

Desarrollo WEB

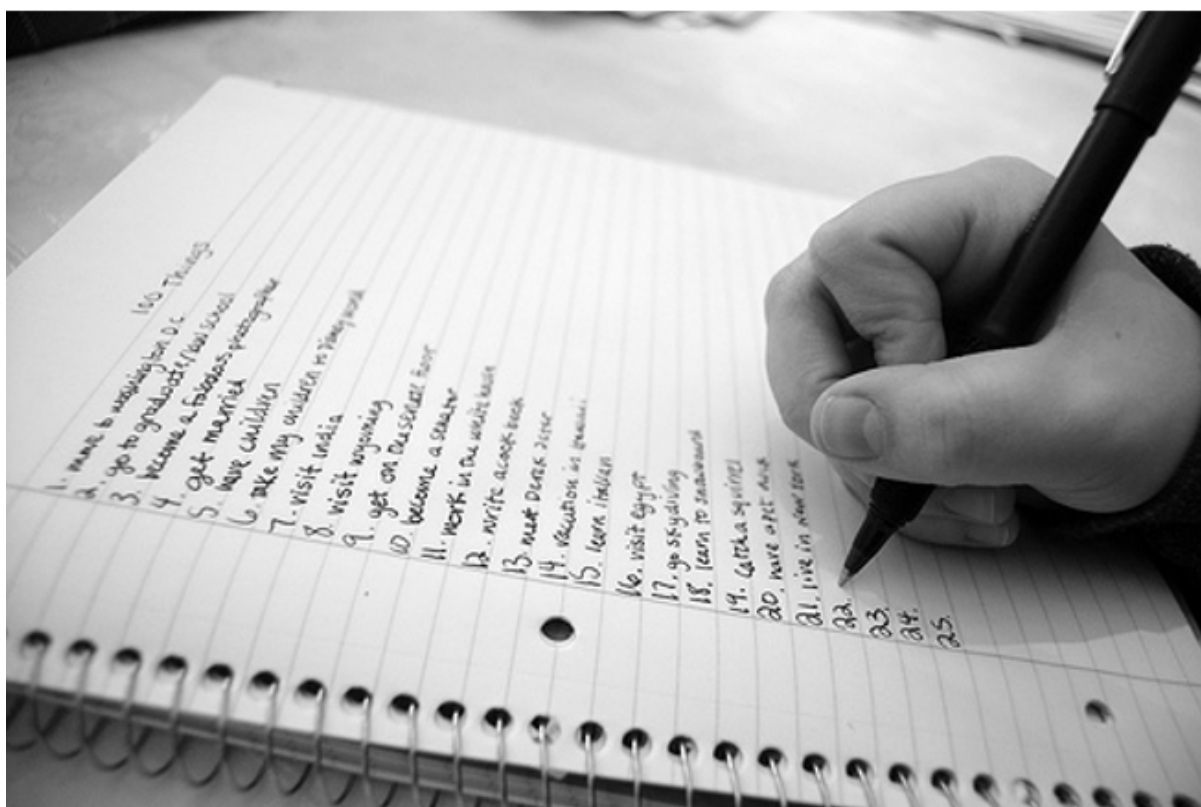
## ***CLASE 2***

### ***Material complementario***

*Primeros pasos con HTML*

***CODER HOUSE***

# LISTAS (NÚMEROS Y VIÑETAS)



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 es denominado como **listas**.

## Tipos de listas:

- Listas no ordenadas

- Listas ordenadas
- Listas de definición

## Implementaciones prácticas

- Toda enumeración de pasos o secuencias a seguir en orden (listas ordenadas).
- Cualquier enumeración de propiedades o características técnicas de una persona o producto (ítem: valor).
- Las galerías de imágenes (al tener todas las fotos la misma información –supongamos título, imagen y descripción– y por formar todas parte del mismo álbum).
- Las botoneras de todo sitio web (ya que cada botón forma parte del listado de opciones que tenemos para navegar nuestra página).

## ¿Viñetas o números?

**Las listas numéricas establecen un orden en la lectura de sus ítems** (indicando qué elemento va antes del próximo). También definen una jerarquía entre los elementos, cuál de los ítems es más importante que el siguiente.

**Las listas de viñetas no representan ningún orden o importancia entre sus ítems**, se asumen todos igual de importantes. Es indistinto en qué orden se leen, el conjunto de los ítems representan un todo.

