

Tema 1. Lenguajes de representación en el lado del cliente.

Desarrollo de Tecnologías Web:
Tecnologías en el Cliente (DTWTC)

*Master Universitario en Ingeniería de Servicios y
Aplicaciones Web*

Índice

- HTML5

- Nuevo concepto de documento
 - Novedades y mejoras sobre HTML4.01

- CSS3

- Novedades y mejoras sobre CSS2.1

HTML 5

- ¿Por qué HTML5?
 - Para mejorar el dinamismo que ofrecía HTML4.01 y XHTML1.0
 - Para añadir semántica al contenido de manera más sencilla
 - Para poder acceder más fácilmente a diferentes tipos de dispositivos
 - Para acelerar el desarrollo de aplicaciones web

HTML 5

- ¿Qué aporta HTML5?
 - Nuevos elementos semánticos y estructurales
 - Nuevos elementos y APIs para trabajar con contenido multimedia
 - Nuevas características funcionales: visualización, validación, organización
 - Nuevas características para dotar al navegador de independencia y poder trabajar, por ejemplo, de forma offline.
 - Integración más simple de HTML, CSS y JS

HTML5

- ¿Qué veremos en este tema? Un resumen de los principales elementos de novedad que aporta HTML5:
 - Estructura y semántica
 - Formularios
 - Drag & Drop
 - Almacenamiento offline
 - Geolocalización
 - Websockets
 - Web Workers
 - Microdata y Custom Data

HTML5. Estructura y semántica

- Un documento HTML contiene etiquetas que:
 - Significan algo
 - Se representan por defecto de algún modo
- La etiqueta como elemento de significado es escasa → Se recurre a los atributos **id** y **class** para añadir semántica adicional.
- La semántica, en muchos casos, tiene que ver con la funcionalidad de la etiqueta en la página: cabecera, menú, pie de página, ...
- HTML5 aporta etiquetas para mejorar la semántica de forma combinada con la estructura del documento
- En conclusión, **aparecen y desaparecen etiquetas**

HTML5. Lo primero, la cabecera

■ Ejemplo HTML4.01

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

<html>
  <head>
    <meta http-equiv=content-type content="text/html;
      charset=UTF-8">
    <title>Hello World! HTML 4 Strict</title>
  </head>
  <body>
    <p>Hello World!</p>
  </body>
</html>
```

HTML5. Lo primero, la cabecera

■ Ejemplo XHTML1.0

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

<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="content-type" content="text/html;
      charset=UTF-8" />
    <title>Hello World! XHTML 1 Strict</title>
  </head>
  <body>
    <p>Hello World!</p>
  </body>
</html>
```


