Características avanzadas de HTML y CSS

Características avanzadas

En esta etapa, que ya somos capaces de crear sin problemas un sitio web sencillo. Vamos a descubrir nuevas características de HTML y CSS, que pueden considerarse un poco más avanzadas (y por lo tanto más complejas).

Tablas

Esenciales para la organización de la información, las tablas son un poco difíciles de construir en HTML. En efecto, tendrás que incorporar nuevas etiquetas HTML en un orden específico.

Vamos a comenzar por la construcción de cuadros básicos, entonces los complicaremos con: fusión de celdas, división de celdas en varias secciones... También descubriremos las propiedades de CSS relacionadas con tablas, que nos permiten personalizar su apariencia.

- Tabla simple

La primera etiqueta es <table> </ table>. Es esta etiqueta la que se usa para indicar el principio y el fin de una tabla. Esta etiqueta es de bloque, por lo que debe ser colocada fuera de un párrafo.

Código HTML

```
<p>Este es un párrafo previo a la tabla.</p>
<table>
<!-- Aquí se escribirá el contenido de la tabla-->
</table>
<p>Este es un párrafo al final de la tabla.</p>
```

¿Y que se escribe dentro de la tabla?

Entonces prepárate para experimentar una avalancha de nuevas etiquetas. Para iniciarla sin problemas, hay dos nuevas etiquetas muy importantes:

<tr> </tr>: indica el comienzo y el final de una línea de la tabla;
<td> </td>: indica el inicio y el final de los contenidos de una celda.

En HTML, un cuadro se construye línea por línea. En cada fila (<tr>), se muestra el contenido de las celdas individuales (<td>).

Básicamente, una tabla se construye como en la siguiente figura:

Nombre producto	Precio unitario	Unidades	Subtotal
Reproductor MP3 (80 GB)	192.02	1	192.02
Fundas de colores	2.50	5	12.50
Reproductor de radio & control remoto	12.99	1	12.99
TOTAL	-	7	**207.51**

Por ejemplo, si quiero hacer una tabla con dos filas, con tres celdas por línea (por lo tanto tres columnas), tendrías que escribir lo siguiente:

Código: HTML

```
<table>
<tr>
        <td>Carmen</td>
        <td>33 años</td>
        <td>España</td>
</tr>
<tr>
        <td>Miguel</td>
        <td>26 años</td>
        <td>Estados Unidos</td>
</tr>
</table>
```

El resultado sería el siguiente:

Carmen	33 Años	España
Miguel	26 Años	Estados Unidos

Una tabla sin CSS parece vacía. Y sólo tienes que añadir los bordes, es muy simple, ya conoces el código CSS correspondiente.

Código CSS

```
td / * Todas las celdas de la tabla ... * /
{
border: 1px solid negro; / * tienen un borde de 1 píxel * /
}
```

y resultaría con bordes la tabla mostrada previamente.

La presentación todavía no es tan “perfecta” como lo deseas. En efecto, nos gustaría que sólo aparezca un borde entre dos celdas, lo que no es el caso que resulta. Afortunadamente, hay una propiedad CSS específica para las tablas, `border-collapse`, que significa "juntar los bordes entre ellos ".

Esta propiedad puede tener dos valores:

`collapse`: los bordes se pegan entre ellos, este es el efecto que estamos buscando;
`separate`: los bordes están separados (por defecto)

Código CSS:

```
table
{
border-collapse: collapse; /* Los bordes de la tabla aparecen juntos */
}
td
{
border: 1px solid black;
}
```

Siendo el resultado una tabla como la primera mostrada en este capítulo.

- Línea de cabecera de las tablas

Ahora que hemos conseguido lo que queríamos, vamos a añadir la línea en el encabezado de la tabla. En el ejemplo, las cabeceras son "Nombre","Edad" y "País".

La línea de cabecera se crea con la etiqueta <tr> como se ha hecho hasta ahora, pero las celdas que contiene, esta vez están enmarcadas por la etiqueta <th> y no la <td>

Código HTML

```
<table>
<tr>
        <th>Nombre</th>
        <th>Edad</th>
        <th>País</th>
</tr>

<tr>
```

```
        <td>Carmen</td>
        <td>33 años</td>
        <td>España</td>
</tr>
<tr>
        <td>Miguel</td>
        <td>26 años</td>
        <td>Estados Unidos</td>
</tr>
</table>
```

El encabezado de la fila es muy fácil de reconocer por dos razones:

- las celdas etiquetadas con <th> en lugar del <td> usual;
- es la primera línea de la tabla (es una tontería, pero debe especificarse).

Como el nombre de las celdas es un poco diferente en la cabecera, en necesario pensar en el CSS para decirle que se aplica borde en las celdas normales y la cabecera.

Código CSS

```
table
{
border-collapse: collapse;
}
td, th /* Poner borde en td y th */
{
border: 1px solid black;
}
```

Como puedes ver en la figura siguiente, el navegador pone en negrita el texto de la celda de cabecera. Esto es lo que suelen hacer de forma general los navegadores, pero si lo deseas, puedes cambiar esto a golpe de CSS: cambiar el color de fondo de las celdas de cabecera, fuente, borde, etc.

Nombre	Edad	Nación
Carmen	33 años	España
Miguel	26 años	Estados Unidos

- Título de la tabla

Normalmente, todas las tablas deben tener un título. El título permite al visitante saber rápidamente el contenido de la tabla. Nuestro ejemplo se compone de una lista de personas, sí, pero ¿y qué? ¿qué significa eso? Sin título, estamos un poco perdidos. Afortunadamente, existe la etiqueta <caption>.

Esta etiqueta se coloca al principio de la tabla, justo antes de la cabecera. Es ella quien tiene el título de la tabla (figura siguiente):

Código HTML

```
<table>
<caption>Pasajeros del vuelo Murcia-Madrid</caption>
<tr>
        <th>Nombre</th>
        <th>Edad</th>
        <th>País</th>
</tr>

<tr>
        <td>Carmen</td>
        <td>33 años</td>
        <td>España</td>
</tr>
<tr>
        <td>Miguel</td>
        <td>26 años</td>
        <td>Estados Unidos</td>
</tr>
</table>
```

Ten en cuenta que puedes cambiar la posición del título con la propiedad CSS `caption-side` que puede tomar cuatro valores:

`top`: el título se coloca encima de la tabla (por defecto);
`botton`: el título se coloca por debajo de la tabla.

Tablas estructuradas

Aprendimos a construir pequeñas tablas simples. Estas tablas pequeñas son suficientes en la mayoría de los casos, pero no sé qué necesidades tienes de hacer tablas más complejas. Vamos a explorar dos técnicas específicas:

Para tablas grandes, es posible dividirlas en tres partes:

- Cabecera;
- Cuerpo de la tabla;
- Pie.

Para algunas celdas, puede ser que tengas que fusionar celdas adjuntas.

- Dividir una tabla de gran tamaño

Si la tabla es lo suficientemente grande, vas a hacer bien en dividirla en varias partes. Para ello, existen etiquetas HTML que definen las tres "zonas" de la tabla:

- encabezado (arriba): se define con las etiquetas <thead> y </ thead>;
- cuerpo (centro): se define con las etiquetas <tbody> y </ tbody>;
- pie (abajo): se define con las etiquetas <tfoot> y </ tfoot>.

¿Qué poner en el pie de la tabla? En general, si se trata de una tabla larga, copiarás las celdas del encabezado. Esto permite ver, incluso desde la parte inferior de la tabla, a que se refiere cada columna. Esquemáticamente, una tabla de tres partes se muestra como:

Pasajeros del vuelo Murcia-Madrid

Nombre	Edad	Nación
Carmen	33 años	España
Miguel	26 años	Estados Unidos
Nombre	**Edad**	**Nación**

Es un poco confuso, pero es aconsejable escribir las etiquetas en el siguiente orden:

1. <thead>
2. <tfoot>
3. <tbody>

En el código, lo que aparece primero es la parte superior, y la parte inferior, y, finalmente, la parte principal (<tbody>). El navegador se encargará de mostrar cada elemento en el lugar correcto, no te preocupes.

Por lo tanto se ha de escribir el código para crear la tabla en tres partes:

```
<table>
```

```
<caption>Pasajeros del vuelo Madrid-Murcia</caption>
<thead> <!-- Encabezado de la tabla -->
<tr>
        <th>Nombre</th>
        <th>Edad</th>
        <th>País</th>
</tr>
<tr>
        <td>Carmen</td>
        <td>33 años</td>
        <td>España</td>
</tr>
<tr>
        <td>Miguel</td>
        <td>26 años</td>
        <td>Estados Unidos</td>
</tr>
<tr>
        <td>Francisco</td>
        <td>43 años</td>
        <td>Francia</td>
</tr>
<tr>
        <td>Martina</td>
        <td>34 años</td>
        <td>Francia</td>
</tr>
<tr>
        <td>Jonathan</td>
        <td>13 años</td>
        <td>Australia</td>
</tr>
<tr>
        <td>Xu</td>
        <td>19 años</td>
        <td>China</td>
</tr>
</tbody>
</table>
```

No es obligatorio el uso de estas tres etiquetas (<thead>, <tbody>, <tfoot>) en todas las tablas. De hecho, lo va a usar, especialmente si la tabla es lo suficientemente grande y es necesario para organizarla con mayor claridad. Para las "pequeñas" tablas, puedes mantener fácilmente

la organización más fácil de lo que vimos en principio.