**Son elementos compuestos.** Dado que por un lado tenemos la inserción de la lista, y por otro lado los elementos (ítems) del listado en cuestión. Entonces, para insertar una lista deberíamos usar:

- **<ol>**: Define una lista ordenada de artículos (numéricas).
- **<ul>**: Define una lista de artículos sin orden (viñetas).
- **<li>**: Define un artículo de una lista.

Cada uno de los ítems que forman parte de cualquiera de ambas listas se debe insertar mediante la etiqueta **<li></li>** (list-item). Tendremos un list-item por cada elemento que enumeramos en su respectiva lista.

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

Ejemplo de pasos para hacer un mate (Lista numérica)

```
<ol>
  <li>Se llena el "mate" en 3/4 partes con yerba</li>
  <li>Se humedece la yerba</li>
  <li>Dejar reposar</li>
  <li>Agregar agua caliente, nunca hirviendo</li>
</ol>
```

1. Se llena el "mate" en 3/4 partes con yerba
2. Se humedece la yerba
3. Dejar reposar
4. Agregar agua caliente, nunca hirviendo

## Anidar listas

Es probable que nos veamos en la necesidad de crear una estructura de sub-listas como la siguiente:

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

En este caso estamos hablando de dos listas desordenadas. Pero tranquilamente podríamos estar hablando de una lista numerada dentro de otra lista. **Es importante saber que dentro de una lista sólo se aceptan list-items, y dentro de un list-item se acepta cualquier elemento (incluso otras listas).**

## Listas de definición

Las listas de definición no forman parte del conjunto anterior, dado que **tienen un**

objetivo diferente que las listas ordenadas o desordenadas y sus elementos son diferentes.

En primera instancia, una lista de definición **no está formada por ítems, sino por significados**. Y para lograrlo, se requieren dos elementos distintos: Por un lado el término a explicar y por otro lado el significado que corresponde a dicho término.

Para insertar, entonces, una lista de definición deberemos trabajar con **tres elementos distintos**:

- **<dl>**: Define una lista de definiciones, es decir, una lista de términos y sus definiciones asociadas.
- **<dt>**: Representa un término definido por el siguiente **<dd>**
- **<dd>**: Representa la definición de los términos listados antes que él.

```
<dl>
  <dt>SGML</dt>
  <dd>Metalenguaje para la definición de otros lenguajes de
  marcado</dd>

  <dt>XML</dt>
  <dd>Lenguaje basado en SGML y que se emplea para describir
  datos</dd>

  <dt>RSS</dt>
  <dt>GML</dt>
  <dt>XHTML</dt>
  <dt>SVG</dt>
  <dt>XUL</dt>
  <dd>Lenguajes derivados de XML para determinadas aplicaciones</dd>
</dl>
```

## ***SGML***

Metalinguaje para la definición de otros lenguajes de marcado

## ***XML***

Lenguaje basado en SGML y que se emplea para describir datos

## ***RSS***

## ***GML***

## ***XHTML***

## ***SVG***

## ***XUL***

Lenguajes derivados de XML para determinadas aplicaciones

# ***TABLAS***

Una tabla es un **conjunto de celdas organizadas** dentro de las cuales podemos 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.

¿Qué elemento vas a usar para crear una grilla tabular? ¿Qué elemento sirve para maquetar un calendario? ¿Querés algo que tenga filas y columnas? Para ello existen las tablas. Además se usa para el desarrollo de newsletters.

## **Etiquetas básicas para tablas en HTML**

Las tablas son definidas por las etiquetas `<table>` y `</table>`.

Las tablas son descritas por líneas de arriba a abajo (y luego por columnas de izquierda a derecha). Cada una de estas líneas, llamada fila, es definida por otra etiqueta y su cierre: `<tr>` y `</tr>`

Dentro de cada línea, habrá diferentes celdas. Cada una de estas celdas será definida por otro par de etiquetas: `<td>` y `</td>`. Dentro de estas etiquetas colocaremos el contenido.

- **<table>**: Representa la tabla como contenedor de las filas y columnas.
- **<caption>**: Representa el título de una tabla.
- **<colgroup>**: Representa un conjunto de una o más columnas de una tabla.
- **<col>**: Representa una columna de una tabla.
- **<tbody>**: Representa el bloque de filas que describen los datos concretos de una tabla.
- **<thead>**: Representa el bloque de filas que describen las etiquetas de columna de una tabla.
- **<tfoot>**: Representa los bloques de filas que describen los resúmenes de columna de una tabla.
- **<tr>**: Representa una fila de celdas en una tabla.
- **<td>**: Representa una celda de datos en una tabla.
- **<th>**: Representa una celda encabezado en una tabla.