HTML5. Lo primero, la cabecera

■ Ejemplo HTML5

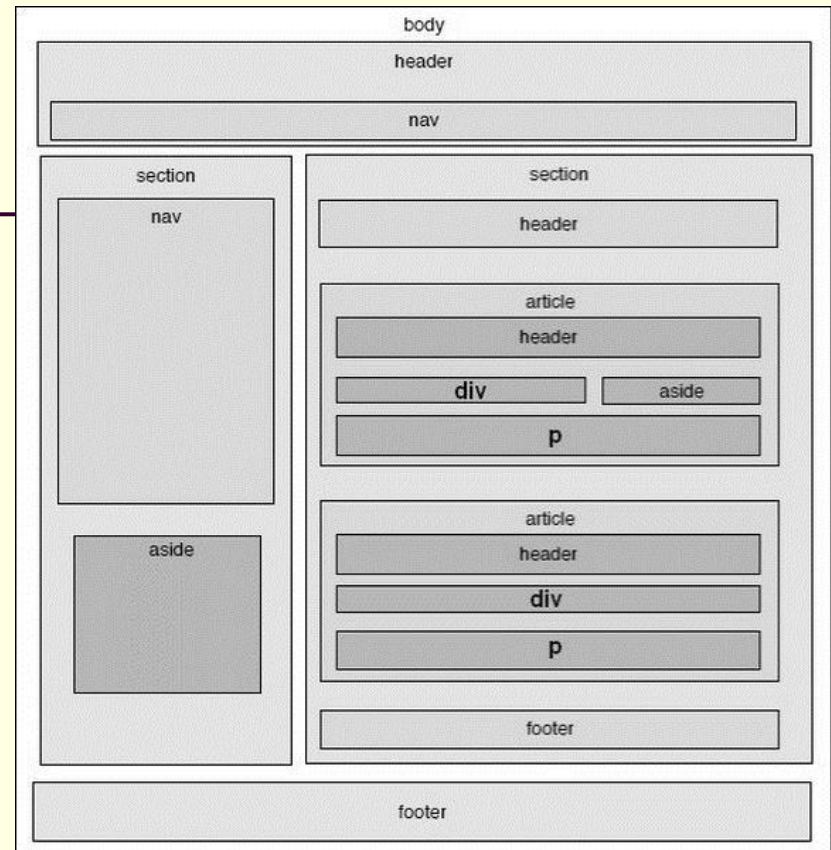
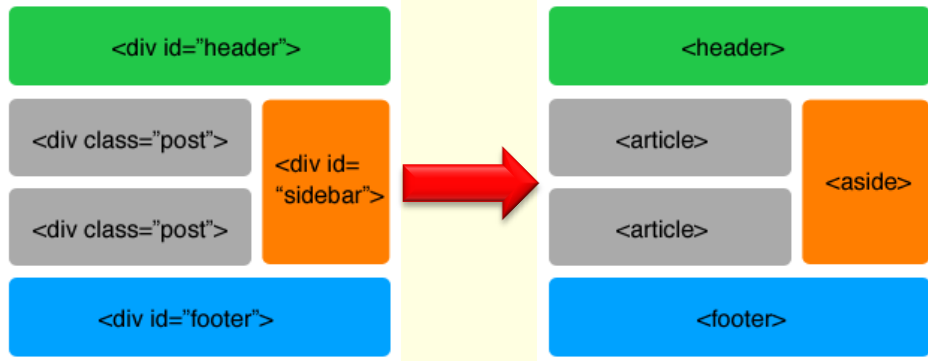
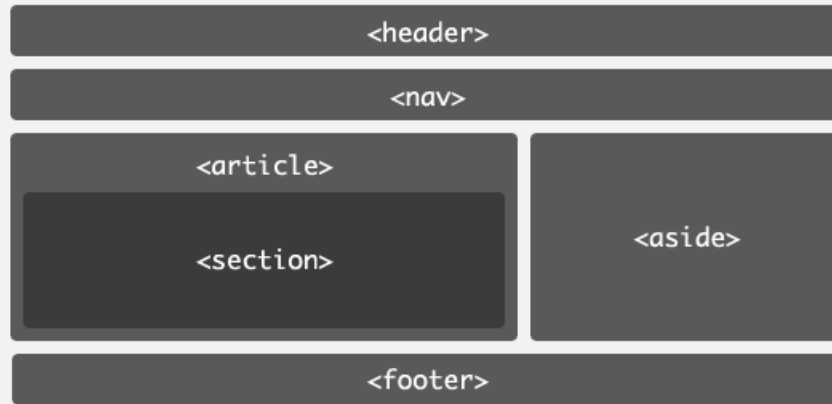
```
<!DOCTYPE html>

<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <title>Hello World!</title>
  </head>
  <body>
    <p>Hello World!</p>
  </body>
</html>
```

HTML5

- Nuevas etiquetas semánticas / estructurales
 - **<section>** - agrupación de contenido
 - **<article>** - contenido independiente
 - **<header>** - sección inicial del documento
 - **<footer>** - sección final del documento
 - **<nav>** - menú o barra de navegación
 - **<aside>** - posicionamiento lateral