Fusión en tablas

En algunas tablas complejas, tendrás que "fusionar" las celdas entre sí. ¿Un ejemplo de la fusión? Mira la tabla en la siguiente figura, que proporciona una lista de películas y programas e indica a quienes van dirigidas.

Pasajeros del vuelo Murcia-Madrid

Nombre	**Edad**	**Nación**
Carmen	33 años	España
Miguel	26 años	Estados Unidos
Total	3 viajeros	
Nombre	**Edad**	**Nación**

En la última película, se ve que las celdas se fusionan: se convierten en una. Este es exactamente el efecto buscado. Para fusionar, agregamos un atributo a <td>. Debes saber que hay dos tipos de fusión:

- **Fusión de columnas**: esto es lo que he hecho en el ejemplo. La fusión se realiza horizontalmente. Usamos el atributo `colspan`.
- **Fusión de líneas**: dos líneas se agrupan. La fusión se llevará a cabo verticalmente. Usamos el atributo `rowspan`.

Como sabes, se ha de proporcionar un valor para el atributo (`colspan` o `rowspan`). Debes indicar el número de las celdas que se fusionan. En nuestro ejemplo, fusionamos dos celdas: de la columna "Edad" y de la "Nación". Por lo tanto, vas a escribir:

Código HTML

```
<td colspan="2">
```

Código HTML que me permite lograr la fusión correspondiente a la tabla anterior:

```
<table>
<caption>Pasajeros del vuelo Murcia-Madrid</caption>
<tr>
<th>Nombre</th>
<th>Edad</th>
```

```
<th>Nación</th>
</tr>
<tr>
<td>Carmen</td>
<td>33 años</td>
<td>España</td>
</tr>
<tr>
<td>Miguel</td>
<td>26 años</td>
<td>Estados Unidos</td>
</tr>
<tr>
<td>Total</td>
<td colspan=2> 3 viajeros</td>
</tr>
<tr>
<th>Nombre</th>
<th>Edad</th>
<th>Nación</th>
</tr>
</table>
```

Una nota importante: se ve que la línea 20 sólo contiene dos celdas en lugar de tres (hay dos etiquetas <td>). Esto es bastante normal porque se fusionaron las dos últimas celdas contiguas. <td colspan="2"> indica que esta celda ocupa el lugar de dos celdas a la vez.

Y la fusión vertical con `rowspan`, ¿cómo lo hacemos?, se puede ver en el siguiente ejemplo:

Código : HTML

```
<table>
<tr>
<th>Ejemplo1</th>
<th>Ejemplo2</th>
<th>Ejemplo3</th>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td rowspan=2>3,6</td>
</tr>
```

```
<tr>
<td>4</td>
<td>5</td>
</tr>
</table>
```

Resultando que las celdas se fusionan verticalmente

Ejemplo1	Ejemplo2	Ejemplo3
1	2	3,6
4	5	

Resumen

- Una tabla se inserta con la etiqueta <table> y la líneas se definen con la etiqueta <tr>.
- Cada línea contiene celdas <td> (celdas normales) o celdas <th> (encabezado).
- El título de la tabla se define con <caption>.
- Puedes agregar un borde a las celdas de la tabla con `border`. Para fusionar los bordes, se utiliza la propiedad CSS border-collapse.
- Una tabla puede ser dividida en tres secciones: <thead> (cabecera) <tbody> (cuerpo) y <tfoot> (pie). El uso de estas etiquetas no es obligatorio.
- Puedes combinar celdas horizontalmente o verticalmente con `colspan` o `rowspan`.
- Debes indicar cuántas celdas deberán fusionarse.

Formularios

Cualquier página HTML puede ser enriquecida con formularios interactivos que invitan a los visitantes a transmitir información: entrar texto, seleccionar entre opciones, confirmar con un botón... todo es posible.

Sin embargo, llegamos a los límites de HTML, ya que entonces es necesario analizar la información que el visitante aportó y esto no se puede hacer en HTML. Como veremos, los resultados del tratamiento se deben hacer en otro lenguaje, como PHP.

Mientras tanto, tenemos un gran número de nuevas etiquetas HTML para descubrir. Bienvenido al maravilloso mundo de la formularios, un mundo donde los botones, casillas de verificación y listas desplegables viven en armonía (bueno, casi).

Crear un formulario

Cuando tienes la súbita necesidad de insertar un formulario en tu página HTML, es necesario empezar a escribir las etiquetas <form> </ form>. Esta es la etiqueta principal del formulario, permite indicar el principio y el fin.

Código HTML

```
<p>Texto anterior al formulario</p>
<form>
<p>Texto en el interior del formulario</p>
</form>
<p>Texto posterior al formulario</p>
```

Ten en cuenta que es obligatorio poner etiquetas de tipo bloque (como </p>) dentro del formulario si deseas incluir texto.

Para la estructura de base, debes prestar atención: tengo que decir que no es fácil porque estamos en el límite del código HTML.

Vamos a poner un ejemplo para aclarar las cosas. Supongamos que el visitante viene a escribir un comentario en un formulario, como por ejemplo un mensaje que te gustaría publicar en los foros. Este mensaje debe ser enviado al propietario de la web, y ser visualizado por los visitantes.

Estos son el problema, o problemas que se plantean:

- Problema # 1: ¿Cómo enviar el texto escrito por el visitante? ¿Con qué medios?
- Problema # 2: Una vez que los datos han sido enviados, ¿cómo tratarlos? ¿Quieres recibir el mensaje automáticamente por correo electrónico o prefieres un programa para cargar la partida de mensajes guardada en algún lugar, para a continuación, mostrarlos.

Para dar respuesta a estos dos problemas, debes agregar dos atributos a la etiqueta <form>:

`method`: Este atributo indica el medio por el cual se enviarán los datos (respuesta al problema # 1). Hay dos soluciones para enviar datos en la web:

- `method` = "`get`" este método suele ser bastante inadecuado, ya que está limitado a 255 caracteres. La característica es que la información se enviará a la dirección de la página (http://...), pero el detalle no tiene importancia en este momento. La mayoría de las veces, te recomiendo que utilices el otro Método: `post`.

- `method` = "`post`": este es el método más común para los formularios, ya que permite enviar un gran número de información. Los datos introducidos en el formulario no pasarán a través de la barra de direcciones.

`acction`: Es la dirección de la página o programa que procesará la información (respuesta al problema # 2). La página se cargará y enviará un mensaje de correo electrónico si es lo que quieres, o guardar el mensaje con los otros en una base de datos.

Esto no se puede hacer en HTML y CSS, por lo general se utiliza otro lenguaje: PHP.

Así que ahora vamos a completar la etiqueta <form> con dos atributos que acabamos de ver. `method` y `post`.

Para `acction`, voy a escribir el nombre de una página PHP ficticia (tratamiento.php). Esta es la página que se llama cuando el visitante hace clic en el botón de envío del formulario.

Código HTML

```
<p>Texto previo al formulario</p>
<form method="post" action="tratamiento.php">
        <p>Texto en el formulario</p>
</form>
<p>Texto después del formulario</p>
```

Por el momento, no sabemos lo que sucede en el interior de la página tratamiento.php: Estoy pidiendo confianza para imaginar que esta página está en marcha y funcionando.

Nuestra prioridad en este momento es descubrir en HTML / CSS cómo insertar cuadros de

texto, botones y casillas de verificación en la página web. Esto es lo que vamos a ver ahora.

Campos de entrada básicos

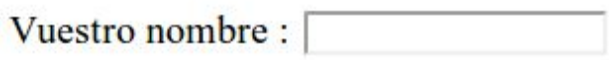
De vuelta a lo concreto. Vamos a revisar las diferentes etiquetas HTML para ingresar texto en un formulario. Debes saber que hay dos cuadros de texto diferentes:

- Cuadro de texto de una sola línea: Como su nombre indica, no se puede escribir más que una sola línea. Se utiliza para introducir texto corto.

- Cuadro de texto multilínea: Este cuadro de texto permite escribir una gran cantidad de texto en varias líneas, por ejemplo un ensayo de la utilidad de HTML en los países en desarrollo del Asia Sudoriental (solo es sugerencia).

- Cuadro de texto de una sola línea

La siguiente figura muestra lo que parece ser un cuadro de texto de una sola línea.



Para insertar un cuadro de texto en una línea, vamos a utilizar la etiqueta <input/>. Encontraremos etiquetas varias veces más adelante en este capítulo. En cada caso, es el valor del atributo `type` el que cambiará.

Para crear un cuadro de texto a una línea, se debe escribir:

Código HTML

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

Todavía no es suficiente: tenemos que dar nombre al cuadro de texto. Este nombre no aparece en la página, pero será posteriormente necesario. De hecho, esto permitirá (por ejemplo PHP) reconocer de dónde viene la información: sabes el texto y el nombre del visitante, dicho texto es la contraseña que se eligió, etc.