A continuación veremos un ejemplo del código de una tabla básica:

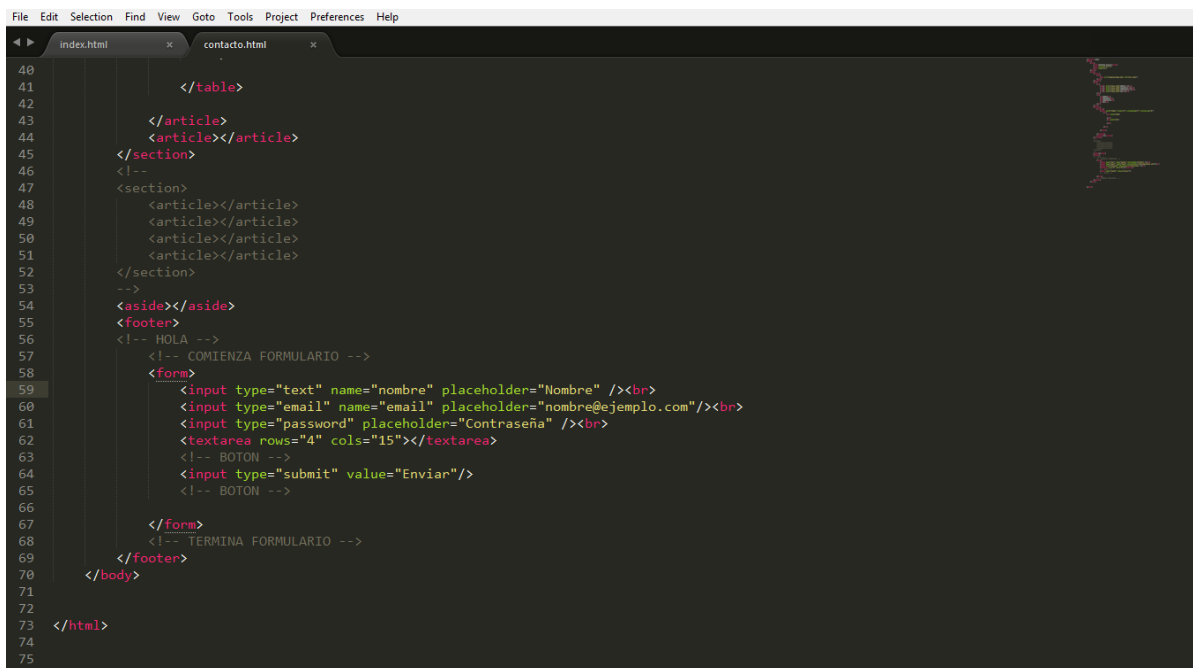
```
<table>
<tr><!-- inicio de fila-->
  <td>Fila 1 - Columna 1</td>
  <td>Fila 1 - Columna 2</td>
  <td>Fila 1 - Columna 3</td>
</tr><!-- cierre de fila -->
<tr><!-- inicio de otra fila-->
  <td>Fila 2 - Columna 1</td>
```



```
<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:** Número que representa el tamaño del borde de todas las filas y columnas (por defecto es 0).
- **Cellpadding:** Número que representa el espacio interno en píxeles de una celda (por defecto es 3).
- **Cellspacing:** Número que representa el espacio hacia afuera entre celda y celda (por defecto es 3).

A screenshot of a code editor with a dark theme. The editor shows an HTML file named 'contacto.html'. The code includes a table, several article and section tags, a footer, and a form with input fields for name, email, password, and a submit button. Line numbers 40 through 75 are visible on the left side of the editor. The form section includes comments in Spanish like 'COMIENZA FORMULARIO' and 'TERMINA FORMULARIO'.

El valor indicado se aplica a los 4 lados por igual. Si querés algo más específico tendrás que utilizar **CSS**.

# FORMULARIOS

A contact form titled "Contact Us" is centered on a yellow background. The form is a white rectangle containing three teal-colored input fields stacked vertically. The first field is labeled "Name", the second "Email", and the third "Message". Below these fields is an orange "Submit" button.

HTML no es un lenguaje de programación sino de estructura (para hacer maquetas).

Si se necesita recopilar información ingresada por un usuario, HTML ofrece controles de formulario.

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

Los formularios se usan bastante seguido en Web:

- Cuando ingresas a un sistema de usuario/clave.
- Cuando publicas un nuevo contenido en Facebook, Twitter, Google plus, Taringa, Blogger, etc.
- Cuando enviás un mensaje desde un sitio web.
- Cuando confirmás una compra o entrada al cine.

## 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: GET y POST.
- Enctype: Cómo se codificarán los contenidos.

## Action

En este atributo indicaremos **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, por defecto el Action es el mismo archivo donde está el formulario.

*Recordemos que:* HTMLno es un lenguaje de programación.

## Method

**Forma en la que se recopilan y envían los datos.** Existendos 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”).

Por defecto (si no se indica el Method) es GET.

## 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"

Todos los controles de formulario deben tener un nombre, que se indica con el atributo "name". Con ese nombre después se accede a su valor en el archivo indicado en el action. Un control sin name no se envía.

### Control de formulario: <input>

Este elemento tiene múltiples propósitos según el valor que tome el atributo "type".

Los posibles **valores del atributo type para el ingreso de texto** son:

**Text:** Es el valor por defecto, el más común. El input toma la estructura de una línea de texto.

