

Desenvolvimento Web

Estrutura básica em HTML



Índice

Introdução	4
Comentário	6
Estrutura de uma página web	6
Organização del head	6
Organização del body	8
Formatos de texto em HTML	9
Cabeçalho	10
Parágrafos	11
A etiqueta <pre>	13
Hyperlinks	13
Hiperlinks ou links internos. Etiqueta <a>	13
Hiperlinks externos. Atribuição href.	14
Imagens como links	16
Listas	16
Listas não-ordenadas	16
Listas ordenadas	17
Listas de definições ou descrições	17
Tabelas	19
Unificação de células	20
Células de cabeçalho	20
O atributo scope	22
Imagens	23
Vídeos	26
Áudio	27
Formulários	27
Formas de envio de dados a partir de um formulario html. Métodos get e post	28
Entrada de dados em formulários	29
Text	29
Label	32
Select	33
Textarea	34
Checkbox	35
Radiobutton	36
Botão submit	37
Botão de reset	37
Botão de imagem	37
<fieldset> e <legend>	37
Arquivos anexos	38
Campos ocultos	38

Iframes	39
Vías em HTML	39
Vías de arquivo absolutas	39
Vías de arquivo relativas	40
Apêndice Etiquetas HTML	41

Introdução

HTML (Linguagem de Marcação de Hipertexto, HyperText Markup Language em inglês) é uma "linguagem" composta de **etiquetas** com as quais dizemos ao programa para nos mostrar textos, imagens, vídeos, etc.

A origem do HTML data de 1980, quando o físico Tim Berners-Lee, um trabalhador do CERN (Organização Européia de Pesquisa Nuclear), propôs um novo sistema de "hipertexto" para compartilhar documentos.

O HTML não é uma linguagem de programação em si, embora às vezes seja referido coloquialmente como uma linguagem de programação, é simplesmente uma série de tags que o navegador interpretará de uma forma ou de outra para exibir conteúdos diferentes na tela.

HTML é a linguagem utilizada para o desenvolvimento de páginas na Internet. É composto por uma série de tags que o navegador interpreta e molda na tela. O HTML tem tags para imagens, hiperlinks que nos permitem ir a outras páginas, quebras de linha, listas, tabelas, etc.

Classicamente, diz-se que as linguagens de programação incluem três capacidades básicas para gerar fluxos de processo: seqüencial (seqüências de instruções), condicional (a capacidade de tomar decisões ou executar um ou outro processo dependendo do valor de um ou mais parâmetros) e repetição (a capacidade de repetir um processo um certo número de vezes). Linguagens clássicas como C, C++, Java, C#, Visual Basic, Fortran, etc. têm estas capacidades. O HTML não os tem, não porque é melhor ou pior, mas porque é algo mais.

Em resumo, poderíamos dizer que o HTML não é uma linguagem de programação, é uma linguagem de layout web ou linguagem de tags destinada a criar estruturas de documentos HTML.

Poderíamos dizer que o HTML é usado para criar páginas web, dar-lhes estrutura e conteúdo.

O padrão atual do HTML5 consiste em três "idiomas" em um, HTML, CSS e Javascript:

- Com HTML, fazemos o "esqueleto" de nossa página web. A disposição inicial dos títulos, parágrafos, imagens... etc.
- Com o CSS3, damos a aparência à nossa página.
- Com Javascript, fornecemos interatividade e animação.

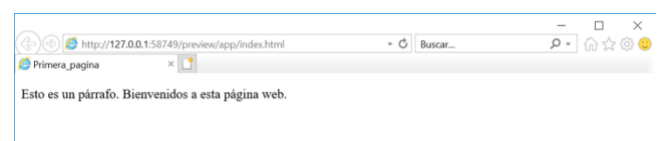
Um exemplo simples de código **HTML** poderia ser:

```
<html>
  <body>
    <p>Esto es un párrafo. Bienvenidos a
    esta página web.</p>
  </body>
</html>
```

Este exemplo consiste em 3 tags **HTML**. Como podemos ver, cada uma das etiquetas deve terminar com uma etiqueta de fechamento. Neste caso, o **<html>** deve ser fechado com **</html>**, la etiqueta **<body>** com **</body>** e la etiqueta **<p>** com **</p>**.

Há muito mais tags que veremos mais tarde, mas devemos ter claro que, em geral, para cada tag que abrimos, devemos incluir a tag de fechamento correspondente (há exceções, as veremos mais tarde). Desta forma, obteremos um código **HTML** bem formado que os navegadores podem interpretar sem ambigüidade.

Este simples exemplo mostraria o seguinte na tela.



O que aconteceria se uma etiqueta que abrimos não tivesse uma etiqueta de fechamento correspondente? Digamos que seria um código **HTML** mal construído, e os navegadores podem interpretar isto de diferentes maneiras. Talvez eles nos mostrem a página como esperávamos, sem nenhum erro aparente. Talvez eles nos mostrem uma página de erro ou o navegador fique em branco. Nosso objetivo deve ser sempre construir páginas **HTML** bem estruturadas e sem ambigüidades, ou seja, usar a linguagem corretamente para que os navegadores possam saber exatamente o que pretendemos exibir.

As etiquetas que não têm um par de fechamento são fechadas com uma sintaxe semelhante a esta:

```
<meta/>
```

O **HTML** está atualmente na versão 5. Estaremos programando nesta versão porque, em princípio, todos os navegadores estão otimizados para esta versão.

Que navegador devemos usar para testar nossas páginas web?...muito simples, **todas elas**. Devemos nos certificar de que nosso website possa ser visualizado corretamente em qualquer dispositivo. Um web programador deve testar suas páginas em todos os dispositivos e navegadores possíveis, pois pode acontecer que os navegadores interpretem a página de diferentes maneiras e, em um navegador, nossa página parece perfeita e, em outro, ela parece fora de ordem. Se utilizarmos corretamente as regras **HTML**, esta situação não deverá ocorrer.

Comentário

A sintaxe para o uso de comentários em **HTML5** é:

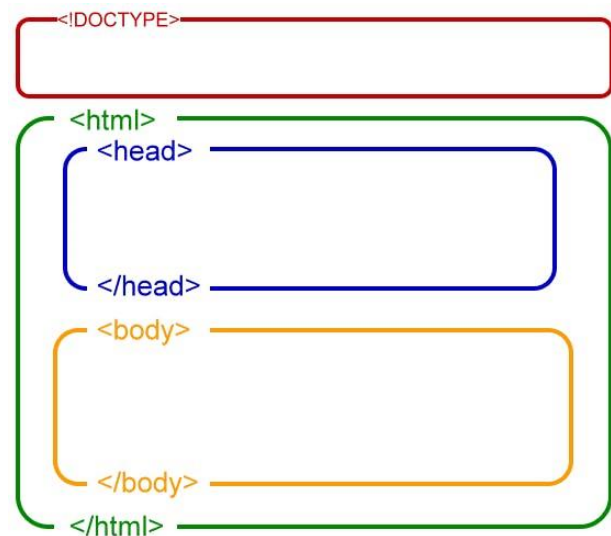
```
<!--Aquí va el comentario-->
```

Os comentários HTML são visíveis para qualquer pessoa que visualize o código fonte da página, mas não são renderizados quando o navegador processa o documento HTML.

É uma boa prática comentar o máximo possível do seu código. Será mais fácil para outro programador ou mesmo para nós mesmos se voltarmos ao código depois de algum tempo.

Estrutura de uma página web

Uma página web **HTML** consiste em princípio de duas partes distintas, a **head** e o **body**, nas quais as tags HTML são distribuídas.:



```
<!DOCTYPE html>
<html>
  <head>
    <title>Título de la página
web</title>
    ...
  </head>
  <body>
    Cuerpo de la página web
  </body>
</html>
```

Por convenção, a página central de qualquer projeto WEB é normalmente chamada **index.html**, mas não é obrigatória.

Organização del head

O **head** (ou cabeçalho da página) contém os metadados (dados que o usuário não vê, mas que o navegador considera úteis), o **title** da página e possíveis **links** para páginas CSS ou Javascript:

A tabela a seguir mostra um resumo dos elementos que podem ir dentro da etiqueta da head.

Etiqueta	Função	Obrigatório?
<title>	Dá um título ao documento HTML	Sí
<base>	Define via de acesso	Não
<link>	Define arquivos vinculados	Não
<meta>	Define metadados tais como descrição e palavras-chave	Não
<script>	Delimita scripts incluídos	Não
<style>	Delimita definição de estilos	Não

Não é necessário que nosso **head** contenha todas essas etiquetas. A única etiqueta obrigatória é **title**.

As meta tags não possuem um par de fechamento.

A etiqueta **Description** é essencial para os motores de busca, dá uma pista sobre o assunto da página e os ajuda a categorizá-la.

A etiqueta **Keywords** não é mais tão importante para os motores de busca devido aos abusos que foram feitos dela. O abuso de palavras-chave nesta etiqueta é penalizado pelos motores de busca. Ela contém palavras-chave que os motores de busca usarão para categorizar nossa página e saber qual é o assunto.

<title></title>: Define o título do documento e muda o título que aparece na guia do navegador e é também muito importante para os mecanismos de busca. Ela só pode conter texto e qualquer outra etiqueta não será interpretada. O **<title>** tag é obrigatório.

<title>Primera página</title>

A ordem das etiquetas é indiferente, exceto pela etiqueta de **link**, que, se existir, deve ser colocada idealmente logo após a etiqueta de **title**.

Etiqueta <base>. Usado para indicar a **URL** base se as **URLs** relativas forem especificadas dentro da página web. Por exemplo:

```
<base
href="http://www.páginaprincipal.com/imag
es/" target="_blank">
```

Isso faria com que, se escrevêssemos "**logo.png**" como o caminho para uma imagem, esse caminho seria na verdade:

<http://www.páginaprincipal.com/images/logo.png>

Etiquetas <link>. Eles são usados para indicar que o documento **html** está relacionado a outro arquivo ou recurso externo. Ele liga nossa página web com outras páginas externas, tais como **CSS**, **Javascript**, etc. Não tem uma etiqueta de fechamento.

Por exemplo:

```
<link rel="stylesheet" type="text/css"
href="estilos.css"/>
```

Neste caso, indicamos que o documento **HTML** está vinculado a um arquivo estilo **css**, **javascript**...etc., veremos isso mais tarde.

Etiqueta <style>. É usado para incluir estilos **CSS** que permitem adicionar cores, bordas, imagens de fundo, etc. aos elementos da página web..

Etiquetas <meta>. Elas servem para incluir informações que não são exibidas como parte da página da web, mas servem para informar aos navegadores sobre características da página da web, tais como sua breve descrição e palavras-chave.

Exemplo:

```
<meta name="description"
content="Programación HTML, CSS y
Javascript">
<meta name="keywords" content="curso,
web, programación, aprender">
```

Neste caso, as etiquetas dizem aos mecanismos de busca o conteúdo de nossas páginas (**description**) e algumas palavras-chave (**keywords**) para sua localização. Isto é muito útil para que nossa página apareça nos motores de busca (google, bing, yahoo, etc.).

Etiquetas <script>. Eles são usados para incluir código em linguagens de script, tais como **javascript**. Veremos isso também mais adiante.

Em muitas páginas, o código **JavaScript** está incluído no cabeçalho, que é uma linguagem de programação que os navegadores são capazes de reconhecer e interpretar. O código **JavaScript** pode ser reconhecido porque está incluído entre as tags `<script></script>`:

```
<script>
  Aquí iría el código
</script>
```

O exemplo a seguir seria uma implementação perfeita da **head** para uma página com conteúdo **HTML** e **CSS**:

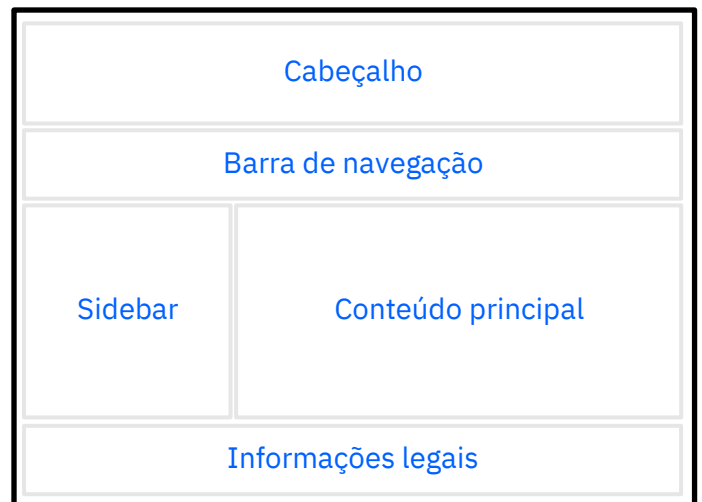
```
<head>
  <meta charset="utf-8">
  <meta name="Description"
content="Página de prueba"/>
  <meta name="Keywords" content=" HTML,
CSS, Javascript"/>
  <title> Curso de HTML, CSS,
Javascript </title>
  <link rel="stylesheet"
href="Hojadeestilos.css"/>
</head>
```

Organização del body

A etiqueta de fechamento da **head** é seguida pela etiqueta de abertura do **body**.

No **body** (ou corpo da página) introduzimos a parte visível da página.

No passado, a página era dividida em células com a etiqueta `<table></table>`, mais tarde foi utilizada a etiqueta `<div></div>`, que era chamada de modelo box-container, hoje em dia é utilizado o seguinte esquema:



Esta distribuição é indicativa, qualquer elemento pode ser omitido ou movido de acordo com nosso projeto. Somente a **HEAD** e o **BODY** são obrigatórios, mas a inclusão de todos estes elementos melhora o posicionamento **SEO**.

header e **footer** podem ser aplicados não apenas ao documento, mas também a qualquer parte dele, ou seja, podemos colocar, por exemplo, um **header** e um **footer** dentro de etiquetas **section**.

Formatos de texto em HTML

Quando trabalhamos com texto em HTML, temos uma série de tags que escrevemos em torno da palavra ou texto e que transformam o texto no formato que queremos dar a ele. Algumas dessas etiquetas são depreciadas e não devem ser usadas. Devido ao seu amplo uso no passado, é útil saber quais eram os usos tradicionais dessas etiquetas, mas hoje em dia elas foram substituídas pelos estilos CSS. No entanto, eles ainda estão disponíveis.

Etiqueta	Uso	Observaciones
<code>...</code>	Texto em negrito	Puede ser sustituido por CSS.
<code>...</code>	Texto em negrito	Puede ser sustituido por CSS.
<code><i>...</i></code>	Italicise text	Puede ser sustituido por CSS.
<code>...</code>	Italicise text	Puede ser sustituido por CSS.
<code><u>...</u></code>	Texto sublinhado	Deprecated. Sustituir por CSS.
<code><small>...</small></code>	Faça o texto menor	Puede ser sustituido por CSS.
<code><big>...</big></code>	Aumentar o texto	Puede ser sustituido por CSS.
<code><sub>...</sub></code>	Definir texto subscrito	Puede ser sustituido por CSS.
<code><sup>...</sup></code>	Definir texto superescrito	Puede ser sustituido por CSS.
<code><strike>...</strike></code>	Definir o texto como riscado	Deprecated. Sustituir por CSS.
<code><s>...</s></code>	Definir o texto como riscado	Deprecated. Sustituir por CSS.
<code>...</code>	Definir o texto como riscado	Puede ser sustituido por CSS.

Tags e seu uso para adaptar a formatação na programação HTML

Nosso idioma ficaria assim:

```
<!DOCTYPE html>
<html lang="es">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible"
content="IE=edge">
  <meta name="viewport"
content="width=device-width, initial-
scale=1.0">
  <title>Document</title>
</head>

<body>
  <b>Poner texto en negrita</b><br>
  <strong>Poner texto en
negrita</strong><br>
  <i>Poner texto en cursiva</i><br>
  <em>Poner texto en cursiva</em><br>
  <u>Poner texto subrayado</u><br>
  <small>Poner texto más
pequeno</small><br>
  <big>Poner texto más grande</big><br>
  <sub>Poner texto subíndice</sub><br>
  <sup>Poner texto
superíndice</sup><br>
  <strike>Poner texto como
tachado</strike><br>
  <s>Poner texto como tachado</s><br>
  <del>Poner texto como
tachado</del><br>
</body>

</html>
```

Considerando que a saída no navegador ficaria assim:

Poner texto en negrita

Poner texto en negrita

Poner texto en cursiva

Poner texto en cursiva

Poner texto subrayado

Poner texto más pequeño

Poner texto más grande

Poner texto subíndice

Poner texto superíndice

~~Poner texto como tachado~~

~~Poner texto como tachado~~

~~Poner texto como tachado~~

Cabeçalho

Os títulos são o que entendemos em qualquer comunicado à imprensa como títulos e legendas.

As etiquetas `<h></h>` são usadas para títulos, variando de `<h1></h1>` a `<h6></h6>`, sendo `<h1></h1>` as mais relevantes e `<h6></h6>` as menos importantes. Não apenas mudam o tamanho do texto, os motores de busca também dão importância a essas etiquetas.

Os próprios cabeçalhos geram uma quebra de linha.

Recomenda-se que haja apenas uma `<h1></h1>` tag em cada página da web, pois para fins de **SEO** o mecanismo de busca sempre procurará o título da WEB.

Quando temos vários títulos H, os agrupamos junto com a etiqueta `<hgroup></hgroup>`

Por exemplo, esta seria uma lista de todas as possíveis etiquetas `H` disponíveis para nós:

```
<body>
  <hgroup>
    <h1>Esto es un H1</h1>
    <h2>Esto es un H2</h2>
    <h3>Esto es un H3</h3>
    <h4>Esto es un H4</h4>
    <h5>Esto es un H5</h5>
    <h6>Esto es un H6</h6>
  </hgroup>
</body>
```

O resultado em nosso navegador seria:

Esto es un H1

Esto es un H2

Esto es un H3

Esto es un H4

Esto es un H5

Esto es un H6

Parágrafos

Para dizer ao navegador que queremos colocar o texto em um parágrafo, devemos escrevê-lo entre as etiquetas `<p>` e seu fechamento `</p>`, com o texto separado por uma margem em branco acima e abaixo.

Quando temos vários parágrafos que estão relacionados, eles são geralmente agrupados em uma única etiqueta. `<article></article>`.

Um atributo amplamente utilizado no passado (agora depreciado e substituído pelo `text-align` do CSS) foi o `align`, que foi utilizado para definir o alinhamento do texto dentro do parágrafo.

```
text-align:
  center
  end
  inherit
  justify
```

Para separar um texto de outro ou um parágrafo de outro, podemos usar uma linha horizontal de um tamanho ou espessura determinados por nós. Escrevemos este separador com a etiqueta `<hr>` e sua espessura, cor...etc. veremos como mudá-lo mais tarde usando o CSS:

```
<p>Esto es un párrafo</p>
<hr>
<p>Esto es otro párrafo bajo una
línea de separación hecha con HR</p>
```

Resultado no navegador:

Esto es un párrafo

Esto es otro párrafo bajo una línea de separación hecha con HR

```
<!doctype html>

<html lang="ES-es"> <!--Inicio de la página-->

<head>                                <!--Inicio del HEAD-->
  <meta charset="utf-8"/>

                                <!--Descripcion de la pagina-->
  <meta name="Description" content="Descripcion de la pagina"/>

                                <!--Palabras clave-->
  <meta name="Keywords" content="Palabras clave, separadas por comas" />

                                <!--Titulo de la pagina-->
  <title>Titulo de mi pagina</title>

                                <!--Vinculo a una hoja de estilos CSS externa-->
  <link rel="stylesheet" href="MiHojaDeEstilos.css"/>

</head>                                <!--Fin del HEAD-->

<body>                                <!--Comienzo del BODY-->

                                <!--Parrafo-->
  <p>Esto es un párrafo de texto. Para crear automáticamente un texto de prueba,
escribimos LoremXX, donde XX será el número de palabras que queremos crear. Y a
continuación le damos a la tecla TAB.
</p>

  <br>                                <!--Salto de linea-->

  <pre>
Etiqueta de texto preformateado. El navegador interpreta el texto escrito tal y como
está, respetando los saltos de línea, espacios, etc. indicados.
Por ejemplo, ahora vamos a poner dos saltos de línea

...y luego seguiremos escribiendo.</pre>

</body>                                <!--Fin del BODY-->
</html>                                <!--Fin de la pagina WEB-->
```

Exemplo de trabalho com parágrafos.

A etiqueta <pre>

A <pre> tag é chamada de "texto pré-formatado" e permite que o navegador interprete o texto escrito como ele é, respeitando as quebras de linha, espaços, etc. indicados.

Na ausência da tag <pre>, o navegador não leva em conta que o código fonte contém quebras de linha, espaços, etc. no texto. Por outro lado, usando a <pre> tag e fechando-a com sua correspondente tag, o navegador respeitará as quebras de linha, espaços, tabuladores, etc., indicados no código fonte...

```
<body>
  <p>Esto es un párrafo hecho
    sin usar texto preformateado.
    A pesar de estar escrito en
diferentes líneas
    el navegador lo mostrará todo
seguido
  </p>
  <br>
  <pre>Esto es un párrafo hecho
    usando texto preformateado.
    El navegador lo mostrará
    tal y como está escrito
    en diferentes líneas
  </pre>
</body>
```

Hyperlinks

Links ou hiperlinks, também chamados de hipertexto, são textos ou objetos sobre os quais podemos clicar para nos levar para outra parte do documento, para outra página da web no mesmo site ou para outra página na Internet, entre outras funções.

Hiperlinks ou links internos. Etiqueta <a>

Links internos ou bookmarks são links dentro da mesma página. Ou seja, clicar em um deles nos levará a uma posição diferente dentro da mesma página que estamos vendo.

Para criar este tipo de ligação, duas operações devem ser realizadas:

- Estabelecer marcadores (âncoras) ao longo da página (estes são os lugares para os quais queremos saltar com os links).
- Coloque links que saltem para os marcadores.

O código dos marcadores é criado com o atributos **name** (não recomendado) ou **id** (recomendado):

```
<a name="nombre_del_marcador">Texto
asociado al marcador</a> (No recomendado:
no es aceptado por las nuevas versiones
de HTML, aunque se usó bastante en el
pasado).
```

```
<a id="nombre_del_marcador">Texto
asociado al marcador</a>
```

O **name** ou **id** de uma tag deve ser único, ou seja, não pode haver duas tags com o mesmo **name** ou **id** dentro de um documento **HTML**.

Por exemplo, imaginemos que no início do código de nossa página web temos a seguinte linha:

```
<a id="marcadorDeportes">Resumen
sección deportes</a>
```

Podemos colocar um link interno na parte inferior da página para que o usuário possa clicar nele e o navegador irá levá-los de volta para o topo da página. Ou seja, um link para saltar para um marcador de página:

```
Pulsa para volver al <a href="#marcadorDeportes">Inicio</a>
```

Hiperlinks externos. Atribuição href.

Um hyperlink externo é um link para outro site na Internet (site externo). É um link para qualquer outro lugar fora do site atual. Quando colocamos um link externo, digitamos o endereço completo da página, incluindo `http://www....`. Estas rotas incluem `http://...` las denominamos rutas absolutas.

No hiperlink, distinguimos as seguintes partes:

- Abertura e fechamento de hyperlink tags `<a>` e ``
- O atributo `href`, com um valor indicado pelo sinal de igual e a **URL** para a qual o hyperlink é direcionado dentro das aspas invertidas.
- Um texto que é visto pelo usuário como um **link**.

Por exemplo:

```
<a href="http://www.miPágina.es" title="Titulo del enlace" target="_blank">Ir a miPágina.es</a>
```

Alvo Hyperlink. Atributo target

Quando criamos um link, por padrão o navegador abrirá a página de destino na mesma janela, mas podemos pedir ao navegador que abra "à parte", ou seja, em outra janela.

Isto é útil, por exemplo, se quisermos abrir uma página externa ao nosso site, mas sem que o visitante perca a nossa. Para isso, usamos o atributo `target` com uma das seguintes opções.

Valores de `target` mais comuns:

`_blank`: Abre o documento vinculado em uma nova janela do navegador.

`_self`: Esta é a opção padrão. Abre o documento vinculado no mesmo quadro ou janela em que o **link** foi clicado.

Exemplo:

```
<a href="http://www.miPágina.es" title="Titulo del enlace" target="_blank">Ir a miPágina.es</a>
```

Como último ponto, devemos levar em conta que é muito aconselhável colocar um atributo extra toda vez que colocamos um hyperlink em nossas páginas. Este atributo é `'title'` e com ele daremos um título ao nosso hyperlink.

Desta forma, obteremos na maioria dos navegadores um efeito 'tooltip' que consiste em que quando colocamos o cursor sobre o hyperlink aparece uma informação adicional, que é a que colocamos no atributo `'title'`.

Exemplo de um website com links internos e externos:

```
<!DOCTYPE html>
<html lang="es">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible"
content="IE=edge">
  <meta name="viewport"
content="width=device-width, initial-
scale=1.0">
  <title>Document</title>
</head>

<body>
  <h1 id="titulo">Título de mi
página</h1>

  <p id="parrafo1">Lorem ipsum dolor
sit amet consectetur adipisicing elit.
Eveniet eos aliquid fugiat maiores iure
voluptate? Iusto hic placeat
beatae libero consequuntur, sapiente in
quae, velit fuga facere iure consequatur
quas, suscipit repellendus
assumenda vero. Architecto repellendus
delectus itaque neque? Consequatur eos
ea,

  cumque, cupiditate soluta
doloremque eveniet veritatis et sed
inventore fugit architecto tempore nihil?
Porro

  excepturi fuga expedita
consequatur, rerum adipisci
necessitatibus facilis animi tenetur
corporis! Sed, rem vero

  cum obcaecati recusandae amet
deleniti cupiditate assumenda error
consectetur modi ex dolore explicabo
temporibus suscipit animi
laudantium at nobis fugit ipsa aspernatur
ab nam! Provident recusandae ipsam maxime
rem ducimus.</p>

  <p id="parrafo2">Lorem ipsum, dolor
sit amet consectetur adipisicing elit.
Illo ab expedita facere natus corporis
```

```
consequuntur nobis eveniet ad
laborum aspernatur, quam similique libero
accusamus tempore distinctio deleniti
dicta vero? Ab possimus ipsam sit
neque aspernatur, rem sed dolores
blanditiis ullam aperiam nemo odio
praesentium ex, maiores
consequatur expedita? Cum enim unde
quaerat ex hic voluptatibus aliquam
eaque! Ut odio,

  animi ad soluta libero dolorum
recusandae! Eos nesciunt rem itaque
consectetur. Esse dolores explicabo enim
quo

  voluptates at numquam excepturi,
quia fugiat. Eligendi dolorem
necessitatibus illum explicabo veniam
perferendis

  eius officia tempora doloribus!
Excepturi minima dolore deserunt possimus
error dolores! Fugiat, porro eum

  minima ratione numquam tenetur
minus, voluptatem quidem molestiae ipsa
iste eveniet nobis asperiores alias ipsam
iure id quibusdam quaerat? Error
tempora officiis quisquam animi tenetur
magni libero dolor aut. Esse maxime

  odio dolores quas expedita
incidunt id consequatur! Deserunt
consequuntur, molestiae officiis
distinctio tempore

  libero dignissimos qui vitae odit
quisquam dolore odio delectus! Vero,
omnis debitis saepe atque cumque

  consequuntur quidem nisi,
quisquam maxime obcaecati, numquam iusto
necessitatibus nobis magnam dicta?
Maxime architecto quibusdam, modi
accusamus alias amet reprehenderit
adipisci enim.

  </p>
  <a id="enlace"
href="http://www.santillana.es"
title="www.santillana.es"
target="_blank">Ir a santillana.es</a>
  <br>
  <a href="#titulo">Inicio</a>
  <br>
  <br>
```

```
<a href="#parrafo3">tercer
parrafo</a>

</body>

</html>
```

Imagens como links

Para colocar uma imagem como um link, basta criar um link e adicionar uma imagem dentro dele, em vez de texto. Vamos ver um exemplo:

```
<a href="http://www.miPágina.es">
  
</a>
```

A sintaxe para a colocação de uma imagem será discutida posteriormente. No momento, o aprendiz só precisa entender que as imagens também podem funcionar como links.

Listas

Listas em **HTML** nos permitem criar conjuntos de elementos na forma de uma lista dentro de uma página, todos geralmente precedidos por um hífen ou número.

Os tipos de listas **HTML** são os seguintes:

- Listas desordenadas
- Listas ordenadas
- Listas de definição

Listas não-ordenadas

Listas sem ordem são aquelas encontradas entre as etiquetas `` (ul indica lista sem ordem), as etiquetas de abertura e fechamento respectivamente. Se quisermos acrescentar um novo ponto, teremos que fazê-lo dentro das tags ``.

Se não indicarmos nada na etiqueta ``, a bala ou marca indicando que é um item da lista será gerada automaticamente. Por padrão, as listas não ordenadas são compostas de pontos. Mas se quisermos definir o símbolo da bala ou marca a ser usada por nós mesmos, devemos indicá-lo especificamente. No passado, o atributo `"type"` era usado, mas hoje não é recomendado (depreciado). Com este atributo, a lista poderia ser definida por pontos negros (`li type="disc"`), pontilhado com fundo branco (`li type="circle"`) ou por quadrados (`li type="square"`).

A sintaxe recomendada para indicar a aparência é baseada no uso do CSS, como mostrado abaixo. Analisaremos mais de perto como isso deve ser feito corretamente ao visualizar os estilos de CSS para listas:

```
<ul style="list-style-type:disc">
<ul style="list-style-type:circle">
<ul style="list-style-type:square">
```

Exemplo de uma lista não ordenada:

```
<ul>
  <li>Primer elemento de la lista
sin ordenar</li>
  <li>Segundo elemento de la lista
sin ordenar</li>
  <li>Tercer elemento de la lista
sin ordenar</li>
</ul>
```


Listas ordenadas

Se quisermos definir uma lista ordenada, teremos que fazê-lo entre as etiquetas `` `` (ol indica lista ordenada). Além disso, cada elemento da lista será escrito com a mesma etiqueta que para listas não ordenadas: ``. Mas como são listas ordenadas, os símbolos padrão serão números e estes serão gerados automaticamente em ordem, à medida que escrevemos novos itens da lista.

```
<ol>
  <li>Primer elemento de la lista
ordenada</li>
  <li>Segundo elemento de la lista
ordenada</li>
  <li>Tercer elemento de la lista
ordenada</li>
</ol>
```

Por meio do CSS podemos mudar os tipos de números que nossa lista de pedidos nos mostrará. Veremos que por meio do CSS podemos colocar números romanos...

Listas de definições ou descrições

Este tipo de listas não é usado com frequência, por isso vamos citá-las caso encontremos este tipo de código em uma página da **web**, para poder interpretar seu significado. Listas de definição são usadas quando queremos fazer uma enumeração do tipo "dicionário", onde temos vários termos e sua definição ou descrição.

Por exemplo, estes seriam vários termos e suas definições:

Termo	Definição ou descrição
FTP	Protocolo para transmissão de arquivos entre computadores
HTML	Linguagem de tag utilizada para gerar páginas web
PHP	Linguagem interpretada por servidor para gerar páginas web dinâmicas.

Estes termos e suas definições ou descrições podem ser colocados de várias maneiras dentro de uma página da **web** (como texto simples, como uma lista ordenada, lista não ordenada...) e uma dessas maneiras é colocá-la como uma lista de definições.

Para criar uma lista de definições, devemos usar as tags `<dl>`, `<dt>` e `<dd>`. Vamos explicá-las em partes:

La etiqueta `<dl>` indica que uma lista de definições ou uma lista de descrições estará contida dentro dela.

La etiqueta `<dt>` indica que contém um termo que iremos definir da seguinte forma.

La etiqueta `<dd>` nos diz que dentro dela está uma definição ou descrição associada a um termo. Um termo poderia ter várias descrições. Por exemplo, o termo Autor poderia ter como descrições: Mateo Renzi, Olivo Pascua, Jorge Guillén.

As listas de definição serão automaticamente separadas se digitarmos várias delas.

Se desejaros, podemos combinar alguns tipos de listas com outros. Por exemplo, ordenaram listas dentro de cada elemento de uma lista não ordenada.

```
<dl>
  <dt>Café</dt>
  <dd>Bebida caliente</dd>
  <dt>Refresco</dt>
  <dd>Bebida fría</dd>
</dl>
```

Todas as listas que temos visto podem ser misturadas de acordo com nossas necessidades. Ou seja, podemos colocar listas dentro de outras listas, criar listas ordenadas dentro de listas não ordenadas... etc. Qualquer combinação que possamos pensar é possível.

Por exemplo:

```
<!doctype html>

<html lang="ES-es"> <!--Inicio de la
página-->

  <head>                                <!--Inicio del
HEAD-->
    <meta charset="utf-8"/>

                                <!--Descripcion
de la pagina-->
    <meta
name="Description" content="Descripcion
de la pagina"/>

                                <!--Palabras
clave-->
    <meta name="Keywords"
content="Palabras clave, separadas por
comas" />

                                <!--Titulo de la
pagina-->
    <title>Titulo de mi
pagina</title>
```

```
                                <!--Vinculo a una
hoja de estilos CSS externa-->
    <link rel="stylesheet"
href="MiHojaDeEstilos.css"/>

  </head>                                <!--Fin del HEAD-
-->

  <body>                                <!--Comienzo del
BODY-->

                                <!--Lista
ordenada-->
    <ol>
      <li>Primer elemento de la
lista ordenada</li>
      <li>Segundo elemento de la
lista ordenada</li>
      <li>Tercer elemento de la
lista ordenada</li>
    </ol>

    <br>
    <br>

                                <!--Lista no
ordenada-->
    <ul>
      <li>Primer elemento de la
lista no ordenada</li>
      <li>Segundo elemento de la
lista no ordenada</li>
      <li>Tercer elemento de la
lista no ordenada</li>
    </ul>

    <br>
    <br>

                                <!--Lista no
ordenada dentro de otra ordenada-->
    <ol>
      <li>Primer elemento de la
lista ordenada</li>
      <li>
        <ul>
```

```

        <li>Segundo de la
ordenada y primero de la lista no
ordenada</li>

        <li>Segundo de la
ordenada y segundo de la lista no
ordenada</li>

        <li>Segundo de la
ordenada y tercero de la lista no
ordenada</li>
    </ul>
</li>
<li>Tercer elemento de la
lista ordenada</li>
</ol>

<br />
<br />
<p>Lista por definición</p>
<br />
<dl>
    <dt>Coffee</dt>
    <dd>Black hot drink</dd>
    <dt>Milk</dt>
    <dd>White cold drink</dd>
</dl>

</body>                <!--Fin del BODY-
->
</html>                <!--Fin de la pagina
WEB-->

```

Tabelas

As tabelas podem ser consideradas como um grupo de linhas onde cada linha contém um grupo de células (colunas). Uma tabela pode ser inserida em um documento **HTML** usando três elementos básicos: o elemento **TABLE** (principal contendo a estrutura), o elemento **TR** (recipiente de linha) e o elemento **TD** (célula).

Quando o conteúdo de uma célula deve estar vazio, você deve usar um espaço em branco (que em **HTML** é escrito como ` `) como seu conteúdo. Isto fará com que sua página seja exibida corretamente, pois

alguns navegadores têm problemas para renderizar células vazias. Exemplo: `<td> </td>`

Exemplo de uma tabela simples:

```

<table border="1px" >
    <caption>Tabla simple de
3x3</caption>
    <tr>
        <td>Celda A</td>
        <td>Celda B</td>
        <td>Celda B</td>
    </tr>
    <tr>
        <td>Celda C</td>
        <td>Celda D</td>
        <td>Celda E</td>
    </tr>
    <tr>
        <td>Celda F</td>
        <td>Celda G</td>
        <td>Celda H</td>
    </tr>
</table>

```

Por padrão, as tabelas HTML não têm bordas. Se quisermos que nossa tabela tenha um **border**, devemos especificá-la na declaração da tabela com o atributo de **border** seguido da espessura da borda. Em nosso exemplo, criamos uma borda para nossa página com uma espessura de 1 pixel:

```
<table border="1px" >
```

Unificação de células

Em algumas ocasiões, pode ser necessário fundir duas ou mais células em uma única célula que tomará o lugar das células afetadas.

Estas unificações podem ser realizadas com os atributos "[rowspan](#)" (para unificação vertical) e "[colspan](#)" (para unificação horizontal).

```
<table border="1">
  <caption>Tabla con celdas unidas
mediante colspan</caption>
  <tr>
    <th>NOMBRE</th>
    <th>Día 1</th>
    <th>Día 2</th>
    <th>Día 3</th>
    <th>Día 4</th>
  </tr>
  <tr>
    <th>Mike (lastimado)</th>
    <td colspan="3">0 km</td>
    <td>4 km</td>
  </tr>
  <tr>
    <th>Susan</th>
    <td>23 km</td>
    <td>18 km</td>
    <td>19 km</td>
    <td>15 km</td>
  </tr>
</table>
```

O exemplo a seguir ilustra a fusão vertical de células em uma tabela:

```
<!--Tabla con
celdas unidas por rowspan-->
<table border="1">
  <caption>Tabla con celdas unidas
mediante rowspan</caption>
  <tr>
    <th>&nbsp;</th> <!--Celda vacía-->
  </tr>
  <tr>
    <th>Básico</th>
    <th>Completo</th>
  </tr>
  <tr>
    <th>Instalación</th>
    <td rowspan="2">Gratis!</td>
    <td>$49.99</td>
  </tr>
  <tr>
    <th>Primer año</th>
    <td>$19.99</td>
  </tr>
  <tr>
    <th>Segundo año</th>
    <td>$9.99</td>
    <td>$29.99</td>
  </tr>
</table>
```

Quando usados juntos na mesma tabela, os atributos de [colspan](#) e [rowspan](#) devem ser cuidadosamente declarados para não produzir células sobrepostas.

Células de cabeçalho

Há dois tipos de células que podem ser definidas em uma tabela **HTML**. Uma é a célula simples (elemento **TD**), já definida acima, e a outra é um tipo especial de célula (**TH** ou elemento de cabeçalho de tabela) que contém informações de cabeçalho para um conjunto específico de células.

Os navegadores normalmente representam o conteúdo de células especiais como texto centralizado e negrito, atributos que também podem

ser visualmente alcançados usando células normais (elemento **TD**). Então, para que servem estes títulos? Quando usamos **th** a célula é definida como um cabeçalho, ela não se parece apenas com um cabeçalho. Para usar uma símile, não é a mesma coisa vestir-se como um policial sem ser um, como é ser um policial. Uma célula que se parece com um cabeçalho sem ser definida como tal não é o mesmo que uma célula que é realmente definida como um cabeçalho. Os navegadores para cegos identificam este tipo de títulos e lhes dão um tratamento especial. Além disso, alguns mecanismos de busca (bing, google, yahoo) dão um tratamento diferente a este tipo de células, e alguns navegadores criam efeitos especiais para este tipo de células.

Usando o elemento [caption](#), podemos definir o título de uma tabela. Este título deve descrever de forma breve e precisa o conteúdo e a natureza da tabela e é geralmente representado em algum lugar próximo à tabela (sua posição pode ser definida usando CSS). O elemento [caption](#) só é permitido se ele for logo após a abertura da tabela.

```
<h1>Su pedido</h1>
<table border="1px">
  <tr>
    <th scope="col">Nombre
producto</th>
    <th scope="col">Precio
unitario</th>
    <th scope="col">Unidades</th>
    <th scope="col">Subtotal</th>
  </tr>
  <tr>
    <td>Reproductor MP3 (80
GB)</td>
    <td>192.02</td>
    <td>1</td>
    <td>192.02</td>
  </tr>
  <tr>
    <td>Fundas de colores</td>
    <td>2.50</td>
    <td>5</td>
    <td>12.50 </td>
```

```
</tr>
<tr>
  <td>Reproductor de radio</td>
  <td>12.99</td>
  <td>1</td>
  <td>12.99</td>
</tr>
<tr>
  <th scope="row">TOTAL</th>
  <td>-</td>
  <td>7</td>
  <td><strong>207.51</strong></td>
</tr>
</table>
```

No exemplo a seguir, construiremos uma tabela para exibir informações sobre o tempo nos próximos dias. Aqui, as células de cabeçalho, representadas pelo elemento **th**, são colocadas na primeira linha da tabela, acima das células comuns.

```
<table border="1px">
  <tr>
    <th>Hoy</th>
    <th>Mañana</th>
    <th>Jueves</th>
  </tr>
  <tr>
    <td>Soleado</td>
    <td>Mayormente soleado</td>
    <td>Parcialmente nublado</td>
  </tr>
```

É fácil ver aqui como cada célula do cabeçalho da tabela fornece informações para o restante das células da coluna a que pertence.

Alguns agentes, tais como navegadores de voz, fazem a mesma análise quando devem informar ao usuário qual célula de cabeçalho está associada a uma determinada célula. Mas em alguns casos, é preciso evitar ambigüidades e é por esta razão que o HTML fornece alguns atributos como [scope](#).

O atributo scope

O atributo `scope` fornece um mecanismo para indicar explicitamente quais células uma célula de cabeçalho afeta. Este atributo só pode ser declarado em uma célula de cabeçalho e pode tomar os valores `"col"`, `"row"`, `"colgroup"` e `"rowgroup"`. Os valores `"col"` e `"row"` indicam que a célula de cabeçalho fornece informações para as demais células da coluna ou linha (respectivamente) em que ela está presente. Os outros dois valores farão sentido mais tarde, neste tutorial.

No exemplo a seguir, a tabela apresentada acima foi melhorada com mais células de cabeçalho, a fim de aumentar a legibilidade. Aqui, a célula no canto superior esquerdo da tabela forneceria informações ambíguas se o atributo `scope` não estivesse presente. Em outras palavras, isso afetaria as células de sua coluna, bem como as células de sua fila.

A presença do atributo `scope` deixou claro que as células afetadas por este cabeçalho são aquelas na mesma linha.

```
<table border="1px">
  <tr>
    <th scope="row">Día</th>
    <th>Hoy</th>
    <th>Mañana</th>
    <th>Jueves</th>
  </tr>
  <tr>
    <th>Condición</th>
    <td>Soleado</td>
    <td>Mayormente soleado</td>
    <td>Parcialmente nublado</td>
  </tr>
  <tr>
    <th>Temperatura</th>
    <td>19°C</td>
    <td>17°C</td>
    <td>12°C</td>
  </tr>
  <tr>
    <th>Vientos</th>
    <td>E 13 km/h</td>
    <td>E 11 km/h</td>
    <td>S 16 km/h</td>
  </tr>
</table>
```

Imagens

As imagens dentro de uma página web são incluídas usando a tag ``, que não tem uma etiqueta de fechamento correspondente.

Ao utilizar a etiqueta ``, dentro da imagem você tem que especificar o caminho onde a imagem está localizada, seja um caminho para um diretório em seu disco rígido ou para um endereço de internet. Isto é feito com o `src`.

Outro atributo, que não é obrigatório, mas altamente recomendado, é o atributo `alt`, o que nos permite exibir uma mensagem caso a imagem não possa ser encontrada.

O atributo `title` nos permite dar um título à nossa imagem, altamente recomendado para o posicionamento **SEO**.

```

```

Neste exemplo, exibimos uma imagem em nosso site especificando seu endereço na Internet com o atributo `src`, demos-lhe um texto alternativo com o atributo `alt`, que será exibido no caso da imagem não ser encontrada, e demos-lhe o título "título da foto".

Os formatos de imagem mais utilizados são **JPG**, **GIF** e **PNG**. O ideal é usar o formato .jpg sempre que possível, pois ele proporciona uma qualidade de imagem muito boa sem ser muito pesado.

A tabela a seguir resume os atributos que podem ser encontrados quando se trabalha com imagens:

Atributo	Observações	Uso
src	Este atributo é obrigatório e indica o nome do arquivo de imagem (entre aspas) ou a URL da qual a imagem deve ser recuperada.	Obrigatório. Se não for incluída, nenhuma imagem será exibida.
align	Permite controlar o alinhamento de uma imagem em relação a uma linha de texto adjacente ou a outras imagens nessa linha. Os valores possíveis são os familiares left, right, middle, top, bottom.	Atributo depreciado. Substituir por CSS.
alt	Entre aspas podemos escrever um texto que será exibido se a imagem não carregar, enquanto está carregando ou, quando a imagem já estiver exibida, passamos o mouse sobre ela.	Atributo obrigatório, recomenda-se incluí-lo, mas se não, a imagem será exibida.
width	Este atributo é opcional, mas podemos configurá-lo para especificar o navegador para exibir a imagem em um tamanho específico. Significa "largura da imagem" que vamos renderizar. Se não for digitada, a imagem é carregada com o tamanho original.	Opcional. Indicar o valor em pixels. Também pode ser indicado com CSS.
height	Assim como o atributo width, é opcional. Este atributo indica a altura da imagem.	Opcional. Indicar o valor em pixels. Também pode ser indicado com CSS.
border	Com borda, especificamos a largura da borda ao redor da imagem. Se for especificado 0, isto é equivalente a "nenhuma fronteira".	Atributo depreciado. Substituir por CSS.

Tabela resumida dos atributos que podem ser encontrados quando se trabalha com imagens.

NOTA: Quando usamos tanto **width** e **height**, nossa imagem pode ser deformada. Para evitar isso, usaremos apenas um dos dois. Desta forma, as proporções de nossa imagem permanecerão as mesmas.

Note que ao especificar o endereço da imagem, se esta imagem estiver em nosso próprio servidor podemos usar um endereço relativo, ou seja, estas duas expressões seriam igualmente válidas:

```



```

Também podemos usar outro servidor como fonte da imagem, o que significa que o navegador irá procurar a imagem no caminho que indicamos. Mas, neste caso, devemos especificar o caminho completo, um caminho relativo não será válido. Exemplo:

```

```

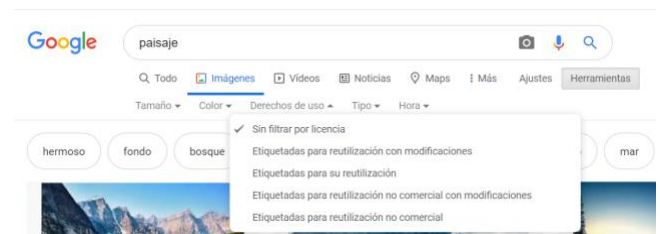
O HTML também nos dá a opção de colocar uma legenda de imagem para descrever o conteúdo de nossas imagens.

Para colocar uma legenda de imagem, usaremos a etiqueta: `<figcaption></figcaption>`

Para melhorar o posicionamento **SEO**, é aconselhável usar a etiqueta `` dentro de um `<figure></figure>`. Esta etiqueta não é usada apenas para inserir imagens, mas também qualquer elemento multimídia, vídeos, sons, animações, flash...

```
<figure>
  
  <figcaption>Descripción de la
imagen</figcaption>
  <!--Debe ir dentro de figure-->
</figure>
```

Nota: Você tem que ter muito cuidado com a questão dos direitos autorais sobre as imagens. O Google tem uma opção de busca de imagens livres de royalties:



O atributo `title` responsável por fornecer um elemento de texto na forma de uma descrição pop-up (tooltip) quando o ponteiro do mouse é posicionado sobre a imagem.

Sua missão é fornecer informações adicionais ao usuário sobre esse elemento.

```

```



Vídeos

Trabalhar com vídeos é muito semelhante a trabalhar com imagens.

Que formatos os navegadores suportam atualmente?

Navegador	MP4	WebM	Ogg
Internet Explorer	SÍ	NÃO	NÃO
Chrome	SÍ	SÍ	SÍ
Firefox	SÍ	SÍ	SÍ
Safari	SÍ	NÃO	NÃO
Opera	SÍ (a partir de Opera 25)	SÍ	SÍ

A etiqueta que nos permite utilizar vídeos é a tag `<video></video>`.

Ela tem os seguintes atributos:

- **Src:** Permite especificar onde o vídeo está localizado.
- **Controls:** Adicionar controles visuais em vídeo.
- **Autoplay:** O vídeo é executado automaticamente quando a página é carregada.
- **Loop:** O vídeo joga em loop.
- **Poster:** Permite definir uma imagem .jpg que aparecerá antes que o vídeo seja reproduzido.
- **Preload:** Permite que você especifique algumas características do vídeo antes de ser carregado. Por exemplo, quantos quadros o vídeo tem, sua duração ...

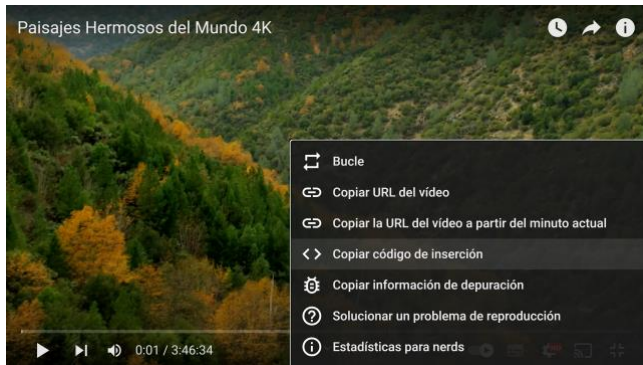
```
<video id="video" width="720px"
controls loop autoplay>
  <source
src="file:///C:/Users/video.mp4">
  <source
src="file:///C:/Users/video.ogg">
</video>
```

Para melhorar os aspectos de acessibilidade, é permitido acrescentar legendas em formato [vtt](#). Isto é feito usando a etiqueta [track](#), o que permitirá uma nova opção nas opções do jogador.

```
<track label="Español" kind="subtitle
s" srclang="es" src="subtitulos.es.vtt"
default>
```

Se quisermos embutir em nossa página HTML do YouTube, devemos fazer isso com a tag `<iframe>`.

Para fazer isto, na página de vídeo do youtube, clique com o botão direito em um menu pop-up e clique na opção para copiar o "código embutido":



Agora copiamos para nossa prancheta o código necessário para incorporar o vídeo em uma página HTML. Nós o incluímos em nossa página web e é isso:

```
<iframe width="640" height="360"
  src="https://www.youtube.com/embed/fOW8Y09GVek"
  title="Paisajes Hermosos del
Mundo 4K"
  frameborder="0"
allow="accelerometer; autoplay;
  clipboard-write; encrypted-media;
gyroscope;
  picture-in-picture"
allowfullscreen>
</iframe>
```

Às vezes podemos descobrir que o vídeo do youtube parece preto. Isto porque o vídeo é protegido por direitos autorais e não é permitido ser usado em outros sites.

Áudio

O uso de áudio em páginas web não é recomendado atualmente. É um pouco desconfortável para o usuário, especialmente quando você abre várias páginas e cada uma toca uma canção, tudo ao mesmo tempo. Entretanto, é interessante saber como utilizar esta ferramenta.

Os formatos suportados por **HTML** são .mp3 e .ogg.

A etiqueta que nos permite utilizar o áudio é `<audio></audio>`

Ela tem os seguintes atributos:

- **Src:** Necessário para especificar o caminho do arquivo de áudio.
- **Controls:** Exibe o painel de controle de áudio, com botões como play, stop, volume+, volume-...
- **Autoplay:** Especifica que o áudio toca automaticamente quando a página é carregada.
- **Loop:** Especifica que o áudio é reproduzido em loop.

Formulários

Os formulários **HTML** têm o propósito de coletar informações fornecidas pelos visitantes do site, que são então enviadas de volta ao servidor para serem processadas.

Para seu correto funcionamento, é importante que o formulário fornecido em **HTML** seja acompanhado por um código do lado do servidor, que chamaremos de "agente de processamento", que se encarregará de receber e processar as informações como o autor julgar conveniente. Este processamento pode consistir, por exemplo, em armazenar as informações ou enviá-las por e-mail.

Um formulário, identificado em HTML pelas tags `<form></form>`, é basicamente um recipiente para controles. Cada controle, em um formulário, tem o objetivo de coletar informações inseridas pelos usuários, em formulários que podem variar de linhas de texto, a upload de arquivos, a opções, datas, senhas e muito mais. Uma vez que os usuários tenham preenchido o formulário com dados, eles podem enviá-lo de volta ao servidor para que o agente de processamento gerencie as informações coletadas.

Os usuários interagem com os formulários através dos chamados controles. Um controle é definido, de forma simplificada, como um objeto que aparece na tela e que pode ser modificado pelo usuário. Por exemplo, um botão, uma caixa de texto, um menu suspenso... etc.

Os formulários geralmente têm uma etiqueta **action** que se refere à página para a qual as informações no formulário irão:

```
<form action = "pagina.php"></form>
```

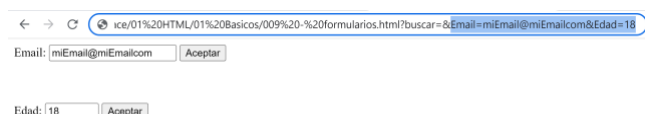
Se não vai ser enviado para nenhuma outra página, é deixado vazio para enviar as informações para a mesma página em que estamos.

```
<form action = ""></form>
```

Isto geralmente é seguido pelo atributo **method**, que pode ser **post** ou **get**.

A diferença entre os métodos **get** e **post** reside na forma como os dados são enviados para a página quando o botão "Enviar" é pressionado. Enquanto o método **get** envia os dados usando o URL, o método **post** envia-os de uma forma que não podemos vê-los (no fundo ou "escondidos" do usuário).

No exemplo a seguir, podemos ver o resultado de um formulário feito com o método **get**:



The screenshot shows a web browser address bar with the URL: `ice/01%20HTML/01%20Basicos/009%20-%20formularios.html?buscar=&Email=miEmail@miEmailcom&Edad=18`. Below the address bar, there is a form with an input field labeled "Email:" containing the text "miEmail@miEmailcom" and a button labeled "Aceptar". Below this, there is another input field labeled "Edad:" containing the text "18" and another button labeled "Aceptar".

Você pode ver como os dados inseridos no formulário aparecem na URL da página quando ela é enviada ao servidor. Este método é muito pouco confiável, pois é fácil de interceptar os dados se você souber como. Portanto, sempre que formulários são enviados, o

método ideal é o método **post**, onde os dados inseridos não são visíveis na URL.

Enctype, permite que você carregue arquivos para o servidor. Pode ser:

```
<form enctype="application/x-www-form-urlencoded"></form>
```

Ou también:

```
<form enctype="multipart/form-data"></form>
```

Este último significa que você pode enviar arquivos do formulário para o servidor, carregar arquivos... etc.

Com **accept**, indicamos que tipo de arquivos nos permite fazer upload:

```
<form enctype="multipart/form-data" accept-charset="UTF-8"></form>
```

Formas de envio de dados a partir de um formulario html. Métodos get e post

Quando um usuário preenche um formulário em uma página web, os dados têm que ser enviados de alguma forma. Vamos considerar as duas maneiras possíveis de enviar dados: usando o método **POST** ou usando o método **GET**.

Por exemplo:

```
<form action="http://www.curso de HTML.com/prog/newuser" method="get">
```

No exemplo acima, a ação que será executada quando o usuário clicar no botão "Enviar" (**submit**) será enviar os dados para a url especificada usando o método **get**.

Como já mencionamos, a diferença entre os métodos **get** e **post** reside na forma como os dados são enviados para a página quando o botão "Submit" é clicado. Enquanto o método **get** envia os dados usando o URL, o método **post** envia-os de uma forma que não podemos vê-los (no fundo ou "escondidos" do usuário).

Um resultado usando o método **get**, como exemplo, poderia ser assim:

<http://www.CursodeHTML.com/newuser.php?nome=Pepe&apellido=Flores&email=h52turam%40uco.es&sexo=Mujer>

Neste **URL** podemos distinguir várias partes:

<http://www.CursodeHTML.com/newuser.php>

é o próprio website.

O símbolo **?** indica onde começam os parâmetros que são recebidos do formulário que enviou os dados para a página.

O símbolo **?** é seguido por pares de dados com seu nome e valor separados pelo símbolo **&**.

Os pares data1=valor1, data2=valor2, data3=valor3... refletem o nome e o valor dos campos enviados pelo formulário.

Por exemplo: name=Pepe, sobrenome=Flores, etc. nos diz que o campo do formulário nomeado nome chega com o valor "Pepe" enquanto o campo do formulário nomeado sobrenome chega com o valor "Flores". Estes valores são recebidos na página web de destino do formulário.

Note que para separar o primeiro par do próprio endereço web usamos o símbolo **'?'** e para separar os pares restantes um do outro usamos o símbolo **'&'**.

Entrada de dados em formulários

Os controles de entrada de dados em formulários são geralmente controles visuais e permitem que o usuário insira dados ou selecione opções. Seu uso depende do tipo de controle e também do tipo de informação que eles podem recuperar.

Os elementos de entrada de um formulário podem ser definidos através do uso desses elementos:

HTML input, HTML textarea, HTML select e outros elementos **HTML**.

O resto das etiquetas já vão dentro da forma, entre outras, temos:

Text

```
<input type="text" name="miFormulario"
id="etiquetaText" value="Angel" >
```

Cria uma caixa para que o usuário insira os dados e os envie. Com **textarea**, fazemos o mesmo, mas criamos uma janela maior. Para enviar os dados:

```
<input type="submit"> // Com isto crio o botão de envio.
```

O **name** é o nome ao qual nos referiremos ao enviá-lo ao servidor, ou seja, se o nomearmos myForm, o recuperaremos com **\$_post("miFormulario")** ou com **\$_get("miFormulario")**

Com **value** mudamos o texto que aparece no botão.

```
<input type="submit" value="Aceptar">
```

Por exemplo:

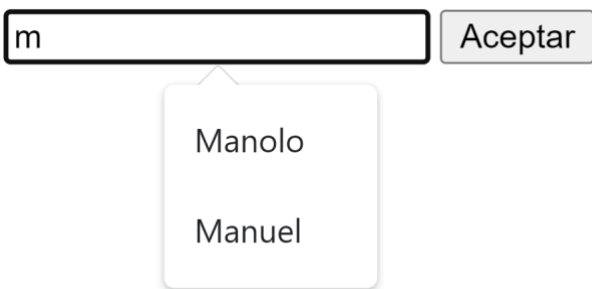
```
<form>
  <input type="text" name
="miFormulario" id="etiquetaText"
value="Angel">
  <input type="submit" value="Aceptar">
</form>
```

Ele criaria um pequeno formulário com uma etiqueta de texto (cujo valor padrão é "Angel") e um botão de submissão (**submit**) ao qual definimos o valor padrão para "Accept" (Aceitar)



Os elementos de tipo **text** geralmente têm um comportamento auto-completo por padrão, ou seja, eles geralmente mantêm as entradas anteriores na memória, por exemplo, neste caso, já preenchemos o formulário duas vezes antes com os valores "Manolo" e "Manuel".

Quando começamos a preencher o formulário pela terceira vez, o **HTML** sugere as entradas anteriores:



Para mudar isso, temos duas etiquetas disponíveis:

- Autocomplete(on/off).
- Novalidate(boolean). Se o definirmos, dizemos a ele para não avaliar o formulário.

```
<form name="formulario" id="formulario"
method="get" autocomplete="off">
```

Desta forma, não armazenará mais as entradas anteriores na memória.

A etiqueta **type** em **HTML5** já está preparada para tomar e validar os seguintes valores:

- **Text**: Etiqueta e tipo de texto genérico.
- **Email**: Etiqueta de tipo de endereço de e-mail.
- **Search**: Etiqueta do tipo de busca.
- **URL**: Etiqueta do tipo de endereço da página WEB.
- **Tel**: Etiqueta de tipo de número telefônico.
- **Number**: Etiqueta de tipo de número genérico. Juntamente com MIN, MAX e STEP.
- **Range**: Etiqueta do tipo faixa de valores. Juntamente com MIN, MAX e STEP.
- **Date**: Tipo de data.
 - **Week**.
 - **Month**.
 - **Time**.
 - **Datetime**.

No exemplo a seguir, implementamos um formulário de amostra que inclui as etiquetas acima mencionadas:

```
<form name="formulario" id="formulario"
method="get" autocomplete="off">
  Email: <input type="email"
name="mail" id="mail">
  <input type="submit"
value="Aceptar"><br><br>
  Edad;
  <input type="number" name="edad"
id="edad" min="18" max="100">
  <input type="submit"
value="Aceptar"><br><br>
  Buscar:
  <input type="search" name="buscar"
id="buscar"><br><br>
  URL:
  <input type="url" name="URL"
id="URL"><br><br>
  Edad:
```

```

<input type="number" name="edad"
id="edad" min="0" max="100" step="5"
placeholder="Introduce edad"><br><br>
Rango: <input type="range" min="0"
max="100" step="5"><br><br>
Fecha: <input type="date"
name="fecha" id="fecha"><br><br>
Hora: <input type="time" name="hora"
id="hora" required><br><br>
Código postal: <input pattern="(0-
9){5}" placeholder="5 dígitos" br><br>
<input type="submit"
value="Enviar"><br><br>
</form>

```

Otras etiquetas:

Com [placeholder](#) adiciono um texto que dá ao usuário uma dica de como preencher um campo. Ela desaparecerá quando o usuário começar a digitar.

Com o atributo [required](#), definimos um campo como obrigatório.

Com o atributo [multiple](#), podemos enviar mais de um dado dentro de um campo. Os dados são separados por vírgulas.

[Autofocus](#) traz o "foco" para o elemento de sua escolha.

[Pattern](#) personaliza o campo, definindo faixas personalizadas. Ou seja, permite-nos incluir [expresiones regulares](#) dentro do **input**.

[Form](#) constrói um elemento da forma fora da própria forma. Preciso acrescentar o nome do meu formulário:

Email2: `<input type="email2" name="mail2" id="mail2" form="formulario">`

[Datalist](#): Cria uma lista de itens a partir da qual o usuário terá que escolher um.

```

<form name="formulario" method="get">
  <datalist id="informacion">
    <option value="952773366"
label=" Telefono 1"></option>
    <option value="952734366"
label=" Telefono 2"></option>
    <option value="952773236"
label=" Telefono 3"></option>
  </datalist>
  Telefono:
  <input type="tel" name="telefono"
id="telefono" list="informacion">
</form>

```


Telefono:

952773366
Telefono 1

952734366
Telefono 2

952773236
Telefono 3

Label

Até agora, sempre que queríamos colocar uma mensagem antes ou depois de um controle de formulário, nós apenas a escrevíamos.

Há um elemento em **HTML** que permite associar um texto a um controle de formulário. Isto será muito útil se você estiver acessando de um navegador não gráfico ou uma pessoa cega usando um programa que lê em voz alta o conteúdo da página.

Vamos ver como o fizemos até agora:

Digite seu nome:

```
<input type="text" name="nombre"
size="20">
```

Usando o elemento **label**, podemos fazer uma referência entre o texto e o controle de entrada:

```
<label for="nombre">Ingrese su
nombre:</label>
<input type="text" name="nombre"
size="20" id="nombre">
```

Vamos ver o que adicionamos:

Definimos um **id** para a marca **input**. Ou seja, nós lhe demos um "pseudônimo" com o qual podemos nos referir a ele.

O elemento **label** tem sua etiqueta de abertura e fechamento, no meio está o texto a ser exibido.

Para vincular esta **label** com o elemento **input**, devemos rubricar o propriedade **for** com o nome atribuído à propriedade **id** do elemento **input**.

Vamos montar um exemplo completo:

```
<form action="registrardatos.php"
method="post">
  <fieldset>
    <legend>Formulario de
comentarios.</legend>
    <label for="nombre">Ingrese su
nombre:</label>
    <input type="text" name="nombre"
size="30" id="nombre"><br>
    <label for="mail">Ingrese su
mail:</label>
    <input type="text" name="mail"
size="50" id="mail"><br>
    <label
for="comentarios">Comentarios:</label><br>
    <textarea name="comentarios"
rows="5" cols="60" id="comentarios">
    </textarea>
    <br>
    <input type="submit"
value="Enviar">
  </fieldset>
</form>
```

O resultado no navegador é:

Formulario de comentarios.

Ingrese su nombre:

Ingrese su mail:

Comentarios:

Como podemos ver, associamos cada etiqueta com o controle de entrada correspondente:

```
<label for="nombre">Ingrese su
nombre:</label>
<input type="text" name="nombre"
size="30" id="nombre"><br>
```

```
<label for="mail">Ingrese su
mail:</label>
<input type="text" name="mail"
size="50" id="mail"><br>
```

```
<label
for="comentarios">Comentarios:</label><br
>
<textarea name="comentarios" rows="5"
cols="60" id="comentarios">
</textarea>
```

Normalmente, as propriedades `id` e `name` dos controles de entrada (`input`, `textarea`...) recebem o mesmo nome, embora não seja obrigatório.

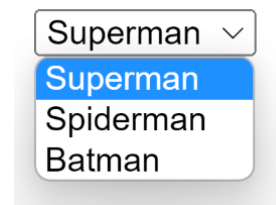
A propriedade `for` da etiqueta `label` se refere à `id` do controle e não ao `name`, isto é importante se rubricarmos a `id` e `name` dos controles com valores diferentes.

Select

O elemento `<select>` define uma lista suspensa a partir da qual o usuário escolherá uma das opções que lhe oferecemos:

```
<form action="/action_page.php"
method="get" enctype="multipart/form-
data" accept-charset="UTF-8">
  <select name="peliculas">
    <option
value="Superman">Superman</option>
    <option
value="Spiderman">Spiderman</option>
    <option
value="Batman">Batman</option>
  </select>
</form>
```

Daria como saída:



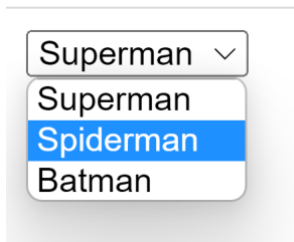
A etiqueta `value` é para que o **backend** receba o valor e normalmente é definida para o mesmo valor que no `<option>`

Os elementos `<option>` define uma opção que pode ser selecionada.

Por padrão, o primeiro item da lista suspensa é selecionado.

Para definir una opción pré-seleccionada, adicione o atributo `selected` à opção:

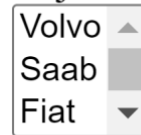
```
<form action="/action_page.php"
method="get" enctype="multipart/form-
data" accept-charset="UTF-8">
  <select name="películas">
    <option
value="Superman">Superman</option>
    <option value=" Spiderman "
selected> Spiderman</option>
    <option
value="Batman">Batman</option>
  </select>
</form>
```



Usaremos o atributo `size` para especificar o número de valores visíveis:

```
<form action="/action_page.php"
method="get" enctype="multipart/form-
data" accept-charset="UTF-8">
  <label for="cars">Elija un modelo de
coche:</label><br>
  <select id="cars" name="cars"
size="3">
    <option
value="volvo">Volvo</option>
    <option value="saab">Saab</option>
    <option value="fiat">Fiat</option>
    <option value="audi">Audi</option>
  </select>
</form>
```

Elija un modelo de coche:



Use o atributo `multiple` para permitir que o usuário selecione mais de um valor:

```
<form action="/action_page.php"
method="get" enctype="multipart/form-
data" accept-charset="UTF-8">
  <label for="cars">Choose a
car:</label>
  <select id="cars" name="cars"
size="4" multiple>
    <option
value="volvo">Volvo</option>
    <option value="saab">Saab</option>
    <option value="fiat">Fiat</option>
    <option value="audi">Audi</option>
  </select>
</form>
```

Textarea

Com o elemento `text` definimos uma área de entrada de texto de uma linha.

O elemento `<textarea>` define um campo de entrada com várias linhas (uma área de texto):

```
<form action="" method="get">
  <textarea name="message" rows="10"
cols="30">
    Aquí el usuario puede escribir un
texto más extenso que si usáramos el
atributo text.
  </textarea>
</form>
```

Aquí el usuario puede escribir un texto más extenso que si usáramos el atributo text.

O atributo `rows` especifica o número visível de linhas em uma área de texto.

O atributo `cols` especifica a largura visível de uma área de texto.

É assim que o código HTML acima será exibido em um navegador:

Você também pode definir o tamanho da área de texto usando o CSS:

```
<form action="" method="get">
  <textarea name="message"
style="width:200px; height:600px;">
    Aquí el usuario puede escribir un
    texto más extenso que si usáramos el
    atributo text.
  </textarea>
</form>
```

Aquí el usuario puede escribir un texto más extenso que si usáramos el atributo text.

Checkbox

As `checkbox` são controles de formulário que permitem que o usuário selecione e desmarque as opções individualmente. Embora várias `checkbox` sejam às vezes exibidas juntas, cada uma é completamente independente das outras. Por este motivo, eles são usados quando o usuário pode ativar e desativar várias opções relacionadas, mas não mutuamente exclusivas.

Puestos de trabajo buscados

- ☐ Dirección
- ☐ Técnico
- ☐ Empleado

```
<form action="" method="get">
  Puestos de trabajo buscados <br/>
  <input name="puesto_directivo"
type="checkbox" value="direccion"/>
Dirección<br>
  <input name="puesto_tecnico"
type="checkbox" value="tecnico"/>
Técnico<br>
  <input name="puesto_empleado"
type="checkbox" value="empleado"/>
Empleado<br>
</form>
```

Puestos de trabajo buscados

- ☐ Dirección
- ☐ Técnico
- ☐ Empleado

El valor del atributo `type` para estos controles de formulario es `checkbox`. Como mostrado no exemplo acima, o texto ao lado de cada `checkbox` não pode ser definido por nenhum atributo, portanto, ele precisa ser adicionado manualmente fora do controle do formulário. Se nenhum texto for adicionado ao lado do `<input />` de `checkbox`, o usuário vê apenas um pequeno quadrado sem qualquer informação a respeito da finalidade da `checkbox`.

O valor do atributo `value`, junto com o valor do atributo `name`, é a informação que chega ao servidor quando o usuário submete o formulário.

Se você deseja exibir uma `checkbox` selecionada por padrão, o atributo `checked` é utilizado. Se o valor do atributo for `checked`, a `checkbox` é mostrada selecionada. Caso contrário, a `checkbox` permanece desmarcada. Embora seja redundante que o nome do atributo e o valor sejam idênticos, é obrigatório indicá-lo desta forma porque os atributos em **XHTML** não podem ter valores vazios:

```
<input type="checkbox"
checked="checked" ... /> Checkbox
seleccionado por defecto
```

- ☒ Checkbox seleccionado por defecto

Exemplo:

```
Selecciona tus intereses:<br/>
<input name="Películas" type="checkbox"
/>Películas<br/>
<input name="Libros" type="checkbox"
checked="checked"/>Libros<br/>
<input name="Deportes" type="checkbox"
/>Internet<br/>
```

Selecciona tus intereses:

- ☐ Películas
- ☒ Libros
- ☐ Deportes

Radiobutton

Os controles `radiobutton` são similares aos controles por `checkbox`, mas têm uma diferença muito importante: eles são mutuamente exclusivos. Os `radiobutton` são usados quando o usuário só pode escolher uma opção entre várias opções relacionadas apresentadas a ele. Cada vez que uma opção é selecionada, a outra opção que foi selecionada é automaticamente desmarcada.

```
Sexo: <br/>
<input type="radio" name="sexo"
value="hombre"/>Hombre<br/>
<input type="radio" name="sexo"
value="mujer" />Mujer
```

Sexo:

- ☒ Hombre
- ☐ Mujer

O valor do atributo `type` para estes controles de formulário é o `radio`. O atributo do `name` é usado para indicar quais `radiobutton` estão relacionados. Portanto, quando vários `radiobutton` têm o mesmo valor no atributo de seu `name`, o navegador sabe que eles estão relacionados e pode desmarcar uma opção no grupo de `radiobutton` quando outra opção é selecionada.

Botão submit

A maioria dos formulários tem um botão para enviar os dados inseridos pelo usuário para o servidor:

```
<input type="submit" name="buscar" value="Buscar" />
```

Buscar

O valor do atributo `type` para este controle do formulário é `submit`. O navegador envia automaticamente os dados quando o usuário clica neste tipo de botão. O valor do atributo `value` é o texto exibido pelo botão. Se o atributo `value` não estiver definido, o navegador exibe o texto pré-definido `Enviar`.

Botão de reset

Embora seu uso fosse muito popular há alguns anos, a maioria das formas modernas não usa mais este tipo de botão. É um botão especial que apaga todos os dados inseridos pelo usuário e retorna o formulário ao seu estado original:

```
<input type="reset" name="limpiar" value="Borrar datos" />
```

Borrar datos

O valor do atributo `type` para este controle de formulário é `reset`. Quando o usuário clica neste botão, o navegador limpa todas as informações inseridas e exibe o formulário em seu estado original. Se o formulário originalmente não continha nenhum valor, o botão `reset` retorna o formulário a um estado vazio. Se o formulário continha informações, o botão de `reset` exibe novamente as informações originais.

Como de costume para os botões de formulário, o atributo `value` permite definir o texto que o botão exibe. Se este atributo não for utilizado, o navegador exibe o texto predefinido do botão, que neste caso é `Reset`.

Botão de imagem

A aparência dos botões do formulário pode ser completamente personalizada, pois é até possível usar uma imagem como um botão:

```
<input type="image" name="enviar" src="accept.png" />
```



O valor do atributo `type` para este controle de formulário é `image`. O atributo `src` indica a URL da imagem a ser exibida pelo navegador ao invés do botão normal.

Sua principal vantagem é que permite personalizar completamente a estética dos botões e exibi-los com uma aparência homogênea em todos os navegadores. A principal desvantagem é que ele retarda o carregamento do formulário e que, se você quiser mudar sua aparência, é necessário criar uma nova imagem.

<fieldset> e <legend>

O elemento `<fieldset>` é usado para agrupar dados relacionados em um formulário.

O elemento `<legend>` define um título para o elemento `<fieldset>`.

Exemplo:

```
<form action="/action_page.php">
  <fieldset>
    <legend>Datos de contacto:</legend>
    <label
for="name">Nombre:</label><br>
    <input type="text" id="name"
name="name" value="Andrés"><br>
    <label
for="apellido">Apellido:</label><br>
    <input type="text" id="apellido"
name="apellido" value="García">
<br><br>
    <input type="submit"
value="Submit">
  </fieldset>
</form>
```

É assim que o código HTML acima será exibido em um navegador:

Arquivos anexos

Os formulários também permitem anexar arquivos para upload para o servidor. Embora do ponto de vista do **HTML** e do navegador não haja limitação quanto ao número, tipo ou tamanho total dos arquivos que podem ser anexados, todos os servidores acrescentam restrições por razões de segurança.

```
Fichero adjunto
<input type="file" name="adjunto" />
```

Fichero adjunto

O valor do atributo `type` para este controle de formulário é `file`. O navegador é responsável por exibir uma caixa de texto com o nome do arquivo selecionado e um botão para navegar pelos diretórios e arquivos no computador do usuário.

Se você incluir um controle de arquivo anexo, é obrigatório adicionar o atributo `enctype` no `<form>` do formulário. O valor do atributo `enctype` deve ser `multipart/form-data`, então a etiqueta `<form>` dos formulários que permitem anexos de arquivos é sempre:

```
<form action="..." method="post"
enctype="multipart/form-data">
  ...
</form>
```

Campos ocultos

Os campos ocultos são usados para adicionar informações ocultas ao formulário:

```
<input type="hidden" name="url_previa"
value="/articulo/primer.html"/>
```

O valor do atributo `type` para este controle de formulário é `hidden`. Os campos ocultos não são exibidos na tela, portanto o usuário não sabe que o formulário os inclui. Normalmente são usados campos ocultos para incluir informações que são necessárias ao servidor, mas que não são necessárias ou possíveis de serem definidas pelo usuário.

Iframes

Um `iframe` ou frame flutuante **HTML** é usado para exibir uma página da web dentro de uma página da web. Por exemplo, para mostrar um vídeo do youtube dentro de nossa página web, como vimos na seção de vídeos.

Sintaxe:

```
<iframe src="url"
title="description"></iframe>
```

Sugestão: é uma boa prática incluir sempre um atributo de `title` para o arquivo `<iframe>`. Os leitores de tela utilizam-no para ler o conteúdo do `iframe`.

Usaremos os atributos `height` e `width` para especificar a largura e a altura do `iframe` ou, se não for o caso, o faremos com CSS.

A altura e a largura são especificadas em pixels por padrão:

Exemplo:

```
<iframe src="demo_iframe.htm"
height="200" width="300"
title="Ej_Iframe"> </iframe>
```

A maioria dos vídeos que importamos do youtube estará neste formato `Iframe`.

Vias em HTML

Um caminho de arquivo descreve a localização de um arquivo na estrutura de pastas de um website.

- ``: O arquivo "imagen.jpg" está localizado na mesma pasta que a página atual.
- ``: O arquivo "imagen.jpg" está localizado na pasta de imagens da pasta atual.
- ``: O arquivo "imagen.jpg" está localizado na pasta de imagens na raiz do site atual.
- ``: O arquivo "image.jpg" está localizado na pasta um nível acima da pasta atual.

Os caminhos de arquivos são usados quando se ligam a arquivos externos, tais como:

- páginas web
- Imagens, vídeos ou áudio.
- folhas de estilo
- JavaScripts

Vias de arquivo absolutas

Um caminho de arquivo absoluto é a URL completa de um arquivo:

```

```

Vias de arquivo relativas

Um caminho de arquivo relativo aponta para um arquivo relativo à página atual.

No exemplo a seguir, o caminho do arquivo aponta para um arquivo na pasta de imagens localizada na raiz da página web atual:

```

```

Desta vez, devemos ter a imagem chamada picture.jpg no diretório que especificamos "imagens" em nosso disco rígido ou servidor.

No exemplo a seguir, o caminho do arquivo aponta para um arquivo na pasta de imagens localizada na pasta um nível acima da pasta atual:

```

```

Recomenda-se o uso de caminhos de arquivos relativos (se possível).

Ao utilizar caminhos de arquivos relativos, suas páginas web não serão vinculadas à sua URL base atual. Todos os links funcionarão em seu próprio computador (localhost), assim como em seu domínio público atual e em seus domínios públicos futuros.

Apêndice Etiquetas HTML

Etiqueta	Função
<!--...-->	Define um comentário
<!DOCTYPE>	Define o tipo de documento
<a>	Define um hyperlink
<abbr>	Define uma abreviação
<address>	Define as informações de contato do autor/proprietário do documento. Deve ir para dentro do footer.
<area>	Define uma área dentro de um mapa de imagem
<article>	Define um artigo. Importante para SEO.
<aside>	Define o conteúdo lateral do recipiente de uma página. Importante para SEO.
<audio>	Define conteúdo de som
	Define texto em negrito
<base>	Especifica a base onde todas as URLs do documento serão abertas
<bdi>	Isola uma parte do texto que pode ser formatada de forma diferente do texto externo
<bdo>	Sobregavar o endereço do texto
<blockquote>	Define uma seção que tem outra fonte. Muitas vezes utilizado para citações longas.
<body>	Define o corpo do documento
 	Define uma quebra de linha
<button>	Define um botão clicável

<canvas>	Usado para desenhar gráficos na tela
<caption>	Define o título de uma tabela
<cite>	Define o título de uma obra. Livros, filmes...
<dl>	Define uma lista de definição
<dt>	Define um termo (um item) em uma lista de definição
	Define ênfase em um texto e o itálico. Substitui o antigo <i></i>
<embed>	Define o recipiente de uma aplicação externa (não html)
<fieldset>	Grupo de elementos relacionados de uma forma
<figcaption>	Define o título de uma figura <figure> Importante para SEO.
<figure>	Especifica o auto-conteúdo Importante para SEO.
<footer>	Define o rodapé de um documento Importante para SEO.
<form>	Define um formulário html
<h1> a <h6>	Define cabeçalhos ou títulos Importante para SEO.
<head>	Define informações sobre o documento
<header>	Define a seção do cabeçalho do documento Importante para SEO.
<hgroup>	Grupo de cabeçalho (<h1> a <h6>) Importante para SEO.
<hr>	Define uma mudança de tema a partir de uma linha traçada
<html>	Define a raiz do documento
<i>	Define uma parte do texto em alternativa
<iframe>	Define um frame online
	Define uma imagem
<input>	Define um controle de entrada de texto
<ins>	Define texto que foi inserido em um documento
<kbd>	Define entrada de teclado

<keygen>	Define um campo gerador chave para formulários
<label>	Define o rótulo de um elemento <input>
<legend>	Define um título para os elementos <fieldset>, <figure>, <details>
	Define um item de uma lista
<link>	Define a relação entre um documento e um recurso externo (geralmente com folhas de estilo)
<map>	Define m mapa de imagem do cliente
<mark>	Define texto destacado ou marcado. Efeito "marcador fluorescente".
<menu>	Define a lista de um menu
<meta>	Define um metadado de um documento
<meter>	Define uma medida escalar em uma faixa conhecida
<nav>	Define um link de navegação Importante para SEO.
<noscript>	Define conteúdo alternativo para usuários que não suportam scripts de clientes.
<objet>	Define um objeto incorporado
	Define uma lista ordenada
<optgroup>	Define um grupo de opções relacionadas em uma lista suspensa
<option>	Define uma opção em uma lista suspensa
<output>	Define o resultado de um cálculo
<p>	Define um parágrafo
<param>	Define um parâmetro para um objeto
<pre>	Define texto pré-formatado
<progress>	Representa o progresso de uma tarefa em uma barra
<q>	Define uma breve citação
<rp>	Define para exibir em navegadores que não suportam scripts de ruby

<rt>	Define uma pronúncia de caracteres
<ruby>	Define uma notação de ruby
<s>	Define texto que não está correto
<samp>	Define um exemplo de resultados do programa
<script>	Define um script do lado do cliente
<section>	Define uma seção de um documento Importante para SEO.
<select>	Define um drop-down list
<small>	Define textos legais.
<source>	Define recursos para elementos multimídia
	Define uma pequena seção de um documento
	Define um texto em negrito e o trata como importante. Importante para SEO
<style>	Define um estilo para as informações de um documento
<sub>	Define um texto que é subscrito
<summary>	Define um cabeçalho visível para o elemento <details>
<sup>	Define um texto que é superescrito
<table>	Define uma tabela
<tbody>	Define o corpo de uma tabela
<td>	Define uma célula em uma tabela
<textarea>	Define um controle de entrada de múltiplas linhas
<tfoot>	Conteúdo do rodapé do grupo em uma tabela
<th>	Define uma célula de cabeçalho em uma tabela
<thead>	Agrupar os títulos de uma tabela
<time>	Define data / hora. Ex: <code><time datetime="2018/01/31" pubdate>Notícia publicada el día...</time></code>
<title>	Define um título para o documento. Importante para SEO.

<tr>	Define uma fileira em uma tabela
<track>	Define texto de faixa para elementos multimídia (vídeo e áudio)
	Define uma lista não-ordenada
<var>	Define uma variável
<video>	Define um vídeo ou filme
<wbr>	Define uma possível quebra de linha