Para dar un nombre a un elemento de formulario, se utiliza el atributo `name`. Aquí, vamos a suponer que se pide al visitante su nombre:

Código HTML

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

Estamos tratando de crear un formulario muy básico con nuestro campo de texto:

```
<form method="post" action="tratamiento.php">
<p><input type="text" name="nombre" /></p>
</form>
```

- Etiquetas

Este cuadro de texto es muy bonito pero si el visitante se cae, no sabe qué escribir. Este es precisamente el papel de <label>:

Código HTML

```
<form method="post" action="tratamiento.php">
<p>
<label>Vuestro nombre</label> : <input type="text" name="nombre" />
</p>
</form>
```

Este código da exactamente el resultado que hemos visto en la figura anterior. Pero esto no es suficiente. Es necesario vincular la etiqueta a la caja de texto. Para ello, tenemos que dar un nombre al cuadro de texto, no con el atributo `name`, pero si con el atributo `id` (que puede ser utilizado en todas las etiquetas).

Para vincular la etiqueta al campo, se le debe dar un atributo `for` que tiene el mismo valor que la `id` del campo. Se aprecia mejor en un ejemplo:

Código: HTML

```
<form method="post" action="tratamiento.php">
<p>
<label for="pseudo">Vuestro nombre</label> : <input type="text"
name="nombre" id="nombre" />
</p>
</form>
```

Algunos atributos adicionales

Puedes agregar otros atributos al <input /> para personalizar su funcionamiento:

- Puedes ampliar el campo con `size`.
- Puedes limitar el número de caracteres que se pueden escribir con `maxlength`.

- Puedes rellenar previamente el campo con un valor por defecto.
- Puedes dar una indicación de los contenidos del campo con `placeholder`. Este indicador desaparecerá cuando el visitante ha hecho clic dentro del campo.

En el siguiente ejemplo, el cuadro de texto contiene una indicación para entender lo que se pide en la entrada, el campo es de 30 caracteres, pero sólo se pueden escribir hasta 10 caracteres en el interior:

Código HTML

```
<form method="post" action="tratamiento.php">
<p>
<label for="nombre">Vuestro nombre:</label>
<input type="text" name="nombre" id="nombre" placeholder="Ej:
Barzanallana" size="30" maxlength="10" />
</p>
</form>
```

Contraseña

Puedes asegurarte que el cuadro de texto actúa como una "zona de contraseña", es decir, una zona donde no se pueden ver en la pantalla los caracteres ingresados. Para crear este tipo de campo de entrada, utiliza el atributo `type` = "`password`".

Cómo completo mi formulario. Ahora se pide nombre y contraseña:

Código: HTML

```
<form method="post" action="tratamiento.php">
<p>
        <label for="nombre">Vuestro nombre :</label>
        <input type="text" name="nombre" id="nombre" />
<br />
        <label for="pass">Clave :</label>
        <input type="password" name="pass" id="pass" />
</p>
</form>
```

Cuadro de texto de líneas múltiples

Para crear un cuadro de texto multilínea, cambia la etiqueta: usaremos <textarea> </ textarea>.

Al igual que cualquier otro elemento con formularios, hay que darle un nombre con `name` y

usar un `label` que explica lo que es.

Código : HTML

```
<form method="post" action="tratamiento.php">
<p>
<label for="mejorar">¿Cómo piensas mejorar tu sitio?</label><br />
<textarea name="mejorar" id="mejorar"></textarea>
</p>
</form>
```

Por defecto la zona para escribir es un poco pequeña. Afortunadamente, puedes cambiar el tamaño con <textarea> de dos maneras diferente.

- CSS: solo se aplican las propiedades CSS `width` y `height` para <textarea>.
- Con atributos: se pueden agregar los atributos `rows` y `cols` a la etiqueta <textarea>. El primer número indica el número de líneas de texto que se pueden visualizar simultáneamente, y el segundo el número de columnas.

Puedes mostrar <textarea> con un valor predeterminado. En este caso, no utilices el atributo `value`: escribimos sólo el texto predeterminado entre la etiqueta de apertura y la etiqueta de cierre.

Código: HTML

```
<form method="post" action="tratamiento.php">
<p>
<label for="mejorar">
¿Cómo piensas mejorar tu sitio?
</label>
<br />
<textarea name="mejorar" id="mejorar" rows="10" cols="50">
¿Mejorar tu sitio? Parece que no es necesario
</p>
</form>
</textarea>
```

Áreas de entrada enriquecida

HTML5 añade muchas nuevas características relacionadas con los formularios. Nuevos tipos de campos han aparecido con esta versión. Es suficiente dar el atributo `type` de la etiqueta <input/> uno de los nuevos valores disponibles. Vamos a hacer un resumen rápido.

Todos los navegadores todavía no conocen las áreas de entrada enriquecida. En su lugar, las versiones antiguas de los navegadores mostrarán un cuadro sencillo de una sola línea de texto (como si hubiera escrito type = "text"). Los nuevos navegadores pueden disfrutar de las últimas funciones, mientras que los primeros parecen un reemplazo adecuado de cuadro de texto.

- E-mail

Puedes pedir que se introduzca un e-mail:

Código : HTML

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

El campo parece idéntico a *priori* pero el explorador sabe ahora que el usuario debe ingresar una dirección de correo electrónico. Puede mostrar una indicación si la dirección introducida no es un e-mail.

- Dirección URL

Con `type URL`, puedes pedir que se introduzca una dirección absoluta (que suele empezar con http://):

Código: HTML

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

Se supone que los visitantes introducen una dirección. En los navegadores móviles se muestra como un teclado adaptado.

- Número de teléfono

Este campo está dedicado a la captura de números de teléfono:

Código: HTML

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

- Cualquier número entero

Este campo permite introducir un número entero:

Código: HTML

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

El campo se mostrará en general con pequeñas flechas para cambiar el valor. Puedes personalizar el funcionamiento del campo con los siguientes atributos:

`min`: valor mínimo permitido.
`max`: valor máximo permitido.
`step`: es el "paso" del desplazamiento. Si se especifica un paso 2, el campo sólo aceptará valores de 2 en 2 (por por ejemplo, 0, 2, 4, 6 ...).

- Cursor

El tipo `range` permite seleccionar un número con un control deslizante (*slider*), como se muestra en la figura siguiente:

Código: HTML

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

Puedes usar de nuevo los atributos, `max`, `min` y `step` para restringir los valores disponibles.

- Color

Este campo permite introducir un color:

Código: HTML

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

En la práctica, está bastante soportado por los navegadores actuales. No te sorprendas si sólo ves un campo de texto clásico.

- Fecha

Los diferentes tipos de campos de selección de fecha que existen:

`date`: fecha (por ejemplo, 08/05/1995);
`time`: hora (por ejemplo, 13:37);
`week`: para la semana;
`month`: para el mes;

`datetime`: para fecha y hora (considerando el desplazamiento horario);
`datetime-local`: para fecha y hora (sin manejo del desplazamiento horario).

Ejemplo:

Código: HTML

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

Como puedes ver, hay una opción, pero en la actualidad, pocos navegadores soportan este tipo de campo, excepto Opera.

- Búsqueda

Se puede crear un cuadro de búsqueda de la siguiente manera:

Código: HTML

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

El navegador decide cómo mostrar el campo de búsqueda. Así, se puede añadir una pequeña lupa al campo para indicar que se trata de un campo de búsqueda y eventualmente permite el almacenamiento de las últimas búsquedas por parte del visitante.

- Elementos de opciones

HTML ofrece una serie de opciones para utilizar en el formulario. Estos son los elementos que requieren los visitantes para elegir de una lista de posibilidades. Vamos a revisarlas:

casillas de verificación;
áreas de opciones;
listas desplegables.

Casillas de verificación

¿Crear una casilla de verificación? Nada más fácil. Vamos a volver a utilizar la etiqueta <input/>, esta vez especificando el tipo `checkbox` :

Código: HTML

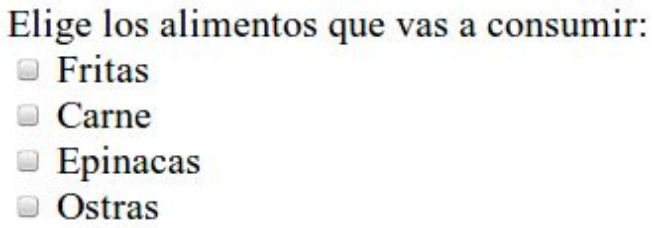
```
<input type="checkbox" name="eleccion" />
```

Agregar un <label> bien ubicado, y eso es todo.

Código: HTML

```
<form method="post" action="tratamiento.php">
<p>
Elige los alimentos que vas a consumir:<br />
<input type="checkbox" name="fritas" id="fritas" /> <label
for="fritas">Fritas</label><br />
<input type="checkbox" name="carne" id="carne" /> <label
for="carne">Carne</label><br />
<input type="checkbox" name="espinacas" id="espinacas" />
<label for="espinacas">Epinacas</label><br />
<input type="checkbox" name="ostras" id="ostras" /> <label
for="ostras">Ostras</label>
</p>
</form>
```

Recuerda que debes dar un nombre diferente a cada casilla, esto permitirá identificar las que marque el visitante.