HTML5



HTML5. Evolución de etiquetas

■ NUEVAS

<canvas>	<summary>
<audio>	<figure>
<video>	<figcaption>
<source>	<footer>
<embed>	<header>
<track>	<mark>
<datalist>	<meter>
<keygen>	<nav>
<output>	<progress>
<article>	<ruby>
<aside>	<rt>
<bdi>	<rp>
<command>	<section>
<details>	<time>
<dialog>	<wbr>

■ ELIMINADAS

- <acronym>
- <applet>
- <basefont>
- <big>
- <center>
- <dir>
-
- <frame>
- <frameset>
- <noframes>
- <strike>
- <tt>

HTML5. Formularios

ELEMENTO	CARACTERISTICA	VALORES
FORM	Atributos	autocomplete, novalidate, oninput
INPUT	Atributo Type	text, password, submit, email, search, url, tel, number, range, date, week, month, time, datetime, datetime-local, color
PROGRESS	Atributos	value, max
METER	Atributos	min, max, value, low , high, optimum
OUTPUT	Comentario	Se maneja con Javascript o con el evento oninput en el form

ATRIBUTO	CARACTERISTICA	VALORES
LIST y DATALIST	Ejemplo	<pre><input list="browsers" name="browser"> <datalist id="browsers"> <option value="Internet Explorer"> <option value="Firefox"> ... </datalist></pre>
Autocomplete, Novalidate	Aplicable en	FORM
autocomplete, autofocus, form, formaction, formenctype, formmethod, formnovalidate, formtarget, height and width, list, min and max, multiple, pattern (regex), placeholder, required, step	Aplicable en	INPUT

HTML5. Drag & Drop

- HTML5 permite trabajar con Drag & Drop en nativo, evitando así el uso de otras librerías.
- HTML5 permite:
 - Mover elementos internamente en una página
 - Interactuar con elementos externos
 - Enviando objetos al sistema
 - Recibiendo objetos del sistema
- Es necesario apoyarse en JavaScript para controlar el proceso

HTML5. Drag & Drop

- Proceso de creación de elementos DnD
 - Definición del contenido arrastrable
 - Control de eventos de arrastre:
 - dragstart
 - drag
 - dragenter
 - dragleave
 - drop
 - dragend
 - Definición y creación de los manejadores que actuarán ante cada evento

HTML5. Drag & Drop

- Un elemento es arrastrable cuando se le añade el atributo **draggable="true"**
 - El navegador no hace nada por defecto, tan solo muestra una copia semitransparente del objeto cuando se arrastra por la ventana
- El evento dragstart pone en marcha la operación de arrastre. Hay que definir un manejador para actuar sobre el:

```
function handleDStrt(e) {  
    // Aquí se sitúa el código para reaccionar al evento  
}  
...  
elemento.addEventListener('dragstart', handleDStrt, false); // opción 1  
<elemento ondragstart="handleDStrt(event);">...</elemento> // opción 2
```


HTML5. Drag & Drop

- dragenter, dragover y dragleave permiten controlar el proceso de arrastre, de nuevo usando manejadores (igual que con dragstart)
- drop y dragend se usan para finalizar el proceso de arrastre (con manejadores)
- El objeto destino debe permitir en dragover y drop el arrastre con **event.preventDefault()**
- Se puede evitar que el navegador propague las acciones de arrastre por defecto tras la definición de los controladores usando el comando **event.stopPropagation()**

HTML5. Drag & Drop

- El control del arrastre se hace con la propiedad **dataTransfer** del evento que se le pasa al navegador.
 - Permite asignar datos con `.setData()` (en el manejador de dragstart)
 - Permite recoger datos con `.getData()` (en el manejador de drop)
- El objeto **event.target** en los manejadores permite acceder al objeto que está siendo manejado (no solo el origen, sino también el destino). Es equivalente a **this** si se usa en la definición del manejador del evento.

