTURA HTML DEL PROYECTO

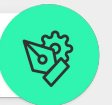
A partir del tema que hayas elegido para tu proyecto final, crea los documentos HTML para cada sección del sitio web y, además, una lista con el menú principal de este último.

CODER HOUSE

ESTRUCTURA HTML DEL PROYECTO

Formato: [Sublime Text/Visual Studio](#). Debe tener el nombre "Idea+Apellido"
Sugerencia: incluir sugerencias.

Desafío
entregable



>> Consigna: a partir del tema que hayas elegido para tu proyecto final, crea los documentos HTML para cada sección del sitio web y, además, una lista con el menú principal de este último.

>>Aspectos a incluir en el entregable:

1. Siguiendo el o los wireframes que creaste en la clase anterior, y en función de las secciones elegidas para tu sitio web, crea los documentos HTML para cada sección del mismo (los cuales luego enlazarás entre sí).
2. Crea una lista con el **menú principal** del sitio web, incluyendo los nombres de cada una de las secciones este último. Deberá estar enlazado a los archivos correspondiente, mediante enlaces relativos.
3. Agrega una **<tabla>** para mostrar los beneficios del producto que estés promocionando en tu sitio web. Incluye su nombre, información técnica (lista), características, así como lista de beneficios, comentarios o recomendaciones (lista ordenada).
4. Incorpora un formulario en la sección que corresponda para contacto. El mismo debe contar con al menos un input, un menú desplegable, un checkbox, un botón para resetear la información introducida y el botón de enviar.

>>Ejemplo: haz clic [aquí](#) para acceder.

CODER HOUSE

¿PREGUNTAS? 

¡MUCHAS GRACIAS!

Resumen de lo visto en clase hoy:

- Sintaxis de HTML.
- Diferentes etiquetas y el uso de cada una.
- Acercamiento al concepto de web semántica, y las etiquetas HTML5 de estructura.



OPINA Y VALORA ESTA CLASE

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





¿DUDAS DEL ON-BOARDING?

[MIRALO AQUI](#)

CODER HOUSE

Clase 03. DESARROLLO WEB

***Incluir Multimedia y CSS a
nuestro Proyecto***



OBJETIVOS DE LA CLASE

- Agregar multimedia con HTML.
- Comprender la sintaxis de CSS.
- Incluir CSS en el proyecto.

CODER HOUSE

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.

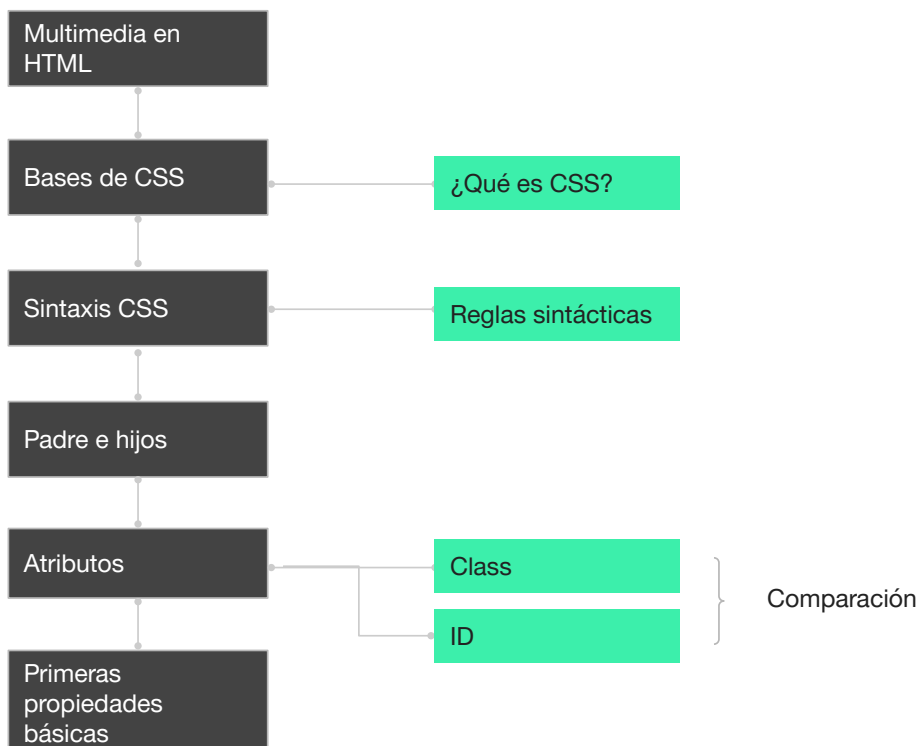
CODER HOUSE

MAPA DE CONCEPTOS

CODER HOUSE

MAPA DE CONCEPTOS CLASE 3

¡Para
recordar!