Finalmente, recuerda que puedes hacer que una casilla esté activada por defecto con el atributo checked:

Código HTML

```
<input type="checkbox" name="elección" checked />
```

Normalmente, cada atributo tiene un valor. En este caso, sin embargo, agregar un valor no es necesario: la presencia del atributo es suficiente para hacer comprender al ordenador que la casilla debe ser revisada.

Áreas de opciones

Las áreas de opciones permiten seleccionar una (y sólo una) de entre una lista de posibilidades. Se asemejan algo a la casillas de verificación, pero hay un pequeño

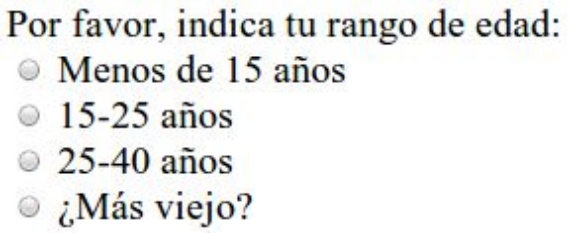
problema adicional: que se organizan en grupos. Las opciones del mismo grupo con el mismo nombre (`name`), pero cada opción debe tener un valor (`value`) diferente.

La etiqueta a usar es siempre un <input />, esta vez con el valor radio para atributo `type`.

Las cosas se verán más claras en el siguiente ejemplo:

Código: HTML

```
<form method="post" action="tratamiento.php">
<p>
Por favor, indica tu rango de edad:<br>
<input type="radio" name="edad" value="menos15" id="menos15"/>
<label for="menos15">Menos de 15 años</label><br />
<input type="radio" name="edad" value="medio15-25"
id="medio15-25" /> <label for="medio15-25">15-25 años</label><br />
<input type="radio" name="edad" value="medio25-40"
id="medio25-40" /> <label for="medio25-40">25-40 años</label><br />
<input type="radio" name="edad" value="mas40" id="mas40" />
<label for="plus40">¿Más viejo?</label>
</p>
</form>
```



Si dispones de dos áreas de opciones diferentes, debes dar un nombre único a cada grupo, de la siguiente manera:

Código: HTML

```
<form method="post" action="tratamiento.php">
<p>
Por favor, indique su rango de edad:<br />
<input type="radio" name="edad" value="menos15" id="menos15"
/> <label for="menos15">Menos de 15 años</label><br />
```

```
<input type="radio" name="edad" value="medio15-25"
id="medio15-25" /> <label for="medio15-25">15-25 años</label><br />
<input type="radio" name="edad" value="medio25-40"
id="medio25-40" /> <label for="medio25-40">25-40 años</label><br />
<input type="radio" name="edad" value="mas40" id="mas40" />
<label for="mas40">¿Mayor?</label>
</p>
<p>
¿En qué continente vive?<br />
<input type="radio" name="continente" value="europa"
id="europe" /> <label for="europa">Europa</label><br />
<input type="radio" name="continente" value="áfrica"
id="africa" /> <label for="africa">África</label><br />
<input type="radio" name="continente" value="asia" id="asia"
/> <label for="asia">Asia</label><br />
<input type="radio" name="continente" value="america"
id="america" /> <label for="america">América</label><br />
<input type="radio" name="continente" value="australia"
id="australia" /> <label for="australia">Australia</label>
</p>
</form>
```

Listas desplegables

Las listas desplegables son otra manera elegante de hacer una elección entre varias posibilidades. El funcionamiento es un poco diferente. Vamos a utilizar la etiqueta <select> </ select> que indica el comienzo y el final de la lista. Se agrega el atributo `name` a la etiqueta para dar un nombre a la lista.

Luego, dentro de <select> </ select> pondremos varias etiquetas <option> </ option> (una por elección posible). Se añade a cada una el atributo `value` con el fin de identificar lo que el visitante ha elegido.

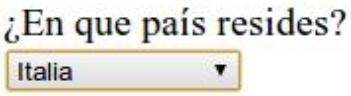
Aquí un ejemplo de uso:

Código: HTML

```
<form method="post" action="tratamiento.php">
<p>
<label for="pais">¿En que país resides?</label><br />
<select name="pais" id="pais">
<option value="francia">Francia</option>
<option value="espana">España</option>
```

```
<option value="italia">Italia</option>
<option value="reino-unido">Reino Unido</option>
<option value="canada">Canadá<Esoption>
<option value="estados-unidoss">Estados Unidos</option>
<option value="china">China</option>
<option value="japon">Japon</option>
</select>
</p>
</form>
```

El resultado se muestra seguidamente



Si quieres que una opción esté seleccionada de forma predeterminada, utiliza el atributo selected:

Código: HTML

```
<option value="canada" selected>Canadá</option>
```

También puedes agrupar las opciones con la etiqueta <optgroup> </optgroup>. En nuestro ejemplo, ¿por qué no separar los países en función de su continente?

Código HTML

```
<form method="post" action="tratamiento.php">
<p>
<label for="pais">¿En qué país resides?</label><br />
<select name="pais" id="pais">
<optgroup label="Europa">
<option value="francia">Francia</option>
<option value="espana">España</option>
<option value="italia">Italia</option>do
<option value="reino-unido">Reino-Unido</option>
</optgroup>
<optgroup label="América">
<option value="canada">Canadá</option>
<option value="estados-unidos">Estados-Unidos</option>
</optgroup>
```

```
<optgroup label="Asia">
<option value="china">China</option>
<option value="japon">Japón</option>
</optgroup>
</select>
</p>
</form>
```

Los grupos no pueden ser seleccionados. Así, en nuestro ejemplo, no se puede elegir "Europa" sólo nombres de países que estén disponibles para la selección.

Finalizar y enviar el formulario

Ya casi llegamos. Sigue siendo para nosotros la forma de decorar nuestras pocas últimas características (por ejemplo, validación), entonces podemos agregar el botón de envío del formulario.

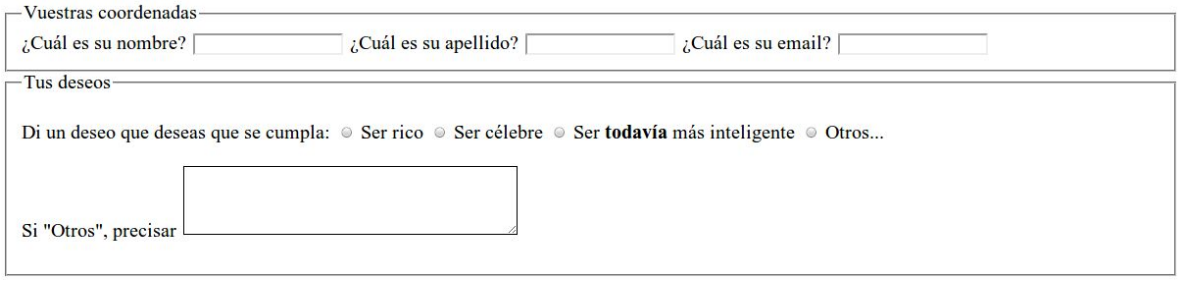
Reagrupamiento de los campos

Si la aplicación crece y tiene muchos campos, puede ser útil agruparlos en varias etiquetas <fieldset>. Cada <fieldset> puede contener una etiqueta con la leyenda <legend>. Mira este ejemplo:

Código: HTML

```
<form method="post" action="tratamiento.php">
<fieldset>
        <legend>Vuestras coordenadas</legend> <!-- Título campo -->
        <label for="nom">¿Cuál es su nombre?</label>
        <input type="text" name="nombre" id="nombre" />
        <label for="apellido">¿Cuál es su apellido?</label>
        <input type="text" name="apellido" id="apellido" />
        <label for="email">¿Cuál es su email?</label>
        <input type="email" name="email" id="email" />
</fieldset>
<fieldset>
        <legend>Tus deseos</legend> <!-- Título de campo -->
        <p>
        Di un deseo que deseas que se cumpla:
        <input type="radio" name="deseo" value="rico" id="rico" /> <label
        for="rico">Ser rico</label>
        <input type="radio" name="deseo" value="celebre" id="celebre" />
        <label for="celebre">Ser célebre</label>
```

```
<input type="radio" name="deseo" value="inteligente"
id="inteligente" /> <label for="inteligente">Ser
<strong>todavía</strong> más inteligente</label>
<input type="radio" name="deseo" value="otro" id="otro" /> <label
for="otro">Otros...</label>
</p>
<p>
:</label>
<label for="precise">Si "Otros", precisar
<textarea name="precise" id="precise" cols="40" rows="4">
</textarea>
</p>
</fieldset>
</form>
```



- Selección automática de un campo

Se puede colocar automáticamente el cursor en uno de los campos del formulario con el atributo `autofocus`, una vez el visitante carga la página, el cursor se situará en este campo.