HTML5. Almacenamiento offline

- También se le conoce como Web Storage
- Útil para crear aplicaciones cuando no existe conexión con un servidor
- Consisten, básicamente, en una mejora de las cookies
- Doble uso:
 - Preservar la información durante la sesión (**sessionStorage**)
 - Preservar la información todo el tiempo que el usuario desee (**localStorage**)

HTML5. Almacenamiento offline

- sessionStorage
 - Conservará los datos solo durante la duración de la sesión de una página. La información almacenada a través de este mecanismo es solo accesible desde una única ventana o pestaña y es preservada hasta que la ventana es cerrada.
- localStorage
 - Los datos son grabados de forma permanente y se encuentran siempre disponibles para la aplicación que los creó.
- Muchos navegadores trabajan correctamente con esta API cuando la fuente es un servidor real y no el sistema de archivos del cliente

HTML5. Almacenamiento offline

- sessionStorage y localStorage trabajan con el mismo API (mismos métodos y atributos):
 - sessionStorage.setItem (clave, valor)
 - sessionStorage[clave] = valor
 - sessionStorage.clave = valor
 - getItem (clave)
 - key (i) → obtiene el valor de la clave en posición i
 - removeItem (clave)
 - clear () → vacía el espacio de almacenamiento
- localStorage debe resolver un problema, puesto que está activo para todas las ventanas:
 - Comunicación entre ventanas
 - Mantener información actualizada en cada ventana

HTML5. Almacenamiento offline

■ Particularidades de localStorage

- storage. Evento disparado por la ventana cada vez que ocurre un cambio en el espacio de almacenamiento para alertar a todas las ventanas (**no en todos los navegadores**)
- Hay que implementar una función de callback y añadir un manejador que se ejecute en todas las ventanas cuando se produzca el evento

```
window.addEventListener("storage", procesar, false);  
  
...  
  
function procesar() {  
    ...  
}
```

HTML5. Geolocation

- La API Geolocation se diseñó para que los navegadores pudieran obtener información física real sobre la ubicación del usuario.
- Utiliza triangulación de red y GPS, en lugar de las anteriores bases de datos de direcciones IP
- Se basa en un uso muy simple de 3 funciones:
 - `getCurrentPosition(ubicación, error, configuración)`
 - `watchPosition(ubicación, error, configuración)`
 - `clearWatch(id)`

HTML5. Geolocation

- `getCurrentPosition` (ubicación, error, configuración)
 - Solo es obligatorio el primer parámetro
 - Devuelve un objeto **Position** con dos atributos: **coords** y **timestamp**
- `watchPosition` (ubicación, error, configuración)
 - Funciona igual que el anterior, pero revisa la ubicación con cierta frecuencia (como `setInterval`)
- `clearWatch` (id)
 - Detiene la vigilancia de `watchPosition`

HTML5. WebSockets

- Están considerados como parte del API de comunicaciones que ofrece HTML5
- Es un protocolo bidireccional que permite conectar 2 o más clientes a través de un servidor
- Son conexiones TCP que no envían cabeceras HTTP (son más ligeras)
- La conexión es persistente (push vs pull)
- Permite desarrollo de aplicaciones con necesidad de comunicaciones en tiempo real

HTML5. WebSockets

- Servidor WS

- Es necesario programar un servidor WS para que funcione esta funcionalidad

- Elementos del API. Constructor

- WebSocket(url)

- Este constructor inicia una conexión entre la aplicación y el servidor WS apuntado por el atributo url. Retorna un objeto WebSocket referenciando esta conexión.

HTML5. WebSockets

- Elementos del API. Métodos

- send(datos)

- Envía un mensaje al servidor WS. datos suele ser una cadena de texto, pero puede ser información con un formato (XML, JSON) que permita el envío de información estructurada o semi-estructurada.

- close()

- Cierra la conexión

HTML5. WebSockets

■ Elementos del API. Propiedades

■ url

- URL a la cual la aplicación está conectada.

■ Protocol

- Esta propiedad retorna el sub protocolo usado.

■ readyState

- Estado de la conexión: 0 (conexión no establecida), 1 (conexión abierta), 2 (la conexión está siendo cerrada), y 3 significa (conexión cerrada).

