UD3.1 Introducción a HTML5 y CSS3.

DISEÑO DE INTERFACES WEB

3.1. Introducción a HTML5 y CSS3

Contenidos

1. Objetivos

En este primer tema se pretenden conseguir los siguientes objetivos:

- Conocer **qué** es y **para qué** sirve HTML5 y CSS3.
- Ver las **diferencias** entre HTML5 y sus antecesores HTML4 y XHTML.
- Repasar las **características** de HTML4 que son similares o iguales a la nueva versión **HTML5**.
- Repasar las **características** de CSS2.1 que son similares o iguales a la nueva versión **CSS3**.
- **Aplicar estilos** CSS a un documento HTML para lograr el diseño deseado.

2. Introducción

Este tema ofrece una visión general de

- Cómo hemos llegado a donde estamos hoy en día
- Por qué HTML5 y CSS3 son tan importantes para las aplicaciones web modernas
- Cómo el uso de estas tecnologías nos va a facilitar ampliamente nuestra labor como desarrolladores web.

2. Introducción

¿Qué es HTML5?

HTML es el **lenguaje de marcado** predominante para describir el contenido o los datos en la World Wide Web.

HTML5 es la **última versión** de este lenguaje, e incluye nuevas características, mejoras en las características existentes, y scripting basado en APIs.

No se trata de una reformulación de las versiones anteriores del lenguaje ya que incluye todos los elementos válidos tanto en HTML 4 para asegurarse de que funciona en la **mayoría de las plataformas existentes** en el mercado y que es **compatible con los navegadores** antiguos.

2. Introducción

¿Cómo hemos llegado hasta aquí?

La industria del diseño web se ha desarrollado en un periodo de tiempo relativamente corto.

Hoy en día, son cada vez más y más comunes aplicaciones web sencillas que **muestran datos en función de resultados** basados en peticiones Ajax y que dependen de scripting del lado del cliente para la funcionalidad critica.

En la actualidad, cada vez encontramos más Sitios Web que son **semejantes a aplicaciones de escritorio**.

2. Introducción

¿Por qué debería interesarme HTML5?

Los nuevos elementos de HTML5, tecnologías y APIs relacionadas, se han introducido con el objetivo de que las páginas web sean **más sencillas** de codificar, usar y acceder.

Estos nuevos elementos, ayudan a que nuestros documentos sean **más accesibles** para los seres humanos y las máquinas, lo cual, resulta beneficioso tanto para la accesibilidad como para el SEO.

Gracias a las API asociadas se ayuda a **mejorar las técnicas** utilizadas por los desarrolladores **simplificando el trabajo** y poniendo más poder en manos de los mismos.

2. Introducción

¿Qué es CSS3?

CSS es un lenguaje de estilo que describe cómo se realiza la **presentación de los contenidos** HTML, y CSS3 es la última versión de este lenguaje.

CSS3 **contiene casi todo** lo que se incluye en CSS 2.1 (la versión anterior de la especificación) y, además, agrega nuevas características para ayudar a los desarrolladores a resolver una serie de problemas sin necesidad de marcado no semántico, complejos scripts o imágenes adicionales.

2. Introducción

El mercado variado de los navegadores

Los cambios introducidos en HTML5 **no causan que los navegadores más antiguos se queden obsoletos** ni que den lugar a problemas de diseño o errores de página.

Podemos tomar cualquiera de nuestros proyectos en HTML4, cambiar el tipo de documento (DOCTYPE) a HTML5 y la página **seguirá funcionando** con el mismo aspecto.

Se asegura la **compatibilidad hacia atrás** con la mayoría de navegadores, incluso IE6.

3. Plantilla HTML5 básica

Esqueleto de un documento HTML5.

```
<!doctype html>
<html lang="es">
<head>
    <meta charset="utf-8">
    <title>Plantilla HTML5</title>
    <meta name="description" content="HTML5">
    <meta name="author" content=”Nombre Apellidos">
    <link rel="stylesheet" href="css/estilos.css">
</head>
<body>
    <script src="js/scripts.js"></script>
</body>
</html>
```

Veamos a continuación las áreas en las que HTML5 difiere.

3. Plantilla HTML5 básica

<DOCTYPE>

Indica al navegador el **tipo de documento** y debe ser el **primer punto** en la parte superior de documentos HTML.

Anteriormente, la declaración de tipo de documento era farragosa y difícil de recordar.