Por ejemplo, para que el cursor se encuentre por defecto en el campo apellido:

Código: HTML

```
<input type="text" name="apellido" id="apellido" autofocus />
```

- Hacer un campo obligatorio

Se puede hacer un campo obligatorio dando el atributo necesario.

Código: HTML

```
<input type="text" name="apellido" id="apellido" required />
```

El navegador le dirá al visitante si el campo está vacío en el momento de rellenarlo, que

siempre debe ser llenado.

Los navegadores más antiguos no reconocen este atributo, envían el contenido del formulario sin verificación. Para estos navegadores, tendrás que completar las verificaciones, por ejemplo, con scripts de JavaScript.

Hay pseudoformatos en CSS para cambiar el estilo de los elementos requeridos `(:required)` e inválidos (`:invalid)`. No olvides que tienes es pseudoformato :focus para cambiar la apariencia de un campo cuando el cursor está dentro.

Código: CSS

```
:required
{
background-color: red;
}
```

Botón enviar

Falta crear el botón de envío. Una vez más, la etiqueta <input /> viene en nuestro rescate. Está disponible en cuatro versiones:

- `type` = "`submit`" para el envío del formulario principal. Este es el que vas a utilizar con más frecuencia. El visitante será dirigido a la página especificada en el atributo action del formulario.
- `type` = "`reset`" botón de reinicio del formulario.
- `type` = "`image`" equivalente al botón `submit`, en esta ocasión presenta como una imagen. Agrega el atributo `src` para introducir la dirección URL de la imagen.
- `type` = "`button`" botón genérico, que no tendrá efecto (por defecto). En general, este botón está controlado en JavaScript para realizar acciones en la página. No lo vamos a utilizar aquí.

Puedes cambiar el texto que aparece en el botón con el atributo `value`.

Para crear un botón de envío escribe, por ejemplo:

Código: HTML

```
<input type="submit" value="Enviar" />
```

Al hacer clic en el botón " Enviar", el formulario te lleva a la página especificada en el atributo de acción. Recuerda, nos imaginamos una página ficticia: tratamiento.php.

El problema es que no sólo se puede crear esta página en HTML. Es necesario aprender un nuevo lenguaje como PHP para ser capaz de "recuperar" la información introducida y decidir qué hacer.

Resumen

- Un formulario es un área interactiva de la página donde los visitantes pueden ingresar información.

- Un formulario se define con <form> al que hay que añadir dos atributos: `method` (método de envío de datos) y `action` (página a la que el visitante es redirigido después de enviar el formulario y tratar la información).

- Gran parte de los elementos del formulario pueden insertarse con <input />. El valor del atributo `type` le permite especificar el tipo de campo que debe ser insertado:

    - `text`: cuadro de texto;
    - `password`: cuadro de texto de contraseña;
    - `tel`: número de teléfono;
    - `checkbox`: casilla de verificación;
    - etc.

- La etiqueta <label> permite escribir un título. Está asociada con un campo de formulario con el atributo `for`, que debe tener el mismo valor que el `id` del campo del formulario.

- Se puede hacer un campo `required` con el atributo necesario, asegúrate que está seleccionado por defecto con autofocus, dando una indicación en el campo con `placeholder`.

- Para recuperar lo que los visitantes rellenan, el lenguaje HTML no es suficiente. Se debe utilizar un lenguaje “servidor” tal como PHP.

Vídeo y audio

Desde la llegada de Youtube, se ha convertido en algo común hoy en día ver vídeos en sitios web. La llegada de la banda ancha ha ayudado a "democratizar" los vídeos en la Web.

Sin embargo, no se permitían etiquetas HTML hasta ahora para gestionar el video. Era preciso utilizar un *plug-in* como Flash. Incluso hoy en día, Flash es, con mucho, el más utilizado para ver vídeos en Youtube, Dailymotion y Vimeo, entre otros servicios. Pero el uso de un *plug-in* tiene muchos defectos: depende de los que lo manejan (en este caso, la empresa Adobe, propietaria de Flash), no siempre se puede controlar su funcionamiento, a veces hay problemas de seguridad. En última instancia, es bastante pesado.

Esto es por lo que se crearon dos etiquetas estándar en HTML5, <video> y <audio>.

Audio y formatos de vídeo

Cuando se mostraron las imágenes y la etiqueta <img />, se empezó con una visión general de los diferentes formatos de imágenes (JPEG, PNG, GIF, etc) .. Para vídeo y audio, es lo mismo, pero es más complicado.

De hecho, la operación de los videos es tan compleja que incluso podrías hacer un curso completo sobre este tema. Dado que estamos hablando de HTML, no pasaremos a estudiar las complejidades de la codificación de vídeo. Voy a simplificar las cosas y diré todo lo que necesitas saber.

Formatos de audio

Para reproducir música o sonido, hay muchos formatos. La mayoría de ellos están comprimidos (como son JPEG, PNG y GIF), lo que reduce su "peso":

- MP3: No puedes no haberlo oído. Este es uno de los más antiguos, pero también uno de los más consistentes (todos los dispositivos pueden leer MP3), por lo que es aún ampliamente utilizado hoy en día.
- AAC: principalmente utilizado por iTunes de Apple, este es un buen formato. IPod, iPhone y iPad entre otros pueden leerlo sin problemas.
- OGG: Ogg Vorbis está muy extendido en el mundo del software libre, incluido Linux. Este formato tiene la ventaja de ser libre, es decir, que no está protegida por ninguna patente.
- WAV (sin comprimir): has de evitar su uso porque los archivos son grandes con este formato. Esto es equivalente al de mapa de bits (BMP) para el audio.

No hay ningún navegador que gestione todos estos formatos a la vez. Recuerda especialmente la compatibilidad para MP3 y OGG.

Formatos de vídeo

El almacenamiento de vídeo es mucho más complejo. Necesitamos tres cosas:

- Un formato de contenedor: es como una caja que se usa para contener los dos elementos siguientes. se reconoce generalmente por el tipo de extensión del archivo: AVI, MP4, MKV ...

- Un códec de audio: es el formato de sonido del video, por lo general comprimido. Usamos los mismos ya vistos: MP3, AAC, OGG ...

- Un códec de vídeo: este es el formato que va a comprimir las imágenes. Aquí es donde las cosas se ponen difíciles, ya que estos formatos son complejos y no siempre pueden ser gratuitos.

Los principales para usar en una Web son:

- H.264: Uno de los más poderosos y utilizados ... pero no es 100% gratis. De hecho, podemos utilizarlo de forma gratuita en algunos casos (como el *streaming* de vídeo en una página web personal), pero hay una inseguridad jurídica que hace que sea arriesgado usarlo a toda costa.

- Ogg Theora: codec libre de derechos, pero menos potente que H.264. Es bien conocido su uso en Linux pero en Windows, es necesario instalar programas para usarlo.

- WebM: otro codec gratuito y libre de derechos. Propuesto por Google es el competidor más cercano H.264.

Inserción de un archivo de audio

La etiqueta <audio> es reconocido por todos los navegadores, incluyendo Internet Explorer desde la versión 9 (IE9). En teoría, sólo hay una única etiqueta para reproducir un sonido en nuestra página:

Código: HTML

```
<audio src="musica.mp3"></audio>
```

En la práctica, esto es un poco más complicado. Si pruebas este código, no verás nada. En

efecto, el navegador sólo descargará la información general sobre el archivo (llamados metadatos), pero no pasa nada en particular.

Se puede completar la etiqueta con los siguientes atributos:

- `controls`: para agregar los botones "*Play*", "Pausa" y la barra de desplazamiento. Esto puede parecer indispensable, y es posible que se pregunte por qué no está allí por defecto, pero algunos sitios web prefieren crear sus propios botones y controlar la reproducción con JavaScript.
- `width`: para cambiar el ancho de la herramienta para la reproducción de audio.
- `loop`: la música se reproducirá en bucle.
- `autoplay`: la música se reproducirá cuando se carga la página. No abuses, suele ser irritante para llegar a un sitio que toca música.
- `preload`: indica si la música se puede precargar al cargar la página o no. Este atributo puede tomar valores:
  - `auto` (predeterminado): el navegador decide si precargar toda la música, sólo los metadatos o nada.
  - `metadata`: carga únicamente metadatos (duración, etc) ..
  - `none`: no hay carga previa. Útil si no quieres desperdiciar ancho de banda en tu sitio. No se puede forzar la precarga de música siempre decide el navegador. Los navegadores móviles, por ejemplo, nunca precargan para ahorrar ancho de banda.

Para añadir los controles esto es lo mejor:

Código: HTML

```
<audio src=”musica.mp3" controls></audio>
```

¿Por qué abrir la etiqueta para cerrarla inmediatamente?

Esto permite mostrar un mensaje o proporcionar una reserva para los navegadores que no soporten este nueva etiqueta. Por ejemplo:

Código: HTML

```
<audio src="musica.mp3" controls>Por favor, actualice su navegador
</audio>
```

Los que tienen un navegador reciente no verán el mensaje. Los navegadores más antiguos no entienden la etiqueta, sin embargo, mostrarán el texto que está dentro.