■ bufferedAmount

- Cantidad de datos requeridos pero aún no enviados al servidor.

HTML5. WebSockets

- Elementos del API. Eventos

- Open

- Se dispara cuando se abre la conexión

- Message

- Se dispara cuando un mensaje proveniente del servidor se encuentra disponible.

- Error

- Se disparar cuando ocurre un error.

- Close

- Se dispara cuando se cierra la conexión

- Ejemplos:

<http://html5demos.com/web-socket>

<https://code.google.com/p/phpwebsocket/>

HTML5. Web Workers

- Web Workers es una API diseñada con el propósito específico de convertir Javascript en un lenguaje multiproceso.
- En HTML5, podemos ejecutar código costoso en background mientras el código principal sigue siendo ejecutado en la página web.
- Un Worker se constituye en un archivo javascript separado, y se comunica con el proceso central a través de mensajes
- Habitualmente, la información intercambiada son peticiones y respuestas

HTML5. Web Workers

- Elementos del API

- Worker(códigoURL)

- Método que retorna un objeto Worker ubicado en el lugar que especifica el atributo códigoURL.

- postMessage(mensaje)

- Método que envía un mensaje hacia o desde el código del trabajador.

- message

- Evento que escucha por mensajes enviados al código principal o al trabajador. Utiliza la propiedad data para obtener el mensaje enviado.

HTML5. Web Workers

- Elementos del API

- `terminate()`

- Detiene el Worker desde el módulo principal.

- `close ()`

- Detiene el Worker desde el propio Worker.

HTML5. Web Workers

■ Shared Workers

■ `sharedWorker(códigoURL)`

- Constructor de un objeto `SharedWorker`.
`códigoURL` es la ruta del Worker.

■ `port`

- Valor del puerto de la conexión con el Worker, es el atributo del worker sobre el que se basa la comunicación (`addEventListener`, `start`, ...)

■ `connect`

- Evento que se dispara en el Worker cuando se solicita una nueva conexión.

■ `start()`

- Método que inicia el envío de mensajes.

HTML5. Microdata y Custom Data

- Los microdatos y los datos configurados por el usuario son mejoras de HTML5 que responden a la necesidad de añadir semántica específica a los contenidos de los documentos HTML
- En versiones anteriores, el uso y abuso de los atributos class y rel permitía resolver esta necesidad de manera más o menos compleja
- La primera solución importante llegó de la mano del RDF y la web semántica, pero no ha tenido mucho éxito entre los desarrolladores

HTML5. Microdata y Custom Data

- HTML5, a través de la especificación WHATWG introduce el concepto de microdata, que proporciona una nueva sintaxis para marcar semántica estructurada adicional.
- HTML5 también permite atributos custom data que permiten ir más allá del estándar cuando los desarrolladores lo necesitan
- Actualmente los navegadores soportan esta nueva semántica, aunque el API del DOM aún no está preparado para manejarlo. No obstante, javascript permite su uso.

HTML5. Microdata y Custom Data

- Microdata estándar: <http://schema.org>
- Ejemplo de microdata

```
<div>
  <h1>Avatar</h1>
  <span>Director: James Cameron (born August 16, 1954)</span>
  <span>Science fiction</span>
  <a href="../../movies/avatar-theatrical-trailer.html">Trailer</a>
</div>
```

```
<div itemscope itemtype="http://schema.org/Movie">
  <h1>Avatar</h1>
  <span>Director: James Cameron (born August 16, 1954) </span>
  <span>Science fiction</span>
  <a href="../../movies/avatar-theatrical-trailer.html">Trailer</a>
</div>
```