Para XHTML 1.0 Strict:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

Y para HTML4 Transitional:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```

HTML5 ha acabado con esa monstruosidad indescifrable. Ahora todo lo que indicamos es lo siguiente:

```
<!DOCTYPE HTML>
```

3. Plantilla HTML5 básica

<HTML>

El siguiente paso en cualquier documento HTML es el elemento `<html>`, que **no ha cambiado** significativamente con HTML5.

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

En nuestro ejemplo, hemos incluido el **atributo `lang`** con un valor de es, que especifica que el documento está en español.

En HTML5, incluso **podríamos prescindir** del atributo `lang` y el documento validaría y funcionaría correctamente.

3. Plantilla HTML5 básica

<HEAD>

Sección cabecera o `head`. La primera línea dentro de la cabecera es la que define la **codificación de caracteres** que se utiliza en el documento.

Anteriormente, hacíamos esto:

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
```

HTML5 reduce este meta a la mínima expresión:

```
<meta charset="utf-8">
```

La declaración de la codificación debe aparecer dentro de los **primeros 512 caracteres** del documento y **antes de** la aparición de cualquier elemento de contenido como por ejemplo `title`

3. Plantilla HTML5 básica

<HEAD>

La siguiente parte de nuestro head es la siguiente:

```
<title>Plantilla HTML5</title>
<meta name="description" content="HTML5">
<meta name="author" content=”Nombre Apellidos">
<link rel="stylesheet" href="css/estilos.css">
```

En estas líneas, HTML5 **apenas se diferencia** de sintaxis anteriores.

El título permanece igual que antes, y las etiquetas <meta> que hemos incluido no son más que ejemplos opcionales.

3. Plantilla HTML5 básica

<HEAD>

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

Sí que vemos una sutil diferencia a la hora de incluir la hoja de estilo. A primera vista, es probable que no hayas notado nada diferente, pero habitualmente, elementos de enlace incluirían un atributo **type con un valor de text/css**. Curiosamente, esto nunca fue necesario en HTML 4, incluso cuando utilizábamos doctype strict.

La sintaxis basada en HTML5 nos anima a **omitir el atributo de tipo**, ya que todos los navegadores reconocen el tipo de contenido de las hojas de estilo vinculadas sin requerir del atributo extra.

3. Plantilla HTML5 básica

El resto del documento no cambia

Al igual que se señaló con el elemento link, la **etiqueta <script>** no requiere que se declare un atributo `type`.

En XHTML, para validar una página que contiene scripts externos, su etiqueta <script> debería tener este aspecto:

```
<script src="js/scripts.js" type="text/javascript"></script>
```

Dado que **JavaScript es el único lenguaje** de programación real utilizado en la Web, todos los navegadores asumirán que lo estamos utilizando por lo que el type es **innecesario** en los documentos HTML5:

```
<script src="js/scripts.js"></script>
```

El elemento <script> debe aparecer en la **parte inferior** de nuestro documento

4. Etiquetas básicas

Cabeceras

Se definen con los elementos desde <h1> hasta <h6>.

```
<h1>Cabecera de nivel 1</h1>
<h2>Cabecera de nivel 2</h2>
<h3>Cabecera de nivel 3</h3>
```

Las cabeceras van a definir el esquema del documento, por lo tanto, debemos **utilizarlas de forma coherente**. No tiene sentido tener un <h2> si no tenemos antes un <h1>.

Párrafos

Se introducen mediante la etiqueta <p>.

```
<p>Esto es un párrafo</p>
<p>Esto es otro párrafo</p>
```

No utilizar los párrafos para introducir **saltos de línea,** para eso ya disponemos del elemento
.

4. Etiquetas básicas

Etiquetas para formatear textos

HTML Text Formatting Tags

Tag	Description
<b>	Defines bold text
	Defines emphasized text
<i>	Defines a part of text in an alternate voice or mood
<small>	Defines smaller text
	Defines important text
<sub>	Defines subscripted text
<sup>	Defines superscripted text
<ins>	Defines inserted text
<del>	Defines deleted text
<mark>	Defines marked/highlighted text

4. Etiquetas básicas

HTML "Computer Output" Tags

Tag	Description
<code>	Defines computer code text
<kbd>	Defines keyboard text
<samp>	Defines sample computer code
<var>	Defines a variable
<pre>	Defines preformatted text

HTML Citations, Quotations, and Definition Tags

Tag	Description
<abbr>	Defines an abbreviation or acronym
<address>	Defines contact information for the author/owner of a document
<bdo>	Defines the text direction
<blockquote>	Defines a section that is quoted from another source
<q>	Defines an inline (short) quotation
<cite>	Defines the title of a work
<dfn>	Defines a definition term

4. Etiquetas básicas

Enlaces

Para crear enlaces se utiliza la etiqueta `<a>` con su atributo `href` para indicar la url de destino del enlace.