Se pueden proponer varias versiones del archivo de audio, para que el navegador detecte el

adecuado. En este caso, vamos a construir nuestra etiqueta como esta:

Código: HTML

```
<audio controls>
<source src="musica1.mp3"></source>
<source src="musica1.ogg"></source>
</audio>
```

Inserción de vídeo

La etiqueta <video> es reconocida por todos los navegadores, incluyendo Internet Explorer desde la versión 9 (IE9). Es suficiente la etiqueta <video> simple para insertar un vídeo en la página:

Código: HTML

```
<video src="sintel.webm" ></video>
```

Pero de nuevo, puedes estar decepcionado si sólo utilizas este código. No hay control para iniciar el video.

Vamos a añadir algunos atributos (en su mayoría los mismos que en la etiqueta <audio>)

- `poster`: imagen en lugar del vídeo, ya que no se pone en marcha. Por defecto, el navegador tiene el primer fotograma del vídeo, pero como es a menudo un cuadro negro, no es representativo del vídeo. Te aconsejo crear uno. Se puede tomar una instantánea de un momento en el video.
- `controls`: para agregar los botones "Play", "Pausa" y la barra de desplazamiento. Esto puede parecer indispensable, pero algunos sitios prefieren crear sus propios botones y controlar la reproducción con JavaScript. Para nosotros, será suficiente.
- `width`: para cambiar el ancho del vídeo.
- `height`: para cambiar la altura del vídeo.
- `loop`: bucle del vídeo.
- `autoplay`: el vídeo se reproducirá cuando se carga la página. Una vez más, no abusar, suele ser irritante llegar a un sitio que muestra el vídeo solo.

`preload`: especifica si el vídeo se puede precargar al cargar la página o no. Este atributo puede tomar los siguientes valores:

- `auto` (predeterminado): el navegador decide si precargar el vídeo entero, sólo los metadatos o nada en absoluto.
- `metadata`: carga sólo los metadatos (tiempo, tamaño, etc) ..
- `none`: no hay precarga previa. Es útil si desea evitar el desperdicio de ancho

de banda en tu sitio.

He aquí el código de ejemplo completo.

Código HTML

```
<video src="mivideo.webm" controls poster="mivideo-img.jpg"
width="600"></video>
```

La respuesta es la misma que para la etiqueta<audio>. Esto permite mostrar un mensaje o utilizar una técnica de emergencia (Flash) si el navegador no reconoce la etiqueta:

Código: HTML

```
<video src="mivideo.webm" controls poster="mivideo.jpg" width="600">
Es tiempo de actualizar el navegador
</video>
```

¿Cómo satisfacer a todos los navegadores, ya que no todo el mundo reconoce distintos formatos de video?

Se utiliza la etiqueta <source> dentro de la etiqueta <video> ofrecer diferentes formatos. El navegador lo reconocerá:

Código: HTML

```
<video controls poster="mivideo.jpg" width="600">
<source src="mivideo.mp4" />
<source src="mivideo.webm" />
<source src="mivideo.ogv" />
</video>
```

¿Cómo visualizar el vídeo en pantalla completa?

No es posible en la actualidad. De hecho, en efecto, hay una manera en Firefox pero está un poco oculta: es necesario hacer clic derecho en el video y seleccionar "Pantalla completa".

No hay manera de forzar a modo de pantalla completa, incluso en JavaScript. Esto es comprensible, ya que los sitios podrían interrumpir la navegación del visitante para mostrar el vídeo en pantalla completa sin pedir consentimiento.

¿Cómo proteger mi vídeo, yo no quiero que se pueda copiar fácilmente?

No es posible. Las etiquetas no fueron diseñadas para limitar o impedir la descarga. Esto es bastante lógico cuando piensas: para que el visitante pueda ver el vídeo, debe haber una forma de descarga de una manera u otra. No trates de evitar la descarga de vídeo con esta técnica.

Los reproductores de vídeo flash permiten "proteger" el contenido de los vídeos, pero una vez más, las soluciones para saltarlas existen. Muchos *plug-ins* permiten descargar vídeos de Youtube, por ejemplo.

Resumen

- Insertar música o vídeo no era posible en HTML. Era necesario el uso de un *plug-in* como Flash.

- Las etiquetas HTML5 <video> y <audio> se han introducido y utilizado para reproducir música y vídeos sin necesidad de complementos.

- Hay varios formatos de vídeo y audio. Es necesario conocer:
    - para audio: MP3 y Ogg Vorbis;
    - para vídeo: H.264, Ogg Theora y WebM.

- El formato no es reconocido por todos los navegadores: debes ofrecer diferentes versiones de música o vídeo para satisfacer todos los navegadores.

- Hay que añadir el atributo controls a las etiquetas <audio> y <video> para permitir a los visitantes iniciar o detener los medios de comunicación.

- Estas etiquetas no están diseñados para prevenir la descarga de música y vídeo. No puedes proteger contra copia tus medios de comunicación.

Diseño web adaptable con *Media Queries*

¿Sabes cuál es la principal preocupación de los *webmasters* que implementan el diseño de un sitio? La resolución de pantalla de los visitantes. En las pantallas, hay más o menos espacio, más o menos píxeles de ancho.

Esta información es importante cuando se crea el diseño de un sitio web en el caso que se visualice en diferentes resoluciones de pantalla. Si tienes una pantalla ancha, puedes olvidar que algunas personas navegan con pantallas más pequeñas. Y ya ni siquiera hablo navegadores para teléfonos inteligentes, que son mucho menos extensos.

Aquí es donde los *media queries* entran en juego, estas son reglas para cambiar el diseño de un sitio basado en características de la pantalla. Usando esta técnica, se puede crear un diseño que se adapta automáticamente a la pantalla de cada visitante.

Diseño web adaptable son una serie de técnicas que **permiten a nuestra página web adaptarse al medio a través del cual un usuario está accediendo a la misma**. Los tamaños de pantalla cambian según el medio con el que se accede (no es lo mismo una pantalla de un teléfono Nexus que la de un monitor panorámico de sobremesa) pero el usuario cada vez más exige que su experiencia usando nuestra web sea la óptima en cada caso concreto. Utilizando HTML y principalmente CSS podemos servir al usuario una versión de nuestra web en función del ancho de pantalla utilizado. Es decir, **nuestra web se adapta al ancho de pantalla**, responde ante los cambios del tamaño la misma. Además podemos pensar en mejorar el acceso a formularios, botones, etc… Eso es *Responsive Web Design* o Diseño Web Adaptable.

Como decíamos **ningún diseño escala de manera adecuada cuando cambia el contexto para el que fue pensado**. Los ***Media Queries*** forman parte de **CSS3** e inspeccionan las características físicas del medio que va a mostrar nuestro diseño (no olvidemos que *"query"* equivale a *"pregunta"*, es como preguntar: *¿qué medio se está usando?*). Si las características del medio utilizado por el usuario están dentro de un condicional establecido con los *Media Queries*, se aplicarán una serie de instrucciones CSS contenidas dentro del mismo, de esta manera cuando nuestro diseño fluido cambia de tamaño podremos aplicar una serie de instrucciones CSS **pensadas en exclusiva para ese nuevo tamaño**.

Ejecución de *media queries*

Las consultas de media queries son parte del nuevo CSS3. No es nuevo, pero son reglas que se pueden aplicar en ciertas circunstancias. Específicamente, serás capaz de decir: "Si la resolución de pantalla del visitante es menor que tanto, entonces aplica las siguientes propiedades CSS". Esto permite cambiar la apariencia del sitio, bajo ciertas condiciones: puedes aumentar el tamaño del texto, cambiar el color de fondo, la posición del menú, diferentes resoluciones, etc.

Contrariamente a lo que pudiera pensarse, las consultas de *media queries* no son sólo acerca de las resoluciones de pantalla. Puedes cambiar la apariencia del sitio en función de otros criterios como el tipo de pantalla (*smartphone*, TV, proyector ...), el número de colores, la orientación de la pantalla (vertical u horizontal), etc. Las posibilidades son infinitas.

Las *media queries* funcionan en todos los navegadores, incluido Internet Explorer desde la versión 9 (IE9).

Aplicar una *media querie*

Son reglas que indican cuándo debes aplicar las propiedades CSS. Hay dos maneras de utilizarlas:

- cargando una hoja de estilo .css diferente, basada en una regla (por ejemplo, "si la resolución es inferior a 1280px cargar el .css de propiedades grande");
- regla por escrito directamente al archivo .css usual (por ejemplo, "si la resolución es inferior a 1280px de ancho, cargar el .css de propiedades pequeña").

- Carga de una hoja de estilo diferente

¿Recuerdas que la etiqueta <link /> permite al código HTML cargar un archivo .css?

Código: HTML

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

Se puede añadir una ***media querie***, en la que vamos a escribir la regla que se aplica al archivo que deseas cargar. Un ejemplo:

Código: HTML

```
<link rel="stylesheet" media="screen and (max-width: 1280px)"
href="resolucion-peq.css" />
```

Al final, el código HTML podría proponer varios archivos CSS: uno por defecto (que es responsable en todos los casos) y uno o dos más que se aplicarán sólo si se aplica la regla correspondiente.