HTML5. Microdata y Custom Data

■ Ejemplo de microdata

```
<div itemscope itemtype ="http://schema.org/Movie">
  <h1 itemprop="name">Avatar</h1>
  <span>Director: <span itemprop="director">James Cameron</span> (born
    August 16, 1954)</span>
  <span itemprop="genre">Science fiction</span>
  <a href=" ../movies/avatar-theatrical-trailer.html" itemprop="trailer">
    Trailer</a>
</div>
```

```
<div itemscope itemtype ="http://schema.org/Movie">
  <h1 itemprop="name"&g;Avatar</h1>
  <div itemprop="director" itemscope itemtype="http://schema.org/Person">
    Director: <span itemprop="name">James Cameron</span> (born
      <span itemprop="birthDate">August 16, 1954)</span>
  </div>
  <span itemprop="genre">Science fiction</span>
  <a href=" ../movies/avatar-theatrical-trailer.html" itemprop="trailer">
    Trailer</a>
</div>
```

HTML5. Otros elementos

- HTML5 también dispone de otros elementos, que por motivos de compatibilidad con otras asignaturas quedarán fuera de este curso:
 - Canvas. Se usa para representar gráficos, vía un lenguaje de script
 - Audio. Permite incorporar elementos de audio
 - Vídeo. Permite incorporar elementos de vídeo
- HTML5 también permite el uso de otras APIs avanzadas como:
 - Acceso a ficheros
 - Elementos de comunicaciones
 - Base de datos local (IndexedDB)

CSS3

- CSS3 nace con el objetivo de facilitar la integración de CSS, javascript y HTML, al igual que HTML5.
- Pretende sustituir a los códigos javascript difíciles de implementar para dotar de dinamismo visual a las páginas HTML
- No sólo cubre estilos y diseño web, sino también forma y movimiento
- Se presenta en módulos que permiten proveer una especificación estándar por cada aspecto involucrado en la presentación.

CSS3









- En la introducción hemos hecho un repaso de la forma en la que se trabaja con estilos, y en particular con CSS
- En este tema veremos:
 - Aspectos novedosos que aporta CSS3 sobre versiones anteriores
 - Contribuciones que hace CSS3 a HTML5
 - Propiedades que simplifican el trabajo de los desarrolladores

CSS3

- Módulos más importantes en CSS3:
 - Selectors
 - Box Model
 - Backgrounds and Borders
 - Text Effects
 - 2D/3D Transformations
 - Animations
 - Multiple Column Layout
 - User Interface

CSS3

■ Etiquetas propias de cada navegador: prefijos

Versión	Navegador	Prefijo	Ejemplo
Firefox 3.6+		-moz	-moz-linear-gradient
Chrome, Safari 4	 	-webkit	-webkit-gradient
Chrome 10+, Safari 5.1+	 	-webkit	-webkit-linear-gradient
Opera 11.10+		-o	-o-linear-gradient
Internet Explorer 6-9		filter	filter
Internet Explorer 10+		-ms	-ms-linear-gradient
Estándar W3C	-	<i>Ninguno</i>	linear-gradient
Navegadores antiguos	-	<i>Sin soporte</i>	No soportan etiquetas de CSS3.

CSS3. Nuevos elementos

■ Bordes:

- border-radius
- box-shadow: h-shadow v-shadow blur spread color inset;
- border-image: source slice width outset repeat;

■ Fondos:

- background-size: length | percentage | cover | contain;
- background-origin: padding-box | border-box | content-box;
- Background-clip: border-box | padding-box | content-box;

CSS3. Nuevos elementos

- Efectos de texto:
 - hanging-punctuation
 - punctuation-trim
 - text-align-last
 - text-emphasis
 - text-justify
 - text-outline
 - text-overflow
 - text-shadow
 - text-wrap
 - word-break
 - word-wrap

CSS3. Nuevos elementos

■ Fuentes:

■ Regla @font-face

```
@font-face
{
  font-family: myFirstFont;
  src: url(sansation_light.woff);
}
```

■ Transformaciones 2D:

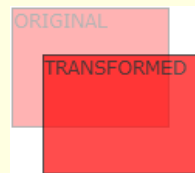
■ translate()

■ rotate()

■ scale()

■ skew()

■ matrix()



CSS3. Nuevos elementos

■ Transformaciones 3D:

- transform-origin: x-axis y-axis z-axis;
 - x-axis|y-axis: left | center | right | length | %
 - z-axis: length
- transform-style: flat | preserve-3d;
- perspective: number | none;
- perspective-origin: x-axis y-axis;
 - x-axis|y-axis: left | center | right | length | %
- backface-visibility: visible | hidden

CSS3. Nuevos elementos

- Transformaciones 3D:
 - rotateX()
 - rotateY()
 - translate3d()
 - translateX(), translateY(), translateZ()
 - scale3d()
 - scaleX(), scaleY(), scaleZ()
 - rotate3d(x,y,z,angle)
 - rotateX(), rotateY(), rotateZ()
 - perspective(n)
 - matrix3d(n,n,n,n,n,n,n,n,n,n,n,n,n,n,n,n)

CSS3. Nuevos elementos

- Transiciones. Para cambiar gradualmente de un estilo a otro. Se debe especificar:
 - Propiedad CSS a la que se añade el efecto
 - Duración del efecto

```
div {  
    transition: width 2s;  
}  
  
...  
  
div:hover {  
    width:300px;  
}
```


CSS3. Nuevos elementos

■ Animaciones:

- Se especifican a través de la regla @keyframes
- Definen las propiedades de un efecto en términos de % desde el inicio al final

■ Propiedades:

- animation-name
- animation-duration
- animation-timing-function
- animation-delay
- animation-iteration-count
- animation-direction: reverse | alternate | normal
- animation-play-state: running | paused

CSS3. Nuevos elementos

- Multiples columnas:
 - column-count
 - column-gap
 - column-rule-color
 - column-rule-style
 - column-rule-width
 - column-span
 - column-width
 - columns

CSS3. Nuevos elementos

■ Interfaz de usuario:

- `resize`: none | both | horizontal | vertical
- `box-sizing`: content-box | border-box | inherit
- `outline-offset`
- `appearance`: normal | icon | window | button | menu | field
- `box-sizing`: content-box | border-box | inherit
- `icon`: auto | url(URL) | inherit
- `nav-down`: auto | id | target-name | inherit
- `nav-index`: auto | id | target-name | inherit
- `nav-left`: auto | id | target-name | inherit
- `nav-right`: auto | id | target-name | inherit
- `nav-up`: auto | id | target-name | inherit

CSS3. Selectores anteriores

Selector	Example	Example description	CSS
<u>.class</u>	.intro	Selects all elements with class="intro"	1
<u>#id</u>	#firstname	Selects the element with id="firstname"	1
<u>*</u>	*	Selects all elements	2
<u>element</u>	p	Selects all <p> elements	1
<u>element,element</u>	div,p	Selects all <div> elements and all <p> elements	1
<u>element element</u>	div p	Selects all <p> elements inside <div> elements	1
<u>element>element</u>	div>p	Selects all <p> elements where the parent is a <div> element	2
<u>element+element</u>	div+p	Selects all <p> elements that are placed immediately after <div> elements	2
<u>[attribute]</u>	[target]	Selects all elements with a target attribute	2
<u>[attribute=value]</u>	[target=_blank]	Selects all elements with target="_blank"	2
<u>[attribute~=value]</u>	[title~=flower]	Selects all elements with a title attribute containing the word "flower"	2
<u>[attribute =value]</u>	[lang =en]	Selects all elements with a lang attribute value starting with "en"	2

CSS3. Selectores anteriores

Selector	Example	Example description	CSS
<u>:link</u>	a:link	Selects all unvisited links	1
<u>:visited</u>	a:visited	Selects all visited links	1
<u>:active</u>	a:active	Selects the active link	1
<u>:hover</u>	a:hover	Selects links on mouse over	1
<u>:focus</u>	input:focus	Selects the input element which has focus	2
<u>:first-letter</u>	p:first-letter	Selects the first letter of every <p> element	1
<u>:first-line</u>	p:first-line	Selects the first line of every <p> element	1
<u>:first-child</u>	p:first-child	Selects every <p> element that is the first child of its parent	2
<u>:before</u>	p:before	Insert content before the content of every <p> element	2
<u>:after</u>	p:after	Insert content after every <p> element	2
<u>:lang(language)</u>	p:lang(it)	Selects every <p> element with a lang attribute equal to "it" (Italian)	2