```
<a href="url">Texto del enlace</a>
```

Si queremos crear un enlace que se dirija a un sitio determinado del **documento actual**, haremos lo siguiente:

1. Colocaremos un **ancla** en el lugar al que nos queremos dirigir:

```
<a id="contacto">Contacto</a>
```

2. En el enlace pondremos como href el símbolo **# seguido del id** que le hemos puesto al ancla:

```
<a href="#contacto">Ir a contacto</a>
```

4. Etiquetas básicas

Imágenes

Para vincular imágenes a nuestro documento, utilizaremos la **etiqueta `<img>`** con su atributo **src** para indicar el origen de la imagen.

<img> es un **elemento vacío**, por lo tanto, no es necesario que introduzcamos la etiqueta de cierre.

Debemos de poner el **atributo alt** a todas las imágenes para indicar un texto alternativo.

```
<img src="url de la imagen" alt="Texto alternativo">
```

4. Etiquetas básicas

Tablas

Se definen con la etiqueta `<table>` y está dividida en **filas** con la etiqueta `<tr>`. Una fila se divide en **celdas** de datos con la etiqueta `<td>`, aunque también se puede dividir en celdas de **cabecera** con la etiqueta `<th>`.

Los elementos <td> son los **contenedores de datos** en la tabla y pueden contener todo tipo de elementos HTML, como texto, imágenes, listas, otras tablas, etc.

Firstname	Lastname	Points
Jill	Smith	50
Eve	Jackson	94
John	Doe	80
Adam	Johnson	67

4. Etiquetas básicas

Tablas

```
<table>
<tr><th>Firstname</th><th>Lastname</th><th>Points</th></tr>
<tr><td>Jill</td><td>Smith</td><td>50</td></tr>
<tr><td>Eve</td><td>Jackson</td><td>94</td></tr>
<tr><td>John</td><td>Doe</td><td>80</td></tr>
<tr><td>Adam</td><td>Johnson</td><td>67</td></tr>
</table>
```

Evidentemente, con esto sólo conseguimos la estructura de la tabla. Si queremos que nuestra tabla tenga el mismo aspecto que aparece en la imagen, tenemos que aplicar los **estilos CSS** correspondientes.

4. Etiquetas básicas

Listas (Desordenadas y Ordenadas)

Listas Desordenadas: Sus elementos **no están numerados**.

Comienzan con la etiqueta `<ul>` y cada elemento de la lista con la etiqueta `<li>`.

Los elementos de la lista están marcados con viñetas, normalmente, **pequeños círculos negros**.

```
<ul>
<li>Coffee</li>
<li>Milk</li>
</ul>
```

El código anterior se vería así:

- Coffee
- Milk

4. Etiquetas básicas

Listas (Desordenadas y Ordenadas)

Listas Ordenadas: Sus elementos **están numerados**.

La única diferencia es que comienzan con la etiqueta `<ol>` en lugar de la etiqueta `<ul>`.

Los elementos de la lista están marcados **números**.

```
<ol>
<li>Coffee</li>
<li>Milk</li>
</ol>
```

El código anterior se vería así:

1. Coffee
2. Milk

4. Etiquetas básicas

Elementos contenedores

En HTML disponemos de **elementos en bloque** y **elementos en línea**.

Un elemento en bloque es aquel que incorpora un **salto de línea** antes y después del mismo, mientras que un elemento en línea se colocará **a continuación del elemento anterior**, sin introducir ningún salto de línea.

En base a esto, tenemos dos tipos de contenedores:

- elemento en bloque
- elemento en línea.

4. Etiquetas básicas

Formularios

Los formularios HTML se utilizan para **pasar datos a un servidor**. Un formulario HTML puede contener elementos de entrada como campos de texto, casillas de verificación, radio botones, etc.

Para crear un formulario HTML utilizamos la etiqueta `<form>` y dentro del `<form>` introduciremos los campos que nos interesen: `<input>`, `<select>`, etc.

Para ver ejemplos de código que utilice formularios se puede consultar la web de w3schools

http://www.w3schools.com/html/html_forms.asp

4. Etiquetas básicas

Tag	Description
<form>	Defines an HTML form for user input
<input>	Defines an input control
<textarea>	Defines a multiline input control (text area)
<label>	Defines a label for an <input> element
<fieldset>	Groups related elements in a form
<legend>	Defines a caption for a <fieldset> element
<select>	Defines a drop-down list
<optgroup>	Defines a group of related options in a drop-down list
<option>	Defines an option in a drop-down list

4. Etiquetas básicas

Una página web dentro de otra

Para introducir una página web dentro de otra, utilizamos el elemento <iframe>:

```
<iframe src="URL"></iframe>
```

Podemos indicar el **alto y ancho** del <iframe> mediante los atributos height y width, normalmente expresados en píxeles, aunque también podemos utilizar porcentajes.

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

Un <iframe> puede ser utilizado como **destino de un enlace**, para ello, sólo tenemos que indicar el nombre del <iframe> en el atributo target del enlace.

```
<iframe src="demo_iframe.htm" name="iframe_a"></iframe>
<p><a href="http://www.w3schools.com" target="iframe_a">W3Schools.com</a></p>
```

5. Definiendo la estructura del cuerpo

La estructura del cuerpo (el código entre las **etiquetas** **<body>**) generará la parte visible del documento. Este es el código que producirá nuestra pagina web

HTML siempre ofreció diferentes formas de construir y organizar la información dentro del cuerpo de un documento. **Uno de los primeros elementos** provistos para este propósito fue `<table>`.

Más adelante, gradualmente, otros elementos reemplazaron su función, permitiendo lograr lo mismo con menos código, facilitando de este modo la creación, permitiendo portabilidad y ayudando al mantenimiento de los sitios web. El elemento `<div>` comenzó a dominar la escena.

5. Definiendo la estructura del cuerpo

Con la aparición de webs más interactivas y la integración de HTML, CSS y JavaScript, el uso de se volvió una practica común. Pero este elemento, así como `<table>`, **no provee demasiada información** acerca de las parte del cuerpo que está representando.

En el tema siguiente, veremos que HTML5 incorpora nuevos elementos que ayudan a **identificar cada sección del documento y organizar el cuerpo del mismo**, pero de momento vamos a ver cómo organizar el diseño de nuestra página utilizando elementos `<div>`.

5. Definiendo la estructura del cuerpo

Ejemplo diseño web clásico

5. Definiendo la estructura del cuerpo

1. Escribe el código HTML haciendo uso de la etiqueta que utilizarías para crear una página web con la disposición anterior.

6. CSS y HTML

Ya hemos visto cómo **organizar la estructura** del documento mediante html. Ahora es momento de analizar CSS, su relevancia dentro de esta unión estratégica y su influencia sobre la presentación de documentos HTML.

- CSS no tiene nada que ver con HTML
- Creado para superar las limitaciones y reducir la complejidad de HTML
- CSS fue adoptado como la forma de separar la estructura de la presentación.

6. CSS y HTML

Incorporar estilos al documento

Estilos en línea: Una de las técnicas más simples para incorporar estilos CSS a un documento HTML es la de asignar los estilos **dentro de las etiquetas** por medio del atributo style.

```
<p style="font-size: 20px">Mi texto</p>
```

Este método es una buena manera de probar estilos y obtener una **vista rápida** de sus efectos, pero **no es recomendable** para aplicar estilos a todo el documento.

6. CSS y HTML

Incorporar estilos al documento

Estilos embebidos: Una mejor alternativa es **insertar los estilos en la cabecera** del documento y luego usar **referencias** para afectar los elementos HTML correspondientes.