CODER HOUSE

CRONOGRAMA DEL CURSO



CODER HOUSE



Momento de exposición

CODER HOUSE

MULTIMEDIA EN HTML

CODER HOUSE

IMÁGENES

- **Enriquecen el HTML:** las imágenes son elementos que enriquecen la experiencia de los usuarios.
- **Insertar imágenes:** se insertan con la etiqueta ``, que pertenece al grupo de las etiquetas que se cierran a sí mismas (con la barra al final). Para funcionar requiere, como mínimo, indicar en dónde está el archivo a mostrar. Eso se hace con el atributo “src” (el source o fuente), que respeta todas las reglas de ruteo vistas en los links. Se comportan como elementos de línea, es significa que se verán una al lado de la otra.

CODER HOUSE

ALT

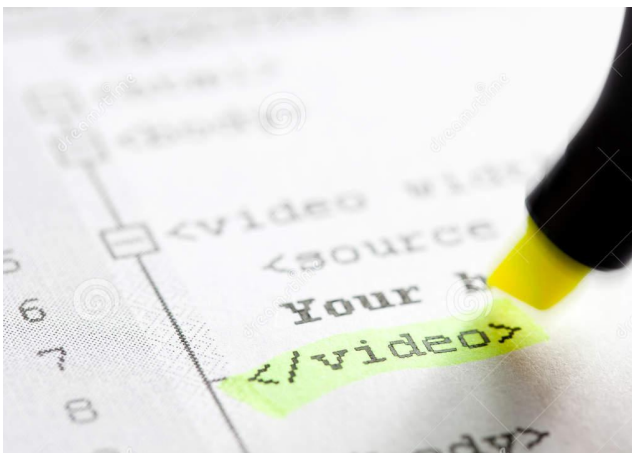
El "alt" es un texto que debe representar la foto que se está visualizando. Tiene que ser conciso y breve, pero dejar en claro de qué se trata la imagen.

```

```

CODER HOUSE

ETIQUETA VIDEO



```
<video src="mivideo.mp4"  
controls>
```

Tu navegador no implementa el
elemento `<code>video</code>`

```
</video>
```

CODER HOUSE

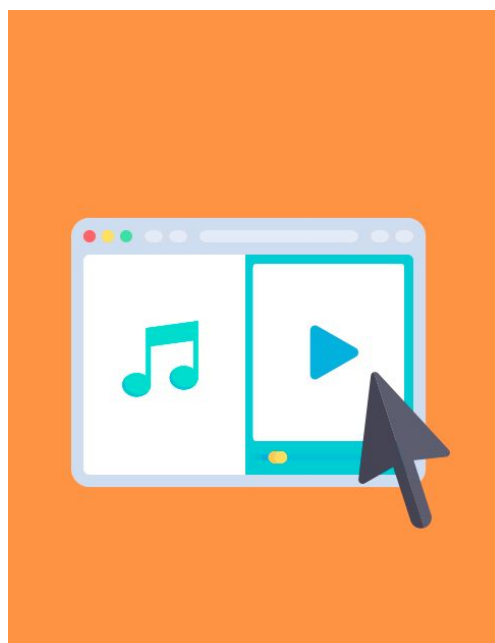
ETIQUETA AUDIO



```
<audio src="perfect.mp3" preload="auto"
controls></audio>
```

CODER HOUSE

ATRIBUTOS DE LA ETIQUETA <AUDIO>



- **Controls:** controles para manejar el audio.
- **Autoplay:** reproducción automática.
- **Loop:** repetición automática.
- **Preload:** almacenamiento temporal del audio.
 - "none" : no almacena el archivo.
 - "auto" : almacenamiento temporal del archivo.
 - "metadata" : almacena temporalmente sólo los metadatos del archivo.