Código HTML:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<link rel="stylesheet" href="estilo.css" /> <!-- Para todo el mundo -->
<link rel="stylesheet" media="screen and (max-width: 1280px)"
href="resolucion-peq.css" />
<!-- Para los que tienen resolución menor a 1280px -->
<title>Consulta multimedia</title>
</head>
```

- Carga de reglas directamente en la hoja de estilo

Otra técnica, que personalmente prefiero, por razones prácticas, es escribir estas reglas en el mismo archivo CSS habitual. En este caso, escribimos la regla en el archivo .css de esta manera:

Código: CSS

```
@media screen and (max-width: 1280px)
{
/* Tus propiedades CSS aquí*/
}
```

Normas disponibles

Hay muchas reglas para su construcción. Presento aquí las principales:

`color`: la gestión del color (en bits/píxel).
`height`: la altura de la zona de visualización (ventana).
`width`: ancho del área de visualización (ventana).
`device-height`: altura del dispositivo.
`device-width`: anchura del dispositivo.
`orientation`: orientación (vertical u horizontal).
`media`: el tipo de salida de la pantalla. Algunos de los valores posibles:
- `screen`: Pantalla "clásica";
- `handheld`: dispositivo móvil;
- `print`: impresión;
- `tv`: televisión;
- `prolection`: proyector;
- `all`: todos los tipos de pantallas

Podemos añadir el prefijo `min` o `max` antes que la mayoría de estas reglas. Así, `min-width` significa "ancho mínimo", `max-height` "altura máxima", etc.

La diferencia entre `width` y `device-width` se percibe especialmente en los navegadores móviles para teléfonos inteligentes, que se considerará más adelante.

Las reglas se pueden combinar con las siguientes palabras:

`only`: "sólo";
`and`: "y";
`not`: "no".

Algunos ejemplos de multimedia que te ayudarán a comprender bien el principio.

Código: CSS

```
/ * En las pantallas, cuando la anchura de la ventana es de un máximo de
1280px * /

@media screen and (max-width: 1280px)
/* En todos los tipos de pantallas, cuando la anchura de la ventana es
entre 1024px y 1280px */
@media all and (min-width: 1024px) and (max-width: 1280px)
/* Sobre los televisores*/
@media tv
/* Sobre todas las pantallas con orientación vertical */
@media all and (orientation: portrait)
```

Prueba de *media querie*

media querie se utiliza sobre todo para adaptar el diseño del sitio para anchos de pantalla diferentes.

Hagamos una prueba sencilla: vamos a cambiar el color y el tamaño de la ventana si el texto es más o menos de 1024 píxeles de ancho. Para esta prueba, voy a utilizar el segundo método que es escribir la regla directamente en el mismo archivo .css usual:

Código: CSS

```
/* Párrafos en azul por defecto */
p
{
color: blue;
}
/* Nuevas reglas si el texto es de más de 1024px de ancho */
@media screen and (max-width: 1024px)
{
p
{
color: red;
background-color: black;
font-size: 1.2em;
}
}
```

En nuestro CSS, lo primero que pedimos es que el texto del párrafo está escrito en azul, hasta ahora nada nuevo. Sin embargo, hemos agregado una ***media querie*** que se aplica a todas las

pantallas cuya anchura no exceda de 1024px. En el interior, se aplican las reglas CSS en los párrafos para escribir el texto más grande y de color rojo. Resultado: la página no tiene la misma apariencia según el tamaño de la ventana.

Puesta en práctica de *media querie*

Cambiar el color del texto es agradable, pero no aporta mucho. Por contra, se vuelve inmediatamente más interesante si se utilizan ***media querie*** para cambiar la apariencia del sitio en función de la resolución. Verás que puedes hacer lo que quieras.

Por ejemplo: el sitio es adecuado para la mayoría de las resoluciones de pantalla, pero cuando la pantalla es menor que 1024px, se hace necesario un *scroll* a la derecha para ver toda la página. El sitio no es muy conveniente para ver en una pantalla pequeña.

Sugiero que uses ***media querie*** para cambiar la apariencia del sitio en las resoluciones por debajo de 1024 px de ancho.

Vamos a hacer los siguientes cambios:

- el menú de navegación en la parte superior derecha se organizarán en altura en lugar de ancho, y los enlaces se escriben más pequeños;
- la banderola se eliminará, porque se necesita una gran cantidad de espacio y no proporciona mucha información;
- el bloque <aside> "Sobre el autor" se colocará en el artículo (y no a un lado), y su contenido será reorganizado (la foto se colocará flotante).

Se podría, por supuesto, haber realizado muchos otros cambios: cambiar el color, arreglo de pie de página, etc. Pero esto será lo suficiente como para entrenarse. Vamos a trabajar directamente en el archivo estilo.css que hicimos previamente. Vamos a añadir algunas instrucciones multimedia para adaptar el diseño.

Página

Por el momento, el ancho de página se establece en 900 píxeles y el contenido se centra:

Código: CSS

```
#bloc_page
{
width: 900px;
margin: auto;
}
```

Siguiendo estas líneas, sugiero añadir la siguiente *media querie:*

Código: CSS

```
@media all and (max-width: 1024px)
{
#bloc_page
{
width: auto;
}
}
```

La regla significa: "Para todos los tipos de pantallas, si el ancho de la ventana no supera 1024px, a continuación, ejecutar las siguientes reglas CSS. "

Las reglas CSS en cuestión son muy simples, en realidad hay sólo una: le da un ancho de página automático (en lugar un ancho fijo de 900 píxeles) a la página. Entonces tendrá todo el espacio disponible en la ventana. Esto impide las barras de desplazamiento horizontal en pequeñas resoluciones.

Auto es el valor predeterminado de la propiedad width. De forma predeterminada, los bloques tienen un ancho de forma automática (toman todo el espacio disponible). Este valor sobreescribe lo que nos obligó a 900px pocas líneas más arriba: así que volvemos al comportamiento predeterminado del bloque.

Menú de navegación

Queremos que el menú de navegación ocupe menos espacio en pequeñas resoluciones. En lugar de darle una dimensión fija, lo vamos a restaurar a su dimensión flexible original de forma automática. Cada elemento del menú está escrito por debajo de la anterior: para ello, debemos transformar las viñetas en `block` en lugar de `inline-block`.

Por último, el texto se escribirá más pequeño y se eliminarán las líneas de los enlaces inferiores cuando se sobrevuelan, ya que el texto es menos adecuado para este disposición.

Código: CSS

```
@media all and (max-width: 1024px)
{
nav
{
width: auto;
text-align: left;
```

```
}
nav li
```

Banderola

Quitar la banderola es simple: utilizar la propiedad de visualización a la que se le asigna el valor `none`. Si la ventana es demasiado pequeña, preferimos ocultar completamente la banderola:

Código: CSS

```
@media all and (max-width: 1024px)
{
#imagen-banderola
{
display: none;
```

Bloque "Sobre el autor"

En lugar de colocar el bloque a la derecha del artículo, vamos a pasar por debajo. Este tipo de arreglo *top-down* es más adecuado para pantallas pequeñas.

En el interior del bloque reajustamos ligeramente la posición de los elementos: la foto en particular, se colocará flotando a la derecha.

Código: CSS

```
@media all and (max-width: 1024px)
{
article, aside
{
width: auto;
display: block;
margin-bottom: 15px;
}
#flecha_burbuja
{
display: none;
}
#photo_zozor img
{
width: 110px;
```

```
float: right;
margin-left: 15px;
aside p:last-child
}
{
text-align: center;
}
```

¿Qué significa, `aside p: last-child` ?

Es un selector avanzado que no hemos utilizado hasta ahora. aside p hace a un lado "Todos los párrafos en la etiqueta <aside> ". Con: `last-child`, apuntamos sólo al último párrafo en el bloque aside (que contiene el enlaces a Facebook y Twitter), con el fin de centrar las imágenes. Por supuesto, también se podría asignar una `class` o `id` a este párrafo para apuntar directamente, pero yo no quiero editar el código HTML.

Resultado

La página está ahora completamente reorganizada cuando la ventana es de 1024px de ancho o menos. Mira los resultados, la siguiente figura habla por sí misma.

Multimedia y navegadores móviles

Como probablemente sabes, las pantallas de los teléfonos inteligentes son mucho más estrechas que las pantallas tradicionales (sólo unos cientos de píxeles de ancho). Para

adaptarse, los navegadores móviles muestran el sitio en "dézoomant", que permite tener una visión general de toda la página. El área de visualización simulada se denomina viewport : el ancho de la ventana en navegador en el móvil.

En CSS, en multimedia si apuntas a la pantalla con `max-width` en un móvil, comparará el ancho que indican con su visión. El problema es que la ventana gráfica cambia en función del navegador móvil utilizado.

Para hacer referencia a los teléfonos inteligentes, en lugar de utilizar *max-width*, puede valer la pena utilizar `max-device-width` : es la anchura del dispositivo. Los dispositivos móviles no exceden de 480 px de ancho, podemos apuntar sólo a medios móviles con navegadores:

Código: CSS