```
<input type="text" name="nombre"/>
```

**Email:** El elemento input acepta como valor una dirección de mail. Al momento de ser validada requiere que el texto ingresado tenga formato de correo electrónico. *Con el atributo **placeholder**, podemos colocar un texto de ayuda para el usuario, dándole una pista de cómo completar el campo*

```
<input type="email" name="email" placeholder="nombre@ejemplo.com"/>
```

**Password:** Este valor es utilizado para campos de contraseñas, el texto ingresado no es visible, se reemplaza por asteriscos o bullets.

```
<input type="password"/>
```

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

Es un campo con múltiples líneas de texto. Tiene un comportamiento similar al de un input de texto con la propiedad de aceptar saltos de líneas.

**El campo textarea acepta los siguientes atributos:**

- **cols:** Indica la cantidad de caracteres que podrá tener el campo a lo ancho.
- **rows:** Indica la cantidad de líneas de texto.

```
<textarea rows="4" cols="15"></textarea>
```

## Botones

Los botones disparan las acciones del formulario.

Hay 3 tipos de botones:

- 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 y está 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.

### 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>
```

## Controles de selección

El usuario no puede ingresar libremente un texto, sino que el programador le da una lista predefinida. En todos los casos, el dato que nos 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 elegir una, todas o ninguna opción.
- Menú desplegable: Sólo se puede elegir una opción.

### Atributo “*value*”

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

### Botones de radio

Radio button define un elemento con múltiples opciones que puede tomar sólo un valor como respuesta. Se usa para datos como estado civil, sexo, etc. Es la etiqueta `<input>`