CODER HOUSE

Se pueden especificar múltiples fuentes de archivos usando el elemento **<source>**, con el fin de proporcionar vídeo o audio codificados en formatos diferentes, para diversos navegadores.

```
<video controls>
  <source src="foo.ogg" type="video/ogg">
  <source src="foo.mp4" type="video/mp4">
  Tu navegador no implementa el elemento
<code>video</code>.
</video>
```

CODER HOUSE


ETIQUETA IFRAME

Es un elemento HTML que permite insertar o incrustar un documento HTML dentro de un documento HTML principal.

```
<iframe src="pagina_fuente.html" width=290
height=250>Texto para cuando el navegador no
conoce la etiqueta iframe</iframe>
```

CODER HOUSE

ETIQUETA IFRAME



CÓDER

Cómo buscar empleo por LinkedIn: "Buscar tra...

Ver más ta... Compartir

Insertar vídeo

```
<iframe width="560" height="315"
src="https://www.youtube.com/embed/
5Ujb8B7JHv4" frameborder="0"
allow="accelerometer; autoplay;
encrypted-media; gyroscope;
picture-in-picture"
allowfullscreen></iframe>
```

☐ Empezar en 0:01

COPIAR

CODER HOUSE

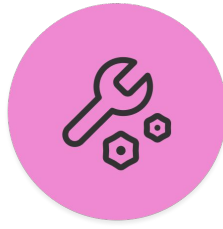
Ejemplo
en vivo



¡VAMOS A PRACTICAR LO VISTO!

Momento de consolidación de aprendizajes

CODER HOUSE



HTML

Crea un archivo HTML.

CODER HOUSE

¡A PRACTICAR!

Desafío
generico



Crea un archivo HTML que contenga: un video, un audio (puedes descargar ambos de Drive), y un iframe que muestre el video de [Coderhouse](https://www.coderhouse.com). Tienes 15 minutos para realizar la actividad.

CODER HOUSE

BASES DE CSS

CODER HOUSE

PREMISAS

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.

CODER HOUSE

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



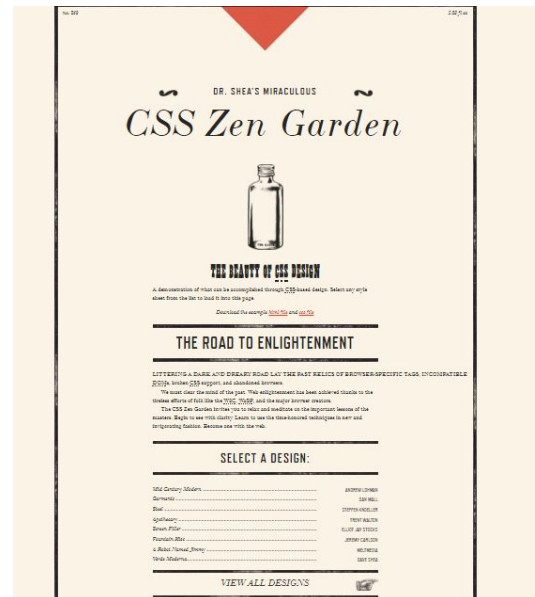
SOBRE EL PROFESOR

El profesor explica, hace y ustedes practican

SOBRE USTEDES

Aprenderan a diseñar una página web

CSS bien implementado permite cambiar **todo el diseño** de un sitio web, sin modificar el HTML. Las siguientes dos imágenes corresponden al mismo código HTML pero distinto CSS:



CODER HOUSE

SINTAXIS CSS

CODER HOUSE

SINTAXIS

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

Ejemplo

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

CODER HOUSE

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.

CODER HOUSE

PADRE E HIJOS

CODER HOUSE

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, con
elit. Temporibus cum magnam eos a c
="#"> possimus </a> iure, eius nemo
debitis!
    </p>
  </article>
</section>
```


CODER HOUSE

PADRE E HIJOS

Esto te 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 **HIJO**



Selector **PADRE**  **section article {**
background-color: #cccccc;
width: 500px;
height: 500px;
}

CODER HOUSE

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.

HTML

```
<section>
  <article>
    <h2>Título</h2>
    <p> Lorem ipsum dolor sit amet,
    consectetur adipisicing elit.
    Temporibus cum magnam eos a commodi
    sequi possimus iure, eius nemo
    voluptates fuga debitis!
    </p>
  </article>
</section>
```

CSS

```
section{
  padding: 50px 30px 20px 60px;
  margin-left: 40px;
}
section article {
  background-color: #cccccc;
  width: 500px;
  height: 500px;
}
section article p{
  line-height: 4;
}
```

CODER HOUSE

INSERTAR CSS

CODER HOUSE

INSERTAR CSS EN EL HTML

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" />
```

1

CODER HOUSE

INSERTAR CSS EN EL HTML

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  
    código CSS, */  
</style>
```

2

CODER HOUSE

INSERTAR CSS EN EL HTML

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

CODER HOUSE

CLASS

CODER HOUSE

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.

CODER HOUSE

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 */  
}
```

CODER HOUSE

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" />
```

CODER HOUSE

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" />
```

CODER HOUSE

ATRIBUTO ID

CODER HOUSE

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.