```
Code : CSS
@media all and (max-device-width: 480px)
{
/* Reglas CCS para móviles */
}
```

Resumen

- Los *media querie* se pueden cargar con diferentes estilos CSS basadas en ciertos parámetros.
- Los parámetros permitidos por los *media querie* son numerosos, número de colores, resolución de pantalla, orientación …
- En la práctica, se utiliza sobre todo para cambiar la apariencia del sitio para diferentes resoluciones de pantalla.
- Hemos creado un *media querie* con la directiva `@media` seguido por tipo de pantalla y una o más condiciones (tales como máximo ancho de la pantalla). El siguiente CSS se activará únicamente si se cumplen las condiciones.
- Los navegadores móviles simulan un ancho de pantalla: esto se llama la ventana gráfica.
- Se pueden dirigidos a los teléfonos inteligentes con una regla basada en el número real de píxeles que aparecen en la pantalla: `max-device-width`.

Ir más allá

Aunque esta documentación está llegando a su fin, es tentador pensar que hemos visto de todo. ¿Lo he visto todo? Hay cientos de cosas por descubrir, ya sea en HTML, CSS, o tecnologías que están relacionadas (PHP, JavaScript ...).

Este capítulo tiene por objetivo de dar algunas instrucciones para completar el aprendizaje. Así que no estés triste, porque no has terminado de hacer descubrimientos.

Aplicaciones web (JavaScript, AJAX ...)

JavaScript es un lenguaje que ha existido desde hace muchos años y que se utiliza frecuentemente en la Web junto con HTML y CSS. Este es probablemente uno de los primeros lenguajes que desearás aprender ahora que tienes conocimientos de HTML y CSS.

¿Para qué puede servir JavaScript? No podemos hacer todo con HTML y CSS.

Podemos hacer muchas cosas en HTML y CSS, pero cuando se quiere hacer la página más interactiva, un lenguaje como JavaScript es necesario. Algunos ejemplos en los que JavaScript se puede utilizar:

- Con mayor frecuencia se utiliza para cambiar las propiedades CSS sin tener que recargar la página. Por ejemplo, se señala una imagen y el fondo del sitio cambia de color (no es posible hacerlo con un: `hover` porque se trata de dos etiquetas diferentes, esto es de hecho una limitación de CSS).

- También se puede utilizar para editar el código fuente HTML sin tener que recargar la página mientras que el visitante visualiza la página.

- También permite mostrar cuadros de diálogo en la pantalla del visitante.

- O cambiar el tamaño de la ventana.

JavaScript es un lenguaje que está más cerca de lenguajes de programación como C, C ++, Python, Ruby ... En contraste, los lenguajes HTML y CSS son lenguajes de descripción: describen cómo debe aparecer la página pero lo hacen sin órdenes directas al ordenador ("haz esto, haz lo otro ..."), a diferencia de JavaScript. JavaScript tiene nada que ver con Java, sólo que los nombres son similares.

JavaScript se utiliza comúnmente hoy en día para hacer AJAX *(Asynchronous JavaScript And*

XML), esta técnica puede cambiar una parte de la página web que el visitante consulta, intercambio de datos con el servidor. Esto da la impresión de que las páginas son más dinámicas y más sensibles. El visitante no tiene que volver a cargar sistemáticamente toda la página.

Los navegadores son cada vez más eficaces en el tratamiento de JavaScript, por lo que las páginas que utilizan JavaScript son más reactivas. Puede suceder hoy para crear sitios que literalmente se convierten en aplicaciones web, pero el software equivalente disponibles como sitios web.

Un ejemplo famoso: Google Docs (denominado recientemente como Google Drive), la suite de oficina de Google en la Web.

Tecnologías relacionadas con HTML5

El W3C no funciona en HTML y CSS. Estos son sin duda los más conocidos, pero el W3C también busca identificar otras tecnologías que complementan HTML y CSS. Son numerosos y a menudo se confunden también con HTML5.

De hecho, HTML5 se ha convertido en un término muy utilizado que se refiere a tecnologías que no sean HTML. Cuando alguien dice "HTML5" hoy en día, también puede referirse a otros elementos que van más allá del estricto HTML.

Esta es una pequeña lista de las nuevas tecnologías introducidas en paralelo con HTML5 (ten en cuenta que algunas no son realmente nuevas:

- Canvas: permite dibujar en la página web, dentro de la etiqueta HTML <canvas>. Podemos no solo dibujar formas (triángulos, círculos ...), sino también añadir imágenes, manipularlas, aplicar filtros gráficos ... En última instancia, nos permite conseguir los juegos reales actuales y las aplicaciones de gráficos directamente en las páginas web.

- SVG permite crear gráficos vectoriales en páginas web. A diferencia de Canvas, estos dibujos pueden extenderse hasta el infinito (esto es el principio de un vector).

- Drag & Drop: permite "arrastrar y soltar" objetos en la página web, de la misma manera que puedes arrastrar y soltar los archivos en el escritorio. Gmail lo utiliza para permitir fácilmente agregar datos adjuntos a un e-mail.

- API archivos: permite acceder a los archivos almacenados en el ordenador del visitante (con permiso). Utilizamos especialmente en

combinación con Drag & Drop.

- Geolocalización: para localizar los servicios a los visitantes y ofertas relacionados con el lugar donde se encuentra (por ejemplo, horarios de cines cercanos). La ubicación no es siempre exacta, pero se puede identificar a un visitante a pocos kilómetros (con su permiso).

-Almacenamiento web: permite almacenar una gran cantidad de información en el ordenador del visitante. Esta es una alternativa poderoso a los cookies tradicionales. La información es jerárquica, como en una base de datos.

- AppCache: decirle al navegador que algunos archivos de caché, entonces se verá más

- AppCache: decirle al navegador que algunos archivos permanezcan en la caché, entonces se verá de manera más sistemática el descargar. Muy útil para crear aplicaciones web que pueden funcionar incluso en *off line* (desconectado).

- Web Sockets: permite un intercambio más rápido en tiempo real entre el navegador del usuario y el servidor que gestiona el sitio web (esto es un tipo de mejora de AJAX). En futuras aplicaciones web, pueden ser tan reactivos como los programas reales.

- WebGL: introduce 3D en páginas web usando el estándar OpenGL 3D (figura siguiente). las escenas 3D son gestionadas directamente por la tarjeta gráfica.

Sitios web dinámicos (PHP, JEE, ASP. NET ...)

Aquí vamos a discutir también los lenguajes de programación. ¿Cómo JavaScript? Sí, pero con una importante diferencia: JavaScript se ejecuta en el equipo de los visitantes, mientras que los lenguajes que veremos se ejecutan en el "servidor" que contiene el sitio web.

¿Qué diferencia hace que el programa se ejecute en la máquina del visitante o en el servidor? Las diferencias son significativas. En primer lugar, en términos de potencia, un servidor a menudo será más rápido que la máquina de los visitantes, lo que le permite realizar cálculos más complejos. También tendrás un mayor control en el lado del servidor

JavaScript ... Sin embargo, JavaScript es insustituible, porque hay algunas cosas que no se pueden hacer en el lado del "visitante".

Los lenguajes servidor pueden generar la página web cuando el visitante llega al sitio (figura siguiente). Cada visitante puede obtener una página web personalizada de acuerdo a sus necesidades.

Los lenguajes no sirven para las mismas cosas, sino que se complementan entre sí. Se combinan HTML, CSS, JavaScript, PHP.

Por ejemplo, con AJAX (intercambio de datos entre la página y el servidor), puedes realizar cálculos, almacenar información en bases de datos, hacer verdaderos sitios web dinámicos.

Los lenguajes *server-side* son numerosos. Citemos algunos:

- PHP: Uno de los más conocidos. Fácil de usar y de gran alcance, se utiliza principalmente por Facebook.

- JEE (Java): ampliamente utilizado en el mundo profesional, es una extensión de Java que permite realizar sitios web dinámicos, potentes y robustos. Al principio es un poco más complejo que PHP.

- ASP. NET (C#) muy similar al JEE, este es el lenguaje de Microsoft. Se utiliza en combinación con otras tecnologías de Microsoft (Windows Server ...). Utiliza los NET Framework, que ofrece muchas características a los desarrolladores.

- Django (Python): una extensión del lenguaje Python que permite realizar sitios web dinámicos rápidos y fáciles. Se sabe para generar interfaces de administración rápidas de usar.

- Ruby on Rails (Ruby): una extensión del lenguaje Ruby, muy similar a la de Django, que permite fácilmente sitios web dinámicos y con mayor flexibilidad.

Conocer uno de estos lenguajes es esencial si se quieren procesar los resultados de los

formularios HTML, recuerda que que la etiqueta <form> sirve para crear formularios, pero no es la forma de recuperar la información ingresada por los visitantes. Se necesita un lenguaje de servidor obligatoriamente, como PHP, para recuperar y procesar los datos.

En última instancia, estos lenguajes permiten alcanzar tus sueños en el sitio web

foros;
boletín de noticias;
contador de visitas;
sistema automatizado de noticias;
gestión de miembros;
web de juegos.
etc...