```
<style>
   p { font-size: 20px }
</style>
```

El elemento <style> permite a los desarrolladores **agrupar estilos** CSS dentro del documento. Este método sería bueno si sólo tuviéramos un documento en nuestra página.

6. CSS y HTML

Incorporar estilos al documento

Archivos externos: La solución es **mover todos los estilos a un archivo externo**, y luego utilizar el elemento <link> para insertar este archivo dentro de cada documento que los necesite. Este método nos permite cambiar los estilos por completo, simplemente, incluyendo un archivo diferente.

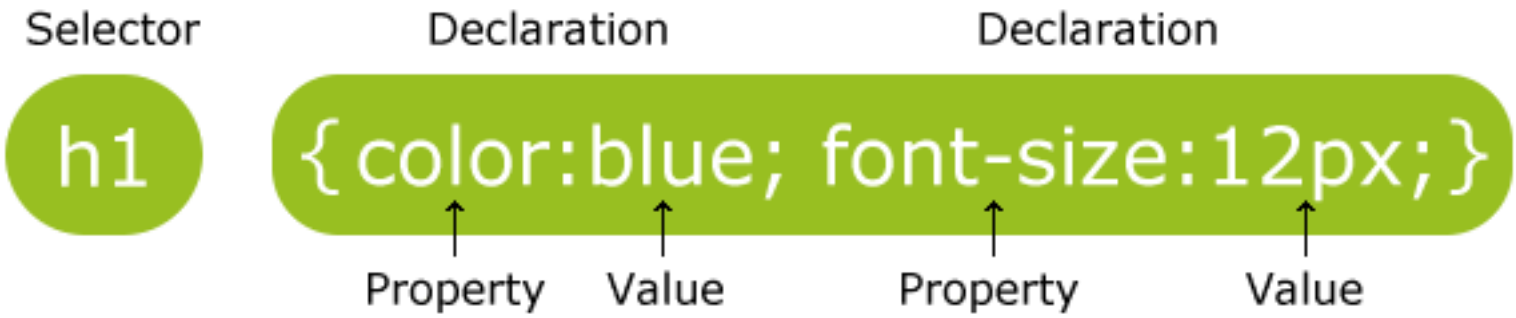
```
<link rel=”stylesheet” href=”misestilos.css”>
```

Con la línea anterior le decimos al navegador que cargue el archivo misestilos.css, que contendrá todos los estilos necesarios para presentar el documento en pantalla.

6. CSS y HTML

Funcionamiento básico de las reglas CSS

CSS define una serie de términos que permiten describir cada una de las **partes que componen los estilos CSS**. El siguiente esquema muestra las partes que forman un estilo CSS muy básico:



6. CSS y HTML

Funcionamiento básico de las reglas CSS

Regla: Compuesta por una parte denominada "**selectores**", un símbolo de "**llave de apertura**" ({), otra parte denominada "**declaraciones**" y, por último, un símbolo de "**llave de cierre**" (}).

Selector: indica el **elemento o elementos** HTML a los que se aplica la regla CSS.

Declaración: especifica los estilos que se aplican a los elementos. Compuesta por **una o más propiedades** CSS.

Propiedad: permite modificar el aspecto de una **característica** del elemento.

Valor: indica el **nuevo valor** de la característica modificada en el elemento.

6. CSS y HTML

Selectores

Para no extendernos demasiado, en este punto, veremos el uso de los selectores CSS **más importantes.** Si alguno de ellos necesita una explicación más detallada se puede encontrar en http://librosweb.es/css/.

Es importante tener en cuenta que a un mismo elemento HTML se le pueden asignar infinitas reglas CSS y cada regla CSS puede aplicarse a un número infinito de elementos.

A continuación, veremos los selectores más importantes de la versión 2.1 de CSS.

6. CSS y HTML

Selector universal *

```
* { margin: 0px; padding: 0px; }
```

Normalmente, para la mayoría de los elementos, necesitamos personalizar los márgenes o, simplemente, mantenerlos al mínimo. Algunos elementos, por defecto, tienen márgenes que son diferentes de cero y, en la mayoría de los casos, demasiado amplios.. Para **evitar tener que repetir estilos constantemente**, podemos utilizar el selector universal.

Con la regla indicada anteriormente, nos aseguramos de que todo elemento tendrá un margen interno y externo de 0 píxeles. De ahora en adelante, **sólo necesitaremos modificar los márgenes de los elementos que queremos que sean mayores** que cero.

6. CSS y HTML

Selector de tipo o etiqueta

Selecciona **todos** los elementos de la página cuya **etiqueta HTML coincide** con el valor del selector. El siguiente ejemplo selecciona todos los párrafos de la página:

```
p { ... }
```

Si se quiere aplicar los mismos estilos a **dos etiquetas diferentes**, se pueden encadenar los selectores. Para ello, se incluyen todos los selectores separados por una **coma** (,).

```
h1, h2, h3 { color: #8A8E27; font-weight: normal; }
```

6. CSS y HTML

Selector descendente

Selecciona los elementos que se encuentran **dentro de otros elementos**. Un elemento es descendiente de otro cuando se encuentra **entre las etiquetas de apertura y de cierre** del otro elemento.

El selector del siguiente ejemplo selecciona todos los elementos `<span>` de la página que se encuentren dentro de un elemento `<p>`:

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

6. CSS y HTML

Selector descendente

Si el código HTML de la página es el siguiente:

```
<p>
   <span>texto1</span>
   <a href=""><span>texto2</span></a>
</p>
```

El selector p span selecciona **tanto texto1 como texto2**. El motivo es que en el selector descendente, un elemento **no tiene que ser "hijo directo" de otro**. La única condición es que un elemento debe estar dentro de otro elemento.

A los elementos de la página que no están dentro de un elemento <p>, no se les aplica la regla CSS anterior.

6. CSS y HTML

Selector de clase .

Consiste en utilizar el **atributo class** de HTML sobre ese elemento para indicar directamente la regla CSS que se le debe aplicar.