CODER HOUSE

ID 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:

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

CODER HOUSE

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>
```

CODER HOUSE

COMPARACIÓN CLASS VS. ID

CODER HOUSE


COMPARACIÓN

	¿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">

CODER HOUSE


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.

CODER HOUSE

HERENCIA Y CASCADA

CODER HOUSE

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>
```

CODER HOUSE

CASCADA

El navegador lee de arriba hacia abajo (forma de cascada)
¿qué color crees que se aplicará al párrafo (p) al ver el
siguiente código?

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

CODER HOUSE

EJEMPLO DE BUENA PRÁCTICA



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; }
```



CODER HOUSE

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

CODER HOUSE

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>
```

CODER HOUSE

ESPECIFICIDAD

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 **Inline**.

CODER HOUSE

!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.



CODER HOUSE

PRIMERAS PROPIEDADES BÁSICAS

CODER HOUSE

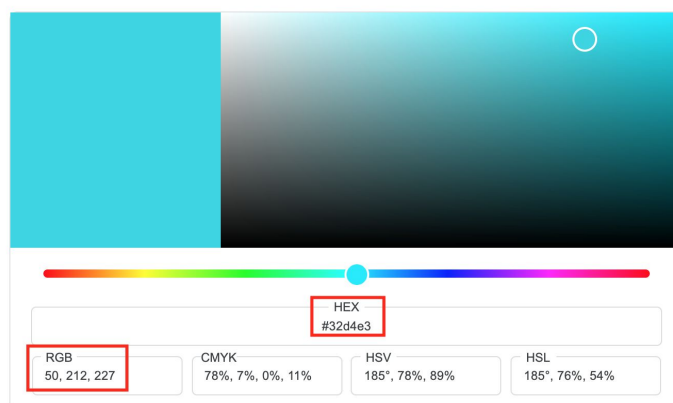
PROPIEDAD: COLOR

Mediante esta propiedad, podrás agregar color a los textos de tu sitio, pero... ¿cómo se eligen los colores?

CODER HOUSE

PROPIEDAD: COLOR

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



CODER HOUSE

PROPIEDAD: COLOR

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



CODER HOUSE

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.

CODER HOUSE



BREAK

¡5/10 MINUTOS Y VOLVEMOS!

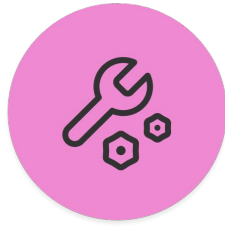
Ejemplo
en vivo



¡VAMOS A PRACTICAR LO VISTO EN BREAKOUT ROOMS!

Momento de consolidación de aprendizajes

CODER HOUSE



ATRIBUTOS

Agrega atributos a los archivos.

CODER HOUSE

Desafío
generico



¡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.

CODER HOUSE



AGREGANDO CSS A NUESTRO HTML

Comienza a utilizar CSS en tu proyecto.

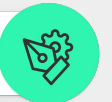
CODER HOUSE

AGREGANDO CSS A NUESTRO HTML

Formato: archivo HTML y CSS. Debe tener el nombre "Idea+Apellido".

Sugerencia: crear carpeta en formato zip o rar, con el/los archivos HTML y CSS.

Desafío
entregable



>> Consigna: crea un archivo CSS, y linkéalo al HTML entregado en el desafío de la clase anterior, esto es, "Primeros pasos con HTML". Asigna color a títulos, párrafos y listas, mediante clases y etiquetas.

>>Aspectos a incluir en el entregable:

Detalle completo acerca de lo que se espera que el estudiante entregue. Se sugiere detallar cada uno de los ítems.

>>Ejemplo:

[Carpeta comprimida con los archivos](#)

CODER HOUSE

¿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**

CODER HOUSE

¡MUCHAS GRACIAS!

Resumen de lo visto en clase hoy:

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