con el tipo “radio”. Es la única etiqueta que debe tener el mismo “name” para todas las opciones del mismo conjunto. No tiene ningún atributo para definir su texto (lo que ve el usuario), se escribe “a mano”.

```
<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

Es un controlador con valor booleano que sólo puede estar deshabilitado o habilitado. También es un `<input>` con el tipo “checkbox”. Necesita un value distinto para cada opción.

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

## Etiqueta <label>

La etiqueta `<label>` define formalmente a cada elemento de un formulario, esta etiqueta es de mucha ayuda para generar un formulario accesible. El **principal atributo de la etiqueta label es “for”, quien va a referenciar a “label” con su elemento del formulario.** El valor del atributo for debe ser igual al valor del atributo id ó name del elemento.

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

## 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). Se necesitan dos elementos distintos:

Para insertar el menú se usa la etiqueta **<select>**, que abre y cierra (pero estará vacío)

Dentro va una etiqueta **<option>** que abre y cierra por cada opción a mostrar. Dentro va el texto a mostrar.

El **<select>** lleva el atributo *"name"* (los **<option>** no)

Los **<option>** llevan el *"value"* (el **<select>** no)

Si no hay *"value"*, se usa el texto interior

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



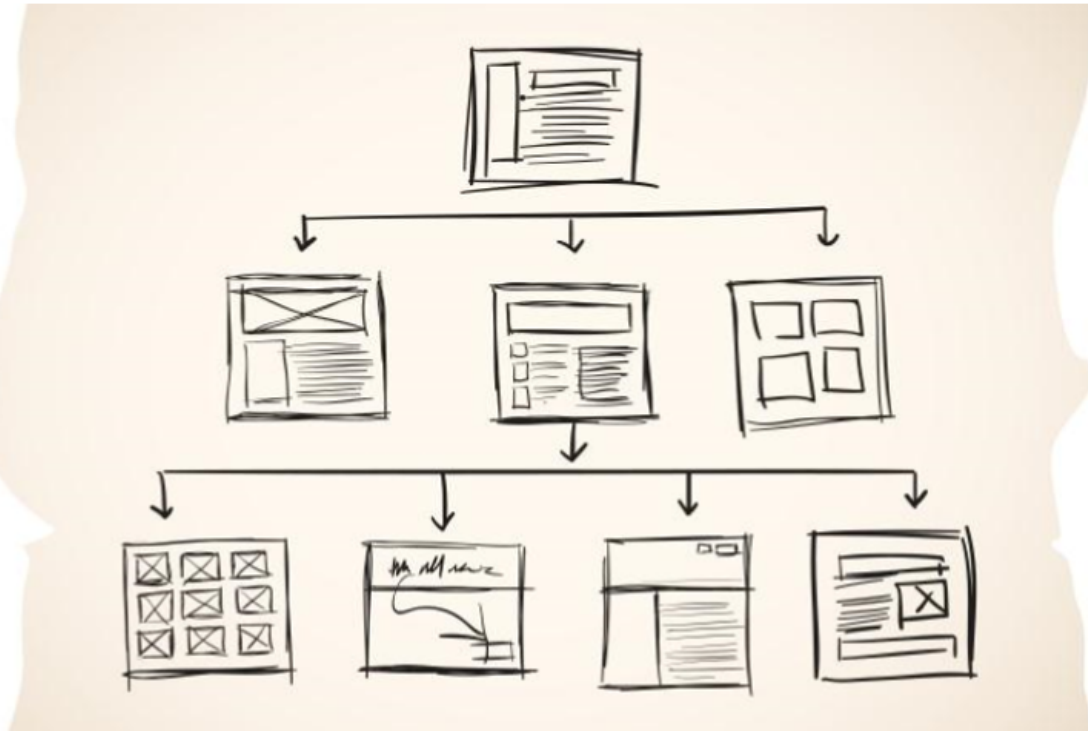
# Conjunto de campo

Las etiquetas `<fieldset>` y `<legend>`

La etiqueta `fieldset` tiene como objetivo crear grupos de elementos del formulario que poseen un mismo propósito. La etiqueta `legend`, define formalmente el propósito del elemento `fieldset`. Se estructura de la siguiente manera:

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

# ENLACES



## Arquitectura de un sitio

Los enlaces (también conocidos como links o anchors) se utilizan para relacionar partes del documento con otros documentos o con partes del mismo documento. Por defecto, los enlaces se visualizan azules y subrayados.

Para crear un enlace es necesario utilizar la etiqueta de ancla `<a>` con el atributo `href`, que establecerá el destino al que apunta.

Sintaxis de un enlace:

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

El texto "productos", que figura entre las etiquetas `<a>`, es el que se visualizará en el navegador y que servirá como enlace a la url especificada en el atributo `"href"`.

## Enlaces relativos, absolutos e internos.

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>
```

En el ejemplo anterior, el destino al cual queremos acceder (en este caso una imagen) se encuentra dentro de la carpeta "imágenes" con lo cual hay que especificar esa ruta en el atributo `"href"`.

Los enlaces absolutos son aquellos cuyo destino apunta a un documento que fuera del

sitio y debe ser especificado utilizando la URL completa:

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

Los enlaces internos permiten referenciar secciones de nuestra página, para hacer esto se utiliza el id:

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

También podemos 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 podemos agregar enlaces a texto, también podemos hacerlo con otros elementos. Por lo general se usan texto o imágenes. Ejemplo de enlaces con una imagen:

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

En este caso la imagen será clickeable y enlazará al destino especificado en el atributo “href”.

## Enlaces de Interés:

<http://www.dirinfo.unsl.edu.ar/dweb/listaetiquetasHTML5.pdf>

<https://www.w3schools.com/html/default.asp>

<https://developer.mozilla.org/es/docs/Web/HTML>