```
<p class="destacado">Lorem ipsum ...</p>
```

A continuación, se crea en el archivo CSS una nueva regla llamada destacado con todos los estilos que se van a aplicar al elemento. Para que el navegador no confunda este selector con los otros tipos de selectores, se prefija el valor del atributo class con **un punto** (.), tal y como muestra el siguiente ejemplo:

```
.destacado { color: red; }
```

6. CSS y HTML

Selector de clase .

```
<p class="destacado">Lorem ipsum dolor sit amet...</p>
<p>sed lacus et <a href="#" class="destacado">est adipiscing</a> accumsan</p>
```

Para **restringir el alcance del selector** de clase, por ejemplo aplicar sólo a los párrafos, combinamos selectores:

```
p.destacado { color: red }
```

También es posible aplicar los **estilos de varias clases** sobre un mismo elemento

```
<p class="especial destacado error">Párrafo de texto...</p>
```

Al párrafo anterior se le aplican los estilos definidos en las reglas .especial, .destacado y .error.

6. CSS y HTML

Selector de ID

Permite seleccionar **un elemento de la página** a través del valor de su **atributo id**. Sólo seleccionan un elemento de la página, ya que el valor del **atributo id no se puede repetir** en dos elementos diferentes de una misma página.

La sintaxis de los selectores de ID es muy parecida a la de los selectores de clase, salvo que se utiliza el símbolo de la **almohadilla (#),** en lugar del punto (.)

```
#destacado { color: red; }

<p>Primer párrafo</p>
<p id="destacado">Segundo párrafo</p>
<p>Tercer párrafo</p>
```

6. CSS y HTML

Selector de hijos >

Se trata de un selector similar al selector descendente, pero muy diferente en su funcionamiento. Se utiliza para seleccionar **un elemento que es hijo directo de otro** elemento y se indica mediante el **"signo de mayor que" (>)**

```
p > span { color: blue; }
<p><span>Texto1</span></p>
<p><a href="#"><span>Texto2</span></a></p>
```

En el ejemplo anterior, el primer párrafo cumple con la condición del selector. Sin embargo, el segundo no por no ser hijo directo.

6. CSS y HTML

Selector adyacente +

El selector adyacente utiliza el signo + y su sintaxis es:

```
elemento1 + elemento2 { ... }
```

La explicación del comportamiento de este selector no es sencilla, ya que selecciona todos los elementos de tipo **elemento2** que **cumplan las dos siguientes condiciones**:

- elemento1 y elemento2 deben ser **hermanos**, por lo que su elemento **padre debe ser el mismo**.
- elemento2 debe aparecer **inmediatamente después** de elemento1 en el código HTML de la página.

Veamos un ejemplo en la siguiente página

6. CSS y HTML

```
h1 + h2 { color: red }
<body>
    <h1>Titulo1</h1>
    <h2>Subtítulo</h2> ...
    <h2>Otro subtítulo</h2>
    ...
</body>
```

Los estilos de h1 + h2 **se aplican al primer** elemento <h2> de la página, **pero no al segundo** <h2>, ya que:

- El padre de <h1> es <body>, el **mismo padre** que el de los dos elementos <h2> (cumple primera condición)
- El primer elemento <h2> aparece **justo después** del elemento <h1>, pero por el contrario, el segundo elemento <h2> no aparece justo después del elemento <h1>, por lo que el segundo h2 no cumple la segunda condición.

6. CSS y HTML

Selector general de elementos hermanos

Su sintaxis es **`elemento1 ~ elemento2`** y selecciona el elemento2 que es hermano de elemento1 y se encuentra detrás en el código HTML.