CSS3. Nuevos selectores

Selector	Example	Example description	CSS
<u>element1~element2</u>	p~ul	Selects every element that are preceded by a <p> element	3
<u>[attribute^=value]</u>	a[src^="https"]	Selects every <a> element whose src attribute value begins with "https"	3
<u>[attribute\$=value]</u>	a[src\$=".pdf"]	Selects every <a> element whose src attribute value ends with ".pdf"	3
<u>[attribute*=value]</u>	a[src*="w3schools"]	Selects every <a> element whose src attribute value contains the substring "w3schools"	3
<u>:first-of-type</u>	p:first-of-type	Selects every <p> element that is the first <p> element of its parent	3
<u>:last-of-type</u>	p:last-of-type	Selects every <p> element that is the last <p> element of its parent	3
<u>:only-of-type</u>	p:only-of-type	Selects every <p> element that is the only <p> element of its parent	3
<u>:only-child</u>	p:only-child	Selects every <p> element that is the only child of its parent	3
<u>:nth-child(n)</u>	p:nth-child(2)	Selects every <p> element that is the second child of its parent	3
<u>:nth-last-child(n)</u>	p:nth-last-child(2)	Selects every <p> element that is the second child of its parent, counting from the last child	3
<u>:nth-of-type(n)</u>	p:nth-of-type(2)	Selects every <p> element that is the second <p> element of its parent	3
<u>:nth-last-of-type(n)</u>	p:nth-last-of-type(2)	Selects every <p> element that is the second <p> element of its parent, counting from the last child	3

CSS3. Nuevos selectores

Selector	Example	Example description	CSS
<u>:last-child</u>	p:last-child	Selects every <p> element that is the last child of its parent	3
<u>:root</u>	:root	Selects the document's root element	3
<u>:empty</u>	p:empty	Selects every <p> element that has no children (including text nodes)	3
<u>:target</u>	#news:target	Selects the current active #news element (clicked on a URL containing that anchor name)	3
<u>:enabled</u>	input:enabled	Selects every enabled <input> element	3
<u>:disabled</u>	input:disabled	Selects every disabled <input> element	3
<u>:checked</u>	input:checked	Selects every checked <input> element	3
<u>:not(selector)</u>	:not(p)	Selects every element that is not a <p> element	3
<u>::selection</u>	::selection	Selects the portion of an element that is selected by a user	3

Referencia: http://www.w3schools.com/cssref/css_selectors.asp

CSS3. Navegadores

- Cada navegador interpreta CSS3 de forma distinta al estándar
- Especificación basada en prefijo
 - **-moz-** para Firefox.
 - **-webkit-** para Safari y Chrome.
 - **-o-** para Opera.
 - **-khtml-** para Konqueror.
 - **-ms-** para Internet Explorer.
 - **-chrome-** específico para Google Chrome.

CSS3. Navegadores

- Cada navegador interpreta CSS3 de forma distinta al estándar:

```
div {  
    width: 100px;  
    margin: 20px;  
    padding: 10px;  
    border: 1px solid #000000;  
  
    -moz-box-sizing: border-box;  
    -webkit-box-sizing: border-box;  
    box-sizing: border-box;  
}
```

Herramientas

- Herramientas para probar desarrollos:
 - <http://jsbin.com>
 - <http://jsfiddle.net/>
 - <http://cssdesk.com/>
 - <http://dabblet.com/>
 - <http://codepen.io/>
 - <http://www.jshint.com/>
 - <http://caniuse.com/>
 - http://en.wikipedia.org/wiki/Comparison_of_JavaScript-based_source_code_editors
 - <http://ace.c9.io/#nav=about>
 - <http://codemirror.net/>