```
h1 + h2 { ... } /* selector adyacente */
h1 ~ h2 { ... } /* selector general de hermanos */
<h1>...</h1>
<h2>...</h2>
<p>...</p>
<div>
     <h2>...</h2>
</div>
<h2>...</h2>
```

(h1 + h2) solo selecciona el primer elemento <h2> de la página, ya que es el único que cumple que es hermano de <h1> y se encuentra justo detrás en el código HTML.

(h1 ~ h2) selecciona todos los elementos <h2> de la página salvo el segundo. Aunque el segundo <h2> se encuentra detrás de <h1> no son elementos hermanos, ya que no tienen el mismo elemento padre.

6. CSS y HTML

Selector de atributos

Los selectores de atributos permiten seleccionar elementos en **función de sus atributos y/o valores** de esos atributos.

Los cuatro tipos de selectores de atributos son:

[nombre_atributo], selecciona los elementos con atributo llamado nombre_atributo **independientemente de su valor**.

[nombre_atributo=valor], selecciona los elementos con atributo llamado nombre_atributo con un valor **igual a valor**.

[nombre_atributo~=valor], selecciona elementos con atributo nombre_atributo y al menos **uno de los valores es valor**.

[nombre_atributo|=valor], selecciona elementos con atributo nombre_atributo, cuyo valor es una serie de palabras separadas con guiones, pero que **comienza con valor**.

6. CSS y HTML

Selector de atributos

/* Todos los enlaces que tengan un atributo "class" */

```
a[class] { color: blue; }
```

/* Todos los enlaces que tengan un atributo "class" con el valor "externo" */

```
a[class="externo"] { color: blue; }
```

/* Todos los enlaces que apunten al sitio "http://www.ejemplo.com" */

```
a[href="http://www.ejemplo.com"] { color: blue; }
```

/* Todos los enlaces que tengan un atributo "class" en el que **al menos uno** de sus valores sea "externo" */

```
a[class~="externo"] { color: blue; }
```

/* Todos los elementos de la página cuyo atributo "lang" sea **igual a "en"**, es decir, todos los elementos en inglés */

```
*[lang=en] { ... }
```

/* Todos los elementos de la página cuyo atributo "lang" **empiece por "es"**, es decir, **"es", "es-ES", "es-AR", etc.** */

```
*[lang|="es"] { color : red }
```

6. CSS y HTML

Pseudo-clases

El concepto pseudo-clase se introdujo para permitir la selección de elementos sobre la base de la **información que se encuentra fuera de la estructura del documento**, o que no se puede expresar con los otros selectores simples.

Una pseudo-clase se compone siempre de **"dos puntos" (:) seguido del nombre de la pseudo-clase** y, opcionalmente, por un valor entre paréntesis.

Las pseudo-clases pueden ser **dinámicas**, es decir, un elemento puede adquirir o perder una pseudo-clase, mientras que un usuario interactúa con el documento.

6. CSS y HTML

Pseudo-clases

A continuación se muestran algunos ejemplos:

- **:link** permite aplicar estilos para los enlaces que aún no han sido visitados.
- **:visited** aplica estilos a los enlaces que han sido visitados anteriormente.
- **:focus** estilos que se aplican al enlace cuando este tiene el foco (acepta eventos de ratón o de teclado).
- **:hover** estilos que muestra el enlace cuando el usuario posiciona el puntero del ratón sobre el enlace.
- **:active** estilos que se aplican al enlace cuando el usuario está pinchando sobre el enlace (el tiempo durante el que se aplica este estilo es muy breve).

6. CSS y HTML

Pseudo-clases

Las pseudo-clases **:link** y **:visited** solamente están definidas para los **enlaces**, pero las pseudo-clases **:hover** y **:active** se definen para **todos los elementos** HTML.

```
a:hover { text-decoration: none; }
```

En este ejemplo se elimina el subrayado del enlace cuando se sitúa el ratón sobre él.

6. CSS y HTML

Pseudo-clases

```
<p>fuera</p>
<div>
  <p>Mi texto1</p>
  <p>Mi texto2</p>
  <p>Mi texto3</p>
  <p>Mi texto4</p>
</div>
```

El código anterior contiene cinco elementos <p>, uno que es hijo del documento y cuatro que son hermanos entre sí e hijos del mismo elemento .

Usando pseudo-clases podemos **referenciar un elemento específico** sin importar cuánto conocemos sobre sus atributos y el valor de los mismos:

```
p:nth-child(1){ background: #999999; }
```

6. CSS y HTML

Pseudo-clases

Si quisiéramos restringir la selección **sólo a los hijos del **, haríamos lo siguiente:

```
div p:nth-child(1){ background: #999999; }
```

También se pueden utilizar las palabras clave even y odd como índice para seleccionar los hijos **pares o impares** respectivamente:

```
div p:nth-child(even){ background: #999999; }
```

En este caso, se seleccionarán los párrafos 2 y 4.

6. CSS y HTML

Pseudo-clases para la selección de hijos y hermanos

Psudo-clase	descripción
:nth-child(n)	Selecciona el enésimo hijo de su padre
:nth-last-child(n)	Selecciona el enésimo hijo de su padre contando desde el último
:nth-of-type(n)	Selecciona el enésimo hermano de su tipo
:nth-last-of-type(n)	Selecciona el enésimo hermano de su tipo comenzando desde el último
:first-child	Selecciona el primer hijo de su padre
:last-child	Selecciona el último hijo de su padre
:first-of-type	Selecciona el primer hermano de su tipo
:last-of-type	Selecciona el último hermano de su tipo
:only-child	Selecciona los que sean hijos únicos
:only-of-type	Selecciona los que sean los únicos hermanos de su tipo
:empty	Selecciona los elementos que no tienen hijos. Si un elemento contiene sólo texto no se considera vacío.

4. CSS3

Pseudo-clases

Otras pseudo-clases nuevas que también son importantes:

`:enabled, :disabled` Selecciona elementos de interfaz de usuario (formularios) según estén **activado o desactivado**.

`:checked` Selecciona elementos de (radio botones o casillas de verificación) que están en estado **checked**

La pseudo-clase :not() se utiliza realizar una negación:

```
:not(p) { margin: 0px; }
```

Este selector asignará un margen de 0 píxeles a todos los elementos del documento **excepto** los elementos <p>. Podemos utilizar cualquier selector válido:

```
:not(.mitexto) –> Todos menos los de la clase mitexto
:not(#menu) ☾ Todos excepto el id menu
```

6. CSS y HTML

Pseudo-elementos

Define elementos especiales llamados "pseudo-elementos"

- **::first-line** selecciona la primera línea del texto.
- **::first-letter** selecciona la primera letra del texto.
- **::after** selecciona el elemento y se sitúa detrás
- **::before** selecciona el elemento y se sitúa delante
- **::selection** selecciona la porción de un elemento seleccionada por el usuario.

```
p::first-letter {
  color: #ff0000;
  font-size: 200%;
}
```

6. CSS y HTML

Propiedades CSS

La potencia de CSS la encontramos en las propiedades disponibles para modificar el aspecto de los elementos. A continuación, las propiedades de uso más habitual.

CSS que afecta al texto	**color**: red; **font-family**: 20px; **font-size**: 20px; **font-weight**: bold; **text-align**: center; **line-height**: 50px; **letter-spacing**: 2px; **word-spacing**: 5px;	Color del texto Tipografía utilizada. Tamaño del texto (expresado en píxeles "10px", en tamaño original de la fuente "2em"..) Estilo de la fuente ("bold", "normal", "italic"..) Alineación de un texto dentro de un bloque (center, left, right o justify). Altura de una línea de texto. Sirve para centrar un texto verticalmente, indicando como valor la altura del bloque. Espacio que hay entre cada letra (se puede definir en píxeles "px" o en cantidades relativas "em"). Espacio que hay entre palabras (se puede definir en píxeles "px" o en cantidades relativas "em").
CSS que afecta a los bloques	**background-color**: red; **padding**: 20px; **margin**: 20px; **position**: relative; **left**: 10px; **top**: 10px; **float**: left **clear**: both	Color de fondo de un bloque. Espacio que hay entre el contenido de un bloque y su borde (sería el espacio interno). Espacio que hay entre bloques (espacio externo). Define el tipo de posicionamiento de un bloque (relative, absolute o fixed). Posición horizontal de un elemento. Posicion vertical de un elemento. Posiciona bloques a la izquierda (left) o derecha (right) de otros. Elimina los float declarados con anterioridad y que aún perduran.

CSS que afectan a los enlaces	**text-decoration**: none;
	cursor: pointer;
	a {
	a:hover {
		Elimina el subrayado de los enlaces (o la viñetas en las listas).
		Transforma el cursor en la mano con el dedo extendido (al usar junto a **a:hover**).
		Selector de CSS que identifica a los enlaces que contenga la página.
		Selector de CSS que identifica el momento en el que el cursor se coloca encima de un enlace.

6. CSS y HTML

Validador CSS

Al igual que ocurre con HTM, disponemos de un validador para CSS que nos **informará de los errores** que tenemos en nuestros archivos de estilos. Este validador se encuentra en la url http://jigsaw.w3.org/css-validator/.

Es **altamente recomendable pasar el validador** antes de probar nuestro sitio web, ya que, muchas veces los estilos no se aplicarán como nosotros esperamos porque tenemos errores, y el validador nos **puede ahorrar mucho tiempo** a la hora de encontrar dónde